WSF: Wireless Software Foundation

Generated by Doxygen 1.8.13

Contents

Chapter 1

Overview

Wireless Software Foundation implements an operating system abstraction layer to enhance system portability. This documentation covers API and internal event handling.

Legal

References

Ref.	Author	Date	Title
[1]	Arm	Oct 16, 2012	"Software Foundation API", 2009-0003, Revison 1.1

Definitions

Term	Definition
API	Applicaion Programming Interface
CS	Critical Section
WSF	Wireless Software Foundation software service and porting layer

Chapter 2

Module Documentation

2.1 STACK_HCI_API

Packet definitions

- #define HCI_CMD_HDR_LEN 3
- #define HCI_ACL_HDR_LEN 4
- #define HCI_ISO_HDR_LEN 4
- #define HCI_EVT_HDR_LEN 2
- #define HCI_EVT_PARAM_MAX_LEN 255
- #define HCI_ACL_DEFAULT_LEN 27
- #define HCI_PB_FLAG_MASK 0x3000
- #define HCI_PB_START_H2C 0x0000
- #define HCI_PB_CONTINUE 0x1000
- #define HCI_PB_START_C2H 0x2000
- #define HCI_HANDLE_MASK 0x0FFF
- #define HCI_HANDLE_NONE 0xFFFF
- #define HCI_TS_FLAG_MASK (1 << 14)
- #define HCI_DATA_LOAD_LEN_MASK 0x3FFF
- #define HCI_ISO_DL_MIN_LEN 4
- #define HCI_ISO_DL_MAX_LEN 8
- #define HCI_ISO_TS_LEN 4
- #define HCI_ISO_DL_SDU_LEN_MASK 0x0FFF
- #define HCI_ISO_DL_PS_MASK 0xC000

Packet types

- #define HCI_CMD_TYPE 0x01
- #define HCI_ACL_TYPE 0x02
- #define HCI_EVT_TYPE 0x04
- #define HCI_ISO_TYPE 0x05

Error codes

- #define HCI_SUCCESS 0x00
- #define HCI_ERR_UNKNOWN_CMD 0x01
- #define HCI_ERR_UNKNOWN_HANDLE 0x02
- #define HCI_ERR_HARDWARE_FAILURE 0x03
- #define HCI_ERR_PAGE_TIMEOUT 0x04
- #define HCI_ERR_AUTH_FAILURE 0x05
- #define HCI_ERR_KEY_MISSING 0x06
- #define HCI_ERR_MEMORY_EXCEEDED 0x07
- #define HCI_ERR_CONN_TIMEOUT 0x08
- #define HCI_ERR_CONN_LIMIT 0x09
- #define HCI_ERR_SYNCH_CONN_LIMIT 0x0A
- #define HCI_ERR_ACL_CONN_EXISTS 0x0B
- #define HCI_ERR_CMD_DISALLOWED 0x0C
- #define HCI_ERR_REJ_RESOURCES 0x0D
- #define HCI_ERR_REJ_SECURITY 0x0E
- #define HCI_ERR_REJ_BD_ADDR 0x0F
- #define HCI_ERR_ACCEPT_TIMEOUT 0x10
- #define HCI_ERR_UNSUP_FEAT 0x11
- #define HCI_ERR_INVALID_PARAM 0x12
- #define HCI_ERR_REMOTE_TERMINATED 0x13
- #define HCI_ERR_REMOTE_RESOURCES 0x14
- #define HCI_ERR_REMOTE_POWER_OFF 0x15
- #define HCI_ERR_LOCAL_TERMINATED 0x16
- #define HCI_ERR_REPEATED_ATTEMPTS 0x17
- #define HCI_ERR_PAIRING_NOT_ALLOWED 0x18
- #define HCI_ERR_UNKNOWN_LMP_PDU 0x19
- #define HCI_ERR_UNSUP_REMOTE_FEAT 0x1A
- #define HCI_ERR_SCO_OFFSET 0x1B
- #define HCI_ERR_SCO_INTERVAL 0x1C
- #define HCI_ERR_SCO_MODE 0x1D
- #define HCI_ERR_LMP_PARAM 0x1E
- #define HCI_ERR_UNSPECIFIED 0x1F
- #define HCI_ERR_UNSUP_LMP_PARAM 0x20
- #define HCI_ERR_ROLE_CHANGE 0x21
- #define HCI_ERR_LL_RESP_TIMEOUT 0x22
- #define HCI_ERR_LMP_COLLISION 0x23
- #define HCI_ERR_LMP_PDU 0x24
- #define HCI_ERR_ENCRYPT_MODE 0x25
- #define HCI_ERR_LINK_KEY 0x26
- #define HCI_ERR_UNSUP_QOS 0x27
- #define HCI_ERR_INSTANT_PASSED 0x28
- #define HCI_ERR_UNSUP_UNIT_KEY 0x29
- #define HCI_ERR_TRANSACT_COLLISION 0x2A
- #define HCI_ERR_CHANNEL_CLASS 0x2E
- #define HCI_ERR_MEMORY 0x2F
- #define HCI_ERR_PARAMETER_RANGE 0x30
- #define HCI_ERR_ROLE_SWITCH_PEND 0x32
- #define HCI_ERR_RESERVED_SLOT 0x34
- #define HCI_ERR_ROLE_SWITCH 0x35
- #define HCI_ERR_INQ_TOO_LARGE 0x36
- #define HCI_ERR_UNSUP_SSP 0x37
- #define HCI_ERR_HOST_BUSY_PAIRING 0x38
- #define HCI_ERR_NO_CHANNEL 0x39

- #define HCI_ERR_CONTROLLER_BUSY 0x3A
- #define HCI_ERR_CONN_INTERVAL 0x3B
- #define HCI_ERR_ADV_TIMEOUT 0x3C
- #define HCI_ERR_MIC_FAILURE 0x3D
- #define HCI_ERR_CONN_FAIL 0x3E
- #define HCI_ERR_MAC_CONN_FAIL 0x3F
- #define HCI_ERR_COARSE_CLK_ADJ_REJ 0x40
- #define HCI_ERR_TYPE0_SUBMAP_NOT_DEF 0x41
- #define HCI_ERR_UNKNOWN_ADV_ID 0x42
- #define HCI_ERR_LIMIT_REACHED 0x43
- #define HCI_ERR_OP_CANCELLED_BY_HOST 0x44
- #define HCI_ERR_PKT_TOO_LONG 0x45

Command groups

- #define HCI_OGF_NOP 0x00
- #define HCI_OGF_LINK_CONTROL 0x01
- #define HCI_OGF_LINK_POLICY 0x02
- #define HCI_OGF_CONTROLLER 0x03
- #define HCI_OGF_INFORMATIONAL 0x04
- #define HCI_OGF_STATUS 0x05
- #define HCI_OGF_TESTING 0x06
- #define HCI_OGF_LE_CONTROLLER 0x08
- #define HCI_OGF_VENDOR_SPEC 0x3F

NOP command

- #define **HCI_OCF_NOP** 0x00

Link control commands

- #define **HCI_OCF_DISCONNECT** 0x06
- #define **HCI_OCF_READ_REMOTE_VER_INFO** 0x1D

Controller and baseband commands

- #define **HCI_OCF_SET_EVENT_MASK** 0x01
- #define **HCI_OCF_RESET** 0x03
- #define **HCI_OCF_READ_TX_PWR_LVL** 0x2D
- #define **HCI_OCF_SET_CONTROLLER_TO_HOST_FC** 0x31
- #define **HCI_OCF_HOST_BUFFER_SIZE** 0x33
- #define **HCI_OCF_HOST_NUM_CMPL_PKTS** 0x35
- #define **HCI_OCF_SET_EVENT_MASK_PAGE2** 0x63
- #define **HCI_OCF_READ_AUTH_PAYLOAD_TO** 0x7B
- #define **HCI_OCF_WRITE_AUTH_PAYLOAD_TO** 0x7C
- #define **HCI_OCF_CONFIG_DATA_PATH** 0x83

Informational commands

- #define **HCI_OCF_READ_LOCAL_VER_INFO** 0x01
- #define **HCI_OCF_READ_LOCAL_SUP_CMDS** 0x02
- #define **HCI_OCF_READ_LOCAL_SUP_FEAT** 0x03
- #define **HCI_OCF_READ_BUF_SIZE** 0x05
- #define **HCI_OCF_READ_BD_ADDR** 0x09
- #define **HCI_OCF_READ_LOCAL_SUP_CODECS** 0x0D
- #define **HCI_OCF_READ_LOCAL_SUP_CODEC_CAP** 0x0E
- #define **HCI_OCF_READ_LOCAL_SUP_CONTROLLER_DLY** 0x0F

Status commands

- #define **HCI_OCF_READ_RSSI** 0x05

LE controller commands

- #define **HCI_OCF_LE_SET_EVENT_MASK** 0x01
- #define **HCI_OCF_LE_READ_BUF_SIZE** 0x02
- #define **HCI_OCF_LE_READ_LOCAL_SUP_FEAT** 0x03
- #define **HCI_OCF_LE_SET_RAND_ADDR** 0x05
- #define **HCI_OCF_LE_SET_ADV_PARAM** 0x06
- #define **HCI_OCF_LE_READ_ADV_TX_POWER** 0x07
- #define **HCI_OCF_LE_SET_ADV_DATA** 0x08
- #define **HCI_OCF_LE_SET_SCAN_RESP_DATA** 0x09
- #define **HCI_OCF_LE_SET_ADV_ENABLE** 0x0A
- #define **HCI_OCF_LE_SET_SCAN_PARAM** 0x0B
- #define **HCI_OCF_LE_SET_SCAN_ENABLE** 0x0C
- #define **HCI_OCF_LE_CREATE_CONN** 0x0D
- #define **HCI_OCF_LE_CREATE_CONN_CANCEL** 0x0E
- #define **HCI_OCF_LE_READ_WHITE_LIST_SIZE** 0x0F
- #define **HCI_OCF_LE_CLEAR_WHITE_LIST** 0x10
- #define **HCI_OCF_LE_ADD_DEV_WHITE_LIST** 0x11
- #define **HCI_OCF_LE_REMOVE_DEV_WHITE_LIST** 0x12
- #define **HCI_OCF_LE_CONN_UPDATE** 0x13
- #define **HCI_OCF_LE_SET_HOST_CHAN_CLASS** 0x14
- #define **HCI_OCF_LE_READ_CHAN_MAP** 0x15
- #define **HCI_OCF_LE_READ_REMOTE_FEAT** 0x16
- #define **HCI_OCF_LE_ENCRYPT** 0x17
- #define **HCI_OCF_LE_RAND** 0x18
- #define **HCI_OCF_LE_START_ENCRYPTION** 0x19
- #define **HCI_OCF_LE_LTK_REQ_REPL** 0x1A
- #define **HCI_OCF_LE_LTK_REQ_NEG_REPL** 0x1B
- #define **HCI_OCF_LE_READ_SUP_STATES** 0x1C
- #define **HCI_OCF_LE_RECEIVER_TEST** 0x1D
- #define **HCI_OCF_LE_TRANSMITTER_TEST** 0x1E
- #define **HCI_OCF_LE_TEST_END** 0x1F
- #define **HCI_OCF_LE_REM_CONN_PARAM_REP** 0x20
- #define **HCI_OCF_LE_REM_CONN_PARAM_NEG_REP** 0x21
- #define **HCI_OCF_LE_SET_DATA_LEN** 0x22
- #define **HCI_OCF_LE_READ_DEF_DATA_LEN** 0x23
- #define **HCI_OCF_LE_WRITE_DEF_DATA_LEN** 0x24
- #define **HCI_OCF_LE_READ_LOCAL_P256_PUB_KEY** 0x25

- #define **HCI_OCF_LE_GENERATE_DHKEY** 0x26
- #define **HCI_OCF_LE_ADD_DEV_RES_LIST** 0x27
- #define **HCI_OCF_LE_REMOVE_DEV_RES_LIST** 0x28
- #define **HCI_OCF_LE_CLEAR_RES_LIST** 0x29
- #define **HCI_OCF_LE_READ_RES_LIST_SIZE** 0x2A
- #define **HCI_OCF_LE_READ_PEER_RES_ADDR** 0x2B
- #define **HCI_OCF_LE_READ_LOCAL_RES_ADDR** 0x2C
- #define **HCI_OCF_LE_SET_ADDR_RES_ENABLE** 0x2D
- #define **HCI_OCF_LE_SET_RES_PRIV_ADDR_TO** 0x2E
- #define **HCI_OCF_LE_READ_MAX_DATA_LEN** 0x2F
- #define **HCI_OCF_LE_READ_PHY** 0x30
- #define **HCI_OCF_LE_SET_DEF_PHY** 0x31
- #define **HCI_OCF_LE_SET_PHY** 0x32
- #define **HCI_OCF_LE_ENHANCED_RECEIVER_TEST** 0x33
- #define **HCI_OCF_LE_ENHANCED_TRANSMITTER_TEST** 0x34
- #define **HCI_OCF_LE_SET_ADV_SET_RAND_ADDR** 0x35
- #define **HCI_OCF_LE_SET_EXT_ADV_PARAM** 0x36
- #define **HCI_OCF_LE_SET_EXT_ADV_DATA** 0x37
- #define **HCI_OCF_LE_SET_EXT_SCAN_RESP_DATA** 0x38
- #define **HCI_OCF_LE_SET_EXT_ADV_ENABLE** 0x39
- #define **HCI_OCF_LE_READ_MAX_ADV_DATA_LEN** 0x3A
- #define **HCI_OCF_LE_READ_NUM_SUP_ADV_SETS** 0x3B
- #define **HCI_OCF_LE_REMOVE_ADV_SET** 0x3C
- #define **HCI_OCF_LE_CLEAR_ADV_SETS** 0x3D
- #define **HCI_OCF_LE_SET_PER_ADV_PARAM** 0x3E
- #define **HCI_OCF_LE_SET_PER_ADV_DATA** 0x3F
- #define **HCI_OCF_LE_SET_PER_ADV_ENABLE** 0x40
- #define **HCI_OCF_LE_SET_EXT_SCAN_PARAM** 0x41
- #define **HCI_OCF_LE_SET_EXT_SCAN_ENABLE** 0x42
- #define **HCI_OCF_LE_EXT_CREATE_CONN** 0x43
- #define **HCI_OCF_LE_PER_ADV_CREATE_SYNC** 0x44
- #define **HCI_OCF_LE_PER_ADV_CREATE_SYNC_CANCEL** 0x45
- #define **HCI_OCF_LE_PER_ADV_TERM_SYNC** 0x46
- #define **HCI_OCF_LE_ADD_DEV_PER_ADV_LIST** 0x47
- #define **HCI_OCF_LE_REMOVE_DEV_PER_ADV_LIST** 0x48
- #define **HCI_OCF_LE_CLEAR_PER_ADV_LIST** 0x49
- #define **HCI_OCF_LE_READ_PER_ADV_LIST_SIZE** 0x4A
- #define **HCI_OCF_LE_READ_TX_POWER** 0x4B
- #define **HCI_OCF_LE_READ_RF_PATH_COMP** 0x4C
- #define **HCI_OCF_LE_WRITE_RF_PATH_COMP** 0x4D
- #define **HCI_OCF_LE_SET_PRIVACY_MODE** 0x4E
- #define **HCI_OCF_LE_RECEIVER_TEST_V3** 0x4F
- #define **HCI_OCF_LE_TRANSMITTER_TEST_V3** 0x50
- #define **HCI_OCF_LE_SET_CONNLESS_CTE_TX_PARAMS** 0x51
- #define **HCI_OCF_LE_SET_CONNLESS_CTE_TX_ENABLE** 0x52
- #define **HCI_OCF_LE_SET_CONNLESS_IQ_SAMP_ENABLE** 0x53
- #define **HCI_OCF_LE_SET_CONN_CTE_RX_PARAMS** 0x54
- #define **HCI_OCF_LE_SET_CONN_CTE_TX_PARAMS** 0x55
- #define **HCI_OCF_LE_CONN_CTE_REQ_ENABLE** 0x56
- #define **HCI_OCF_LE_CONN_CTE_RSP_ENABLE** 0x57
- #define **HCI_OCF_LE_READ_ANTENNA_INFO** 0x58
- #define **HCI_OCF_LE_SET_PER_ADV_RCV_ENABLE** 0x59
- #define **HCI_OCF_LE_PER_ADV_SYNC_TRANSFER** 0x5A
- #define **HCI_OCF_LE_PER_ADV_SET_INFO_TRANSFER** 0x5B
- #define **HCI_OCF_LE_SET_PAST_PARAM** 0x5C

- #define **HCI_OCF_LE_SET_DEFAULT_PAST_PARAM** 0x5D
- #define **HCI_OCF_LE_GENERATE_DHKEY_V2** 0x5E
- #define **HCI_OCF_LE_MODIFY_SLEEP_CLK_ACC** 0x5F
- #define **HCI_OCF_LE_READ_BUF_SIZE_V2** 0x60
- #define **HCI_OCF_LE_READ_ISO_TX_SYNC** 0x61
- #define **HCI_OCF_LE_SET_CIG_PARAMS** 0x62
- #define **HCI_OCF_LE_SET_CIG_PARAMS_TEST** 0x63
- #define **HCI_OCF_LE_CREATE_CIS** 0x64
- #define **HCI_OCF_LE_REMOVE_CIG** 0x65
- #define **HCI_OCF_LE_ACCEPT_CIS_REQ** 0x66
- #define **HCI_OCF_LE_REJECT_CIS_REQ** 0x67
- #define **HCI_OCF_LE_CREATE_BIG** 0x68
- #define **HCI_OCF_LE_CREATE_BIG_TEST** 0x69
- #define **HCI_OCF_LE_TERMINATE_BIG** 0x6A
- #define **HCI_OCF_LE_BIG_CREATE_SYNC** 0x6B
- #define **HCI_OCF_LE_BIG_TERMINATE_SYNC** 0x6C
- #define **HCI_OCF_LE_REQUEST_PEER_SCA** 0x6D
- #define **HCI_OCF_LE_SETUP_ISO_DATA_PATH** 0x6E
- #define **HCI_OCF_LE_REMOVE_ISO_DATA_PATH** 0x6F
- #define **HCI_OCF_LE_ISO_TX_TEST** 0x70
- #define **HCI_OCF_LE_ISO_RX_TEST** 0x71
- #define **HCI_OCF_LE_ISO_READ_TEST_COUNTERS** 0x72
- #define **HCI_OCF_LE_ISO_TEST_END** 0x73
- #define **HCI_OCF_LE_SET_HOST_FEATURE** 0x74
- #define **HCI_OCF_LE_READ_ISO_LINK_QUAL** 0x75
- #define **HCI_OCF_LE_READ_ENHANCED_TX_POWER** 0x76
- #define **HCI_OCF_LE_READ_REMOTE_TX_POWER** 0x77
- #define **HCI_OCF_LE_SET_PATH_LOSS_REPORTING_PARAMS** 0x78
- #define **HCI_OCF_LE_SET_PATH_LOSS_REPORTING_ENABLE** 0x79
- #define **HCI_OCF_LE_SET_TX_POWER_REPORT_ENABLE** 0x7A

Opcode manipulation macros

- #define **HCI_OPCODE**(ogf, ocf) (((ogf) << 10) + (ocf))
- #define **HCI_OGF**(opcode) ((opcode) >> 10)
- #define **HCI_OCF**(opcode) ((opcode) & 0x03FF)

Command opcodes

- #define **HCI_OPCODE_NOP** HCI_OPCODE(HCI_OGF_NOP, HCI_OCF_NOP)
- #define **HCI_OPCODE_DISCONNECT** HCI_OPCODE(HCI_OGF_LINK_CONTROL, HCI_OCF_DISCONNECT)
- #define **HCI_OPCODE_READ_REMOTE_VER_INFO** HCI_OPCODE(HCI_OGF_LINK_CONTROL, HCI_OCF_READ_REMOTE_VER_INFO)
- #define **HCI_OPCODE_SET_EVENT_MASK** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_SET_EVENT_MASK)
- #define **HCI_OPCODE_RESET** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_RESET)
- #define **HCI_OPCODE_HOST_BUFFER_SIZE** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_HOST_BUFFER_SIZE)
- #define **HCI_OPCODE_READ_TX_PWR_LVL** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_READ_TX_PWR_LVL)
- #define **HCI_OPCODE_SET_EVENT_MASK_PAGE2** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_SET_EVENT_MASK_PAGE2)

- #define **HCI_OPCODE_READ_AUTH_PAYLOAD_TO** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_READ_AUTH_PAYLOAD_TO)
- #define **HCI_OPCODE_WRITE_AUTH_PAYLOAD_TO** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_WRITE_AUTH_PAYLOAD_TO)
- #define **HCI_OPCODE_CONFIG_DATA_PATH** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_CONFIG_DATA_PATH)
- #define **HCI_OPCODE_READ_LOCAL_VER_INFO** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_LOCAL_VER_INFO)
- #define **HCI_OPCODE_READ_LOCAL_SUP_CMDS** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_LOCAL_SUP_CMDS)
- #define **HCI_OPCODE_READ_LOCAL_SUP_FEAT** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_LOCAL_SUP_FEAT)
- #define **HCI_OPCODE_READ_BUF_SIZE** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_BUF_SIZE)
- #define **HCI_OPCODE_READ_BD_ADDR** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_BD_ADDR)
- #define **HCI_OPCODE_READ_LOCAL_SUP_CODECS** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_LOCAL_SUP_CODECS)
- #define **HCI_OPCODE_READ_LOCAL_SUP_CODEC_CAP** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_LOCAL_SUP_CODEC_CAP)
- #define **HCI_OPCODE_READ_LOCAL_SUP_CONTROLLER_DLY** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_LOCAL_SUP_CONTROLLER_DLY)
- #define **HCI_OPCODE_READ_RSSI** HCI_OPCODE(HCI_OGF_STATUS, HCI_OCF_READ_RSSI)
- #define **HCI_OPCODE_LE_SET_EVENT_MASK** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EVENT_MASK)
- #define **HCI_OPCODE_LE_READ_BUF_SIZE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_BUF_SIZE)
- #define **HCI_OPCODE_LE_READ_LOCAL_SUP_FEAT** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_LOCAL_SUP_FEAT)
- #define **HCI_OPCODE_LE_SET_RAND_ADDR** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_RAND_ADDR)
- #define **HCI_OPCODE_LE_SET_ADV_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_ADV_PARAM)
- #define **HCI_OPCODE_LE_READ_ADV_TX_POWER** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_ADV_TX_POWER)
- #define **HCI_OPCODE_LE_SET_ADV_DATA** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_ADV_DATA)
- #define **HCI_OPCODE_LE_SET_SCAN_RESP_DATA** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_SCAN_RESP_DATA)
- #define **HCI_OPCODE_LE_SET_ADV_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_ADV_ENABLE)
- #define **HCI_OPCODE_LE_SET_SCAN_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_SCAN_PARAM)
- #define **HCI_OPCODE_LE_SET_SCAN_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_SCAN_ENABLE)
- #define **HCI_OPCODE_LE_CREATE_CONN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CREATE_CONN)
- #define **HCI_OPCODE_LE_CREATE_CONN_CANCEL** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CREATE_CONN_CANCEL)
- #define **HCI_OPCODE_LE_READ_WHITE_LIST_SIZE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_WHITE_LIST_SIZE)
- #define **HCI_OPCODE_LE_CLEAR_WHITE_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CLEAR_WHITE_LIST)
- #define **HCI_OPCODE_LE_ADD_DEV_WHITE_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ADD_DEV_WHITE_LIST)
- #define **HCI_OPCODE_LE_REMOVE_DEV_WHITE_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REMOVE_DEV_WHITE_LIST)

- #define **HCI_OPCODE_LE_CONN_UPDATE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CONN_UPDATE)
- #define **HCI_OPCODE_LE_SET_HOST_CHAN_CLASS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_HOST_CHAN_CLASS)
- #define **HCI_OPCODE_LE_READ_CHAN_MAP** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_CHAN_MAP)
- #define **HCI_OPCODE_LE_READ_REMOTE_FEAT** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_REMOTE_FEAT)
- #define **HCI_OPCODE_LE_ENCRYPT** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ENCRYPT)
- #define **HCI_OPCODE_LE_RAND** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_RAND)
- #define **HCI_OPCODE_LE_START_ENCRYPTION** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_START_ENCRYPTION)
- #define **HCI_OPCODE_LE_LTK_REQ_REPL** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_LTK_REQ_REPL)
- #define **HCI_OPCODE_LE_LTK_REQ_NEG_REPL** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_LTK_REQ_NEG_REPL)
- #define **HCI_OPCODE_LE_READ_SUP_STATES** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_SUP_STATES)
- #define **HCI_OPCODE_LE_RECEIVER_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_RECEIVER_TEST)
- #define **HCI_OPCODE_LE_TRANSMITTER_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_TRANSMITTER_TEST)
- #define **HCI_OPCODE_LE_TEST_END** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_TEST_END)
- #define **HCI_OPCODE_LE_REM_CONN_PARAM_REP** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REM_CONN_PARAM_REP)
- #define **HCI_OPCODE_LE_REM_CONN_PARAM_NEG_REP** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REM_CONN_PARAM_NEG_REP)
- #define **HCI_OPCODE_LE_SET_DATA_LEN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_DATA_LEN)
- #define **HCI_OPCODE_LE_READ_DEF_DATA_LEN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_DEF_DATA_LEN)
- #define **HCI_OPCODE_LE_WRITE_DEF_DATA_LEN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_WRITE_DEF_DATA_LEN)
- #define **HCI_OPCODE_LE_READ_LOCAL_P256_PUB_KEY** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_LOCAL_P256_PUB_KEY)
- #define **HCI_OPCODE_LE_GENERATE_DHKEY** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_GENERATE_DHKEY)
- #define **HCI_OPCODE_LE_ADD_DEV_RES_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ADD_DEV_RES_LIST)
- #define **HCI_OPCODE_LE_REMOVE_DEV_RES_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REMOVE_DEV_RES_LIST)
- #define **HCI_OPCODE_LE_CLEAR_RES_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CLEAR_RES_LIST)
- #define **HCI_OPCODE_LE_READ_RES_LIST_SIZE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_RES_LIST_SIZE)
- #define **HCI_OPCODE_LE_READ_PEER_RES_ADDR** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_PEER_RES_ADDR)
- #define **HCI_OPCODE_LE_READ_LOCAL_RES_ADDR** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_LOCAL_RES_ADDR)
- #define **HCI_OPCODE_LE_SET_ADDR_RES_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_ADDR_RES_ENABLE)
- #define **HCI_OPCODE_LE_SET_RES_PRIV_ADDR_TO** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_RES_PRIV_ADDR_TO)
- #define **HCI_OPCODE_LE_READ_MAX_DATA_LEN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_MAX_DATA_LEN)

- #define **HCI_OPCODE_LE_READ_PHY** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_PHY)
- #define **HCI_OPCODE_LE_SET_DEF_PHY** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_DEF_PHY)
- #define **HCI_OPCODE_LE_SET_PHY** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PHY)
- #define **HCI_OPCODE_LE_ENHANCED_RECEIVER_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ENHANCED_RECEIVER_TEST)
- #define **HCI_OPCODE_LE_ENHANCED_TRANSMITTER_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ENHANCED_TRANSMITTER_TEST)
- #define **HCI_OPCODE_LE_SET_ADV_SET_RAND_ADDR** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_ADV_SET_RAND_ADDR)
- #define **HCI_OPCODE_LE_SET_EXT_ADV_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EXT_ADV_PARAM)
- #define **HCI_OPCODE_LE_SET_EXT_ADV_DATA** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EXT_ADV_DATA)
- #define **HCI_OPCODE_LE_SET_EXT_SCAN_RESP_DATA** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EXT_SCAN_RESP_DATA)
- #define **HCI_OPCODE_LE_SET_EXT_ADV_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EXT_ADV_ENABLE)
- #define **HCI_OPCODE_LE_READ_MAX_ADV_DATA_LEN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_MAX_ADV_DATA_LEN)
- #define **HCI_OPCODE_LE_READ_NUM_SUP_ADV_SETS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_NUM_SUP_ADV_SETS)
- #define **HCI_OPCODE_LE_REMOVE_ADV_SET** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REMOVE_ADV_SET)
- #define **HCI_OPCODE_LE_CLEAR_ADV_SETS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CLEAR_ADV_SETS)
- #define **HCI_OPCODE_LE_SET_PER_ADV_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PER_ADV_PARAM)
- #define **HCI_OPCODE_LE_SET_PER_ADV_DATA** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PER_ADV_DATA)
- #define **HCI_OPCODE_LE_SET_PER_ADV_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PER_ADV_ENABLE)
- #define **HCI_OPCODE_LE_SET_EXT_SCAN_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EXT_SCAN_PARAM)
- #define **HCI_OPCODE_LE_SET_EXT_SCAN_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EXT_SCAN_ENABLE)
- #define **HCI_OPCODE_LE_EXT_CREATE_CONN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_EXT_CREATE_CONN)
- #define **HCI_OPCODE_LE_PER_ADV_CREATE_SYNC** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_PER_ADV_CREATE_SYNC)
- #define **HCI_OPCODE_LE_PER_ADV_CREATE_SYNC_CANCEL** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_PER_ADV_CREATE_SYNC_CANCEL)
- #define **HCI_OPCODE_LE_PER_ADV_TERMINATE_SYNC** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_PER_ADV_TERM_SYNC)
- #define **HCI_OPCODE_LE_ADD_DEV_PER_ADV_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ADD_DEV_PER_ADV_LIST)
- #define **HCI_OPCODE_LE_REMOVE_DEV_PER_ADV_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REMOVE_DEV_PER_ADV_LIST)
- #define **HCI_OPCODE_LE_CLEAR_PER_ADV_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CLEAR_PER_ADV_LIST)
- #define **HCI_OPCODE_LE_READ_PER_ADV_LIST_SIZE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_PER_ADV_LIST_SIZE)
- #define **HCI_OPCODE_LE_READ_TX_POWER** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_TX_POWER)

- #define **HCI_OPCODE_LE_WRITE_RF_PATH_COMP** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_WRITE_RF_PATH_COMP)
- #define **HCI_OPCODE_LE_READ_RF_PATH_COMP** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_RF_PATH_COMP)
- #define **HCI_OPCODE_LE_SET_PRIVACY_MODE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PRIVACY_MODE)
- #define **HCI_OPCODE_LE_RECEIVER_TEST_V3** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_RECEIVER_TEST_V3)
- #define **HCI_OPCODE_LE_TRANSMITTER_TEST_V3** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_TRANSMITTER_TEST_V3)
- #define **HCI_OPCODE_LE_SET_CONNLESS_CTE_TX_PARAMS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CONNLESS_CTE_TX_PARAMS)
- #define **HCI_OPCODE_LE_SET_CONNLESS_CTE_TX_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CONNLESS_CTE_TX_ENABLE)
- #define **HCI_OPCODE_LE_SET_CONNLESS_IQ_SAMP_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CONNLESS_IQ_SAMP_ENABLE)
- #define **HCI_OPCODE_LE_SET_CONN_CTE_RX_PARAMS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CONN_CTE_RX_PARAMS)
- #define **HCI_OPCODE_LE_SET_CONN_CTE_TX_PARAMS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CONN_CTE_TX_PARAMS)
- #define **HCI_OPCODE_LE_CONN_CTE_REQ_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CONN_CTE_REQ_ENABLE)
- #define **HCI_OPCODE_LE_CONN_CTE_RSP_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CONN_CTE_RSP_ENABLE)
- #define **HCI_OPCODE_LE_READ_ANTENNA_INFO** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_ANTENNA_INFO)
- #define **HCI_OPCODE_LE_SET_PER_ADV_RCV_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PER_ADV_RCV_ENABLE)
- #define **HCI_OPCODE_LE_PER_ADV_SYNC_TRANSFER** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_PER_ADV_SYNC_TRANSFER)
- #define **HCI_OPCODE_LE_PER_ADV_SET_INFO_TRANSFER** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_PER_ADV_SET_INFO_TRANSFER)
- #define **HCI_OPCODE_LE_SET_PAST_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PAST_PARAM)
- #define **HCI_OPCODE_LE_SET_DEFAULT_PAST_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_DEFAULT_PAST_PARAM)
- #define **HCI_OPCODE_LE_GENERATE_DHKEY_V2** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_GENERATE_DHKEY_V2)
- #define **HCI_OPCODE_LE_MODIFY_SLEEP_CLK_ACC** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_MODIFY_SLEEP_CLK_ACC)
- #define **HCI_OPCODE_LE_READ_BUF_SIZE_V2** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_BUF_SIZE_V2)
- #define **HCI_OPCODE_LE_READ_ISO_TX_SYNC** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_ISO_TX_SYNC)
- #define **HCI_OPCODE_LE_SET_CIG_PARAMS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CIG_PARAMS)
- #define **HCI_OPCODE_LE_SET_CIG_PARAMS_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CIG_PARAMS_TEST)
- #define **HCI_OPCODE_LE_CREATE_CIS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CREATE_CIS)
- #define **HCI_OPCODE_LE_REMOVE_CIG** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REMOVE_CIG)
- #define **HCI_OPCODE_LE_ACCEPT_CIS_REQ** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ACCEPT_CIS_REQ)
- #define **HCI_OPCODE_LE_REJECT_CIS_REQ** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REJECT_CIS_REQ)

- #define **HCI_OPCODE_LE_CREATE_BIG** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CREATE_BIG)
- #define **HCI_OPCODE_LE_CREATE_BIG_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CREATE_BIG_TEST)
- #define **HCI_OPCODE_LE_TERMINATE_BIG** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_TERMINATE_BIG)
- #define **HCI_OPCODE_LE_BIG_CREATE_SYNC** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_BIG_CREATE_SYNC)
- #define **HCI_OPCODE_LE_BIG_TERMINATE_SYNC** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_BIG_TERMINATE_SYNC)
- #define **HCI_OPCODE_LE_REQUEST_PEER_SCA** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REQUEST_PEER_SCA)
- #define **HCI_OPCODE_LE_SETUP_ISO_DATA_PATH** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SETUP_ISO_DATA_PATH)
- #define **HCI_OPCODE_LE_REMOVE_ISO_DATA_PATH** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REMOVE_ISO_DATA_PATH)
- #define **HCI_OPCODE_LE_ISO_TX_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ISO_TX_TEST)
- #define **HCI_OPCODE_LE_ISO_RX_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ISO_RX_TEST)
- #define **HCI_OPCODE_LE_ISO_READ_TEST_COUNTERS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ISO_READ_TEST_COUNTERS)
- #define **HCI_OPCODE_LE_ISO_TEST_END** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ISO_TEST_END)
- #define **HCI_OPCODE_LE_SET_HOST_FEATURE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_HOST_FEATURE)
- #define **HCI_OPCODE_LE_READ_ISO_LINK_QUAL** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_ISO_LINK_QUAL)
- #define **HCI_OPCODE_LE_READ_ENHANCED_TX_POWER** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_ENHANCED_TX_POWER)
- #define **HCI_OPCODE_LE_READ_REMOTE_TX_POWER** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_REMOTE_TX_POWER)
- #define **HCI_OPCODE_LE_SET_PATH_LOSS_REPORTING_PARAMS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PATH_LOSS_REPORTING_PARAMS)
- #define **HCI_OPCODE_LE_SET_PATH_LOSS_REPORTING_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PATH_LOSS_REPORTING_ENABLE)
- #define **HCI_OPCODE_LE_SET_TX_POWER_REPORT_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_TX_POWER_REPORT_ENABLE)

Packetcraft Vendor Specific

- #define **HCI_OPCODE_LE_VS_ENABLE_READ_FEAT_ON_CONN** ((uint16_t)(0xfff3))

Command parameter lengths

- #define **HCI_LEN_NOP** 0
- #define **HCI_LEN_DISCONNECT** 3
- #define **HCI_LEN_READ_REMOTE_VER_INFO** 2
- #define **HCI_LEN_SET_EVENT_MASK** 8
- #define **HCI_LEN_SET_EVENT_MASK_PAGE2** 8
- #define **HCI_LEN_RESET** 0
- #define **HCI_LEN_READ_TX_PWR_LVL** 3
- #define **HCI_LEN_SET_CONTROLLER_TO_HOST_FC** 1
- #define **HCI_LEN_HOST_BUFFER_SIZE** 7

- #define **HCI_LEN_HOST_NUM_CMPL_PKTS** 1
- #define **HCI_LEN_CONFIG_DATA_PATH**(cLen) (3 + (cLen))
- #define **HCI_LEN_READ_LOCAL_VER_INFO** 0
- #define **HCI_LEN_READ_LOCAL_SUP_CMDS** 0
- #define **HCI_LEN_READ_LOCAL_SUP_FEAT** 0
- #define **HCI_LEN_READ_BUF_SIZE** 0
- #define **HCI_LEN_READ_BD_ADDR** 0
- #define **HCI_LEN_READ_LOCAL_SUP_CODECS** 0
- #define **HCI_LEN_READ_LOCAL_SUP_CODEC_CAP** 7
- #define **HCI_LEN_READ_LOCAL_SUP_CONTROLLER_DLY**(ccLen) (8 + (ccLen))
- #define **HCI_LEN_READ_RSSI** 2
- #define **HCI_LEN_READ_AUTH_PAYLOAD_TO** 2
- #define **HCI_LEN_WRITE_AUTH_PAYLOAD_TO** 4
- #define **HCI_LEN_LE_SET_EVENT_MASK** 8
- #define **HCI_LEN_LE_READ_BUF_SIZE** 0
- #define **HCI_LEN_LE_READ_LOCAL_SUP_FEAT** 0
- #define **HCI_LEN_LE_SET_RAND_ADDR** 6
- #define **HCI_LEN_LE_SET_ADV_PARAM** 15
- #define **HCI_LEN_LE_READ_ADV_TX_POWER** 0
- #define **HCI_LEN_LE_SET_ADV_DATA** 32
- #define **HCI_LEN_LE_SET_SCAN_RESP_DATA** 32
- #define **HCI_LEN_LE_SET_ADV_ENABLE** 1
- #define **HCI_LEN_LE_SET_SCAN_PARAM** 7
- #define **HCI_LEN_LE_SET_SCAN_ENABLE** 2
- #define **HCI_LEN_LE_CREATE_CONN** 25
- #define **HCI_LEN_LE_CREATE_CONN_CANCEL** 0
- #define **HCI_LEN_LE_READ_WHITE_LIST_SIZE** 0
- #define **HCI_LEN_LE_CLEAR_WHITE_LIST** 0
- #define **HCI_LEN_LE_ADD_DEV_WHITE_LIST** 7
- #define **HCI_LEN_LE_REMOVE_DEV_WHITE_LIST** 7
- #define **HCI_LEN_LE_CONN_UPDATE** 14
- #define **HCI_LEN_LE_SET_HOST_CHAN_CLASS** 5
- #define **HCI_LEN_LE_READ_CHAN_MAP** 2
- #define **HCI_LEN_LE_READ_REMOTE_FEAT** 2
- #define **HCI_LEN_LE_ENCRYPT** 32
- #define **HCI_LEN_LE_RAND** 0
- #define **HCI_LEN_LE_START_ENCRYPTION** 28
- #define **HCI_LEN_LE_LTK_REQ_REPL** 18
- #define **HCI_LEN_LE_LTK_REQ_NEG_REPL** 2
- #define **HCI_LEN_LE_READ_SUP_STATES** 0
- #define **HCI_LEN_LE_RECEIVER_TEST** 1
- #define **HCI_LEN_LE_TRANSMITTER_TEST** 3
- #define **HCI_LEN_LE_TEST_END** 0
- #define **HCI_LEN_LE_REM_CONN_PARAM_REP** 14
- #define **HCI_LEN_LE_REM_CONN_PARAM_NEG_REP** 3
- #define **HCI_LEN_LE_SET_DATA_LEN** 6
- #define **HCI_LEN_LE_READ_DEF_DATA_LEN** 0
- #define **HCI_LEN_LE_WRITE_DEF_DATA_LEN** 4
- #define **HCI_LEN_LE_READ_LOCAL_P256_PUB_KEY** 0
- #define **HCI_LEN_LE_GENERATE_DHKEY** 64
- #define **HCI_LEN_LE_ADD_DEV_RES_LIST** 39
- #define **HCI_LEN_LE_REMOVE_DEV_RES_LIST** 7
- #define **HCI_LEN_LE_CLEAR_RES_LIST** 0
- #define **HCI_LEN_LE_READ_RES_LIST_SIZE** 0
- #define **HCI_LEN_LE_READ_PEER_RES_ADDR** 7

- #define **HCI_LEN_LE_READ_LOCAL_RES_ADDR** 7
- #define **HCI_LEN_LE_SET_ADDR_RES_ENABLE** 1
- #define **HCI_LEN_LE_SET_RES_PRIV_ADDR_TO** 2
- #define **HCI_LEN_LE_READ_MAX_DATA_LEN** 0
- #define **HCI_LEN_LE_READ_PHY** 2
- #define **HCI_LEN_LE_SET_DEF_PHY** 3
- #define **HCI_LEN_LE_SET_PHY** 7
- #define **HCI_LEN_LE_ENHANCED_RECEIVER_TEST** 3
- #define **HCI_LEN_LE_ENHANCED_TRANSMITTER_TEST** 4
- #define **HCI_LEN_LE_SET_ADV_SET_RAND_ADDR** 7
- #define **HCI_LEN_LE_SET_EXT_ADV_PARAM** 25
- #define **HCI_LEN_LE_SET_EXT_ADV_DATA**(len) (4 + (len))
- #define **HCI_LEN_LE_SET_EXT_SCAN_RESP_DATA**(len) (4 + (len))
- #define **HCI_LEN_LE_EXT_ADV_ENABLE**(numSets) (2 + (4 ∗ (numSets)))
- #define **HCI_LEN_LE_READ_MAX_ADV_DATA_LEN** 0
- #define **HCI_LEN_LE_READ_NUM_OF_SUP_ADV_SETS** 0
- #define **HCI_LEN_LE_REMOVE_ADV_SET** 1
- #define **HCI_LEN_LE_CLEAR_ADV_SETS** 0
- #define **HCI_LEN_LE_SET_PER_ADV_PARAM** 7
- #define **HCI_LEN_LE_SET_PER_ADV_DATA**(len) (3 + (len))
- #define **HCI_LEN_LE_SET_PER_ADV_ENABLE** 2
- #define **HCI_LEN_LE_SET_EXT_SCAN_PARAM**(numPhys) (3 + (5 ∗ (numPhys)))
- #define **HCI_LEN_LE_SET_EXT_SCAN_ENABLE** 6
- #define **HCI_LEN_LE_EXT_CREATE_CONN**(numPhys) (10 + (16 ∗ (numPhys)))
- #define **HCI_LEN_LE_PER_ADV_CREATE_SYNC** 14
- #define **HCI_LEN_LE_PER_ADV_CREATE_SYNC_CANCEL** 0
- #define **HCI_LEN_LE_PER_ADV_TERMINATE_SYNC** 2
- #define **HCI_LEN_LE_ADD_DEV_PER_ADV_LIST** 8
- #define **HCI_LEN_LE_REMOVE_DEV_PER_ADV_LIST** 8
- #define **HCI_LEN_LE_CLEAR_PER_ADV_LIST** 0
- #define **HCI_LEN_LE_READ_PER_ADV_LIST_SIZE** 0
- #define **HCI_LEN_LE_READ_TX_POWER** 0
- #define **HCI_LEN_LE_READ_RF_PATH_COMP** 0
- #define **HCI_LEN_LE_WRITE_RF_PATH_COMP** 4
- #define **HCI_LEN_LE_SET_PRIVACY_MODE** 8
- #define **HCI_LEN_LE_SET_CONN_CTE_RX_PARAMS**(spLen) (5 + (spLen))
- #define **HCI_LEN_LE_SET_CONN_CTE_TX_PARAMS**(spLen) (4 + (spLen))
- #define **HCI_LEN_LE_CONN_CTE_REQ_ENABLE** 7
- #define **HCI_LEN_LE_CONN_CTE_RSP_ENABLE** 3
- #define **HCI_LEN_LE_READ_ANTENNA_INFO** 0
- #define **HCI_LEN_LE_SET_PER_ADV_RCV_ENABLE** 3
- #define **HCI_LEN_LE_PER_ADV_SYNC_TRANSFER** 6
- #define **HCI_LEN_LE_PER_ADV_SET_INFO_TRANSFER** 5
- #define **HCI_LEN_LE_SET_PAST_PARAM** 8
- #define **HCI_LEN_LE_SET_DEFAULT_PAST_PARAM** 6
- #define **HCI_LEN_LE_GENERATE_DHKEY_V2** 65
- #define **HCI_LEN_LE_SET_CIG_PARAMS**(numCis) (15 + (9 ∗ (numCis)))
- #define **HCI_LEN_LE_CREATE_CIS**(numCis) (1 + (4 ∗ (numCis)))
- #define **HCI_LEN_LE_REMOVE_CIG** 1
- #define **HCI_LEN_LE_ACCEPT_CIS_REQ** 2
- #define **HCI_LEN_LE_REJECT_CIS_REQ** 3
- #define **HCI_LEN_LE_REQUEST_PEER_SCA** 2
- #define **HCI_LEN_LE_CREATE_BIS** (15 + HCI_BC_LEN)
- #define **HCI_LEN_LE_TERMINATE_BIG** 2
- #define **HCI_LEN_LE_BIG_CREATE_SYNC**(numBis) (8 + HCI_BC_LEN + (numBis))

- #define **HCI_LEN_LE_BIG_TERMINATE_SYNC** 1
- #define **HCI_LEN_LE_SETUP_ISO_DATA_PATH**(ccLen) (13 + (ccLen))
- #define **HCI_LEN_LE_REMOVE_ISO_DATA_PATH** 3
- #define **HCI_LEN_LE_ISO_TX_TEST** 3
- #define **HCI_LEN_LE_ISO_RX_TEST** 3
- #define **HCI_LEN_LE_ISO_READ_TEST_COUNTERS** 2
- #define **HCI_LEN_LE_ISO_TEST_END** 2
- #define **HCI_LEN_LE_SET_HOST_FEATURE** 2
- #define **HCI_LEN_LE_DISABLE_SLAVELATENCY** 3
- #define **HCI_LEN_LE_OVERRULE_REMOTE_MAX_RX_OCTETS_AND_TIME** 6
- #define **HCI_LEN_LE_SET_TRANSMIT_POWER** 1
- #define **HCI_LEN_LE_SET_EVENT_NOTIFICATION_BIT** 1
- #define **HCI_LEN_LE_RESET_EVENT_NOTIFICATION_BIT** 1

Events

- #define **HCI_DISCONNECT_CMPL_EVT** 0x05
- #define **HCI_ENC_CHANGE_EVT** 0x08
- #define **HCI_READ_REMOTE_VER_INFO_CMPL_EVT** 0x0C
- #define **HCI_CMD_CMPL_EVT** 0x0E
- #define **HCI_CMD_STATUS_EVT** 0x0F
- #define **HCI_HW_ERROR_EVT** 0x10
- #define **HCI_NUM_CMPL_PKTS_EVT** 0x13
- #define **HCI_DATA_BUF_OVERFLOW_EVT** 0x1A
- #define **HCI_ENC_KEY_REFRESH_CMPL_EVT** 0x30
- #define **HCI_LE_META_EVT** 0x3E
- #define **HCI_AUTH_PAYLOAD_TIMEOUT_EVT** 0x57
- #define **HCI_VENDOR_SPEC_EVT** 0xFF

LE Subevents

- #define **HCI_LE_CONN_CMPL_EVT** 0x01
- #define **HCI_LE_ADV_REPORT_EVT** 0x02
- #define **HCI_LE_CONN_UPDATE_CMPL_EVT** 0x03
- #define **HCI_LE_READ_REMOTE_FEAT_CMPL_EVT** 0x04
- #define **HCI_LE_LTK_REQ_EVT** 0x05
- #define **HCI_LE_REM_CONN_PARAM_REQ_EVT** 0x06
- #define **HCI_LE_DATA_LEN_CHANGE_EVT** 0x07
- #define **HCI_LE_READ_LOCAL_P256_PUB_KEY_CMPL_EVT** 0x08
- #define **HCI_LE_GENERATE_DHKEY_CMPL_EVT** 0x09
- #define **HCI_LE_ENHANCED_CONN_CMPL_EVT** 0x0A
- #define **HCI_LE_DIRECT_ADV_REPORT_EVT** 0x0B
- #define **HCI_LE_PHY_UPDATE_CMPL_EVT** 0x0C
- #define **HCI_LE_EXT_ADV_REPORT_EVT** 0x0D
- #define **HCI_LE_PER_ADV_SYNC_EST_EVT** 0x0E
- #define **HCI_LE_PER_ADV_REPORT_EVT** 0x0F
- #define **HCI_LE_PER_ADV_SYNC_LOST_EVT** 0x10
- #define **HCI_LE_SCAN_TIMEOUT_EVT** 0x11
- #define **HCI_LE_ADV_SET_TERM_EVT** 0x12
- #define **HCI_LE_SCAN_REQ_RCVD_EVT** 0x13
- #define **HCI_LE_CH_SEL_ALGO_EVT** 0x14
- #define **HCI_LE_CONNLESS_IQ_REPORT_EVT** 0x15
- #define **HCI_LE_CONN_IQ_REPORT_EVT** 0x16

- #define **HCI_LE_CTE_REQ_FAILED_EVT** 0x17
- #define **HCI_LE_PER_SYNC_TRSF_RCVD_EVT** 0x18
- #define **HCI_LE_CIS_EST_EVT** 0x19
- #define **HCI_LE_CIS_REQ_EVT** 0x1A
- #define **HCI_LE_CREATE_BIG_CMPL_EVT** 0x1B
- #define **HCI_LE_TERMINATE_BIG_CMPL_EVT** 0x1C
- #define **HCI_LE_BIG_SYNC_EST_EVT** 0x1D
- #define **HCI_LE_BIG_SYNC_LOST_EVT** 0x1E
- #define **HCI_LE_REQ_PEER_SCA_CMPLT_EVT** 0x1F
- #define **HCI_LE_PATH_LOSS_REPORT_EVT** 0x20
- #define **HCI_LE_POWER_REPORT_EVT** 0x21
- #define **HCI_LE_BIG_INFO_ADV_REPORT_EVT** 0x22

Event parameter lengths

- #define HCI_LEN_DISCONNECT_CMPL 4
- #define HCI_LEN_READ_REMOTE_VER_INFO_CMPL 8
- #define HCI_LEN_CMD_CMPL 3
- #define HCI_LEN_CMD_STATUS 4
- #define HCI_LEN_HW_ERR 1
- #define HCI_LEN_NUM_CMPL_PKTS(numHdls) (1 + (4 ∗ numHdls))
- #define HCI_LEN_ENC_CHANGE 4
- #define HCI_LEN_ENC_KEY_REFRESH_CMPL 3
- #define HCI_LEN_LE_CONN_CMPL 19
- #define HCI_LEN_LE_ADV_RPT_MIN 12
- #define HCI_LEN_LE_CONN_UPDATE_CMPL 10
- #define HCI_LEN_LE_READ_REMOTE_FEAT_CMPL 12
- #define HCI_LEN_LE_LTK_REQ 13
- #define HCI_LEN_LE_REM_CONN_PARAM_REQ 11
- #define HCI_LEN_LE_DATA_LEN_CHANGE 11
- #define HCI_LEN_LE_READ_PUB_KEY_CMPL 66
- #define HCI_LEN_LE_GEN_DHKEY_CMPL 34
- #define HCI_LEN_LE_ENHANCED_CONN_CMPL 31
- #define HCI_LEN_LE_DIRECT_ADV_REPORT 18
- #define HCI_LEN_AUTH_PAYLOAD_TIMEOUT 2
- #define HCI_LEN_LE_PHY_UPDATE_CMPL 6
- #define HCI_LEN_LE_PHY_UPDATE_CMPL 6
- #define HCI_LEN_LE_CH_SEL_ALGO 4
- #define HCI_LEN_LE_EXT_ADV_REPORT_MIN 26
- #define HCI_LEN_LE_PER_ADV_SYNC_EST 16
- #define HCI_LEN_LE_PER_ADV_REPORT 8
- #define HCI_LEN_LE_PER_ADV_SYNC_LOST 3
- #define HCI_LEN_LE_SCAN_TIMEOUT 1
- #define HCI_LEN_LE_ADV_SET_TERM 6
- #define HCI_LEN_LE_SCAN_REQ_RCVD 9
- #define HCI_LEN_LE_PER_SYNC_TRSF_RCVT 20
- #define HCI_LEN_LE_CIS_EST 29
- #define HCI_LEN_LE_CIS_REQ 7
- #define HCI_LEN_LE_PEER_SCA_CMPL 5
- #define HCI_LEN_LE_CREATE_BIG_CMPL(numBis) (19 + (2 ∗ numBis))
- #define HCI_LEN_LE_TERMINATE_BIG_CMPL 3
- #define HCI_LEN_LE_BIG_SYNC_EST(numBis) (15 + (2 ∗ numBis))
- #define HCI_LEN_LE_BIG_SYNC_LOST 3
- #define HCI_LEN_LE_POWER_REPORT 9
- #define HCI_LEN_LE_PATH_LOSS_ZONE 5
- #define HCI_LEN_LE_BIG_INFO_ADV_REPORT 20

Supported commands

- #define HCI_SUP_DISCONNECT 0x20
- #define HCI_SUP_READ_REMOTE_VER_INFO 0x80
- #define HCI_SUP_SET_EVENT_MASK 0x40
- #define HCI_SUP_RESET 0x80
- #define HCI_SUP_READ_TX_PWR_LVL 0x04
- #define HCI_SUP_READ_LOCAL_VER_INFO 0x08
- #define HCI_SUP_READ_LOCAL_SUP_FEAT 0x20
- #define HCI_SUP_READ_BD_ADDR 0x02
- #define HCI_SUP_READ_RSSI 0x20
- #define HCI_SUP_SET_EVENT_MASK_PAGE2 0x04
- #define HCI_SUP_LE_SET_EVENT_MASK 0x01
- #define HCI_SUP_LE_READ_BUF_SIZE 0x02
- #define HCI_SUP_LE_READ_LOCAL_SUP_FEAT 0x04
- #define HCI_SUP_LE_SET_RAND_ADDR 0x10
- #define HCI_SUP_LE_SET_ADV_PARAM 0x20
- #define HCI_SUP_LE_READ_ADV_TX_POWER 0x40
- #define HCI_SUP_LE_SET_ADV_DATA 0x80
- #define HCI_SUP_LE_SET_SCAN_RESP_DATA 0x01
- #define HCI_SUP_LE_SET_ADV_ENABLE 0x02
- #define HCI_SUP_LE_SET_SCAN_PARAM 0x04
- #define HCI_SUP_LE_SET_SCAN_ENABLE 0x08
- #define HCI_SUP_LE_CREATE_CONN 0x10
- #define HCI_SUP_LE_CREATE_CONN_CANCEL 0x20
- #define HCI_SUP_LE_READ_WHITE_LIST_SIZE 0x40
- #define HCI_SUP_LE_CLEAR_WHITE_LIST 0x80
- #define HCI_SUP_LE_ADD_DEV_WHITE_LIST 0x01
- #define HCI_SUP_LE_REMOVE_DEV_WHITE_LIST 0x02
- #define HCI_SUP_LE_CONN_UPDATE 0x04
- #define HCI_SUP_LE_SET_HOST_CHAN_CLASS 0x08
- #define HCI_SUP_LE_READ_CHAN_MAP 0x10
- #define HCI_SUP_LE_READ_REMOTE_FEAT 0x20
- #define HCI_SUP_LE_ENCRYPT 0x40
- #define HCI_SUP_LE_RAND 0x80
- #define HCI_SUP_LE_START_ENCRYPTION 0x01
- #define HCI_SUP_LE_LTK_REQ_REPL 0x02
- #define HCI_SUP_LE_LTK_REQ_NEG_REPL 0x04
- #define HCI_SUP_LE_READ_SUP_STATES 0x08
- #define HCI_SUP_LE_RECEIVER_TEST 0x10
- #define HCI_SUP_LE_TRANSMITTER_TEST 0x20
- #define HCI_SUP_LE_TEST_END 0x40
- #define HCI_SUP_READ_AUTH_PAYLOAD_TO 0x10
- #define HCI_SUP_WRITE_AUTH_PAYLOAD_TO 0x20
- #define HCI_SUP_LE_REM_CONN_PARAM_REQ_REPL 0x10
- #define HCI_SUP_LE_REM_CONN_PARAM_REQ_NEG_REPL 0x20
- #define HCI_SUP_LE_SET_DATA_LEN 0x40
- #define HCI_SUP_LE_READ_DEF_DATA_LEN 0x80
- #define HCI_SUP_LE_WRITE_DEF_DATA_LEN 0x01
- #define HCI_SUP_LE_READ_LOCAL_P256_PUB_KEY 0x02
- #define HCI_SUP_LE_GENERATE_DHKEY 0x04
- #define HCI_SUP_LE_ADD_DEV_RES_LIST_EVT 0x08
- #define HCI_SUP_LE_REMOVE_DEV_RES_LIST 0x10
- #define HCI_SUP_LE_CLEAR_RES_LIST 0x20
- #define HCI_SUP_LE_READ_RES_LIST_SIZE 0x40

- #define HCI_SUP_LE_READ_PEER_RES_ADDR 0x80
- #define HCI_SUP_LE_READ_LOCAL_RES_ADDR 0x01
- #define HCI_SUP_LE_SET_ADDR_RES_ENABLE 0x02
- #define HCI_SUP_LE_SET_RES_PRIV_ADDR_TO 0x04
- #define HCI_SUP_LE_READ_MAX_DATA_LEN 0x08
- #define HCI_SUP_LE_READ_PHY 0x10
- #define HCI_SUP_LE_SET_DEF_PHY 0x20
- #define HCI_SUP_LE_SET_PHY 0x40
- #define HCI_SUP_LE_ENHANCED_RECEIVER_TEST 0x80
- #define HCI_SUP_LE_ENHANCED_TRANSMITTER_TEST 0x01
- #define HCI_SUP_LE_SET_ADV_SET_RAND_ADDR 0x02
- #define HCI_SUP_LE_SET_EXT_ADV_PARAM 0x04
- #define HCI_SUP_LE_SET_EXT_ADV_DATA 0x08
- #define HCI_SUP_LE_SET_EXT_SCAN_RESP_DATA 0x10
- #define HCI_SUP_LE_SET_EXT_ADV_ENABLE 0x20
- #define HCI_SUP_LE_READ_MAX_ADV_DATA_LEN 0x40
- #define HCI_SUP_LE_READ_NUM_OF_SUP_ADV_SETS 0x80
- #define HCI_SUP_LE_REMOVE_ADV_SET 0x01
- #define HCI_SUP_LE_CLEAR_ADV_SETS 0x02
- #define HCI_SUP_LE_SET_PER_ADV_PARAM 0x04
- #define HCI_SUP_LE_SET_PER_ADV_DATA 0x08
- #define HCI_SUP_LE_SET_PER_ADV_ENABLE 0x10
- #define HCI_SUP_LE_SET_EXT_SCAN_PARAM 0x20
- #define HCI_SUP_LE_SET_EXT_SCAN_ENABLE 0x40
- #define HCI_SUP_LE_EXT_CREATE_CONN 0x80
- #define HCI_SUP_LE_PER_ADV_CREATE_SYNC 0x01
- #define HCI_SUP_LE_PER_ADV_CREATE_SYNC_CANCEL 0x02
- #define HCI_SUP_LE_PER_ADV_TERMINATE_SYNC 0x04
- #define HCI_SUP_LE_ADD_DEV_PER_ADV_LIST 0x08
- #define HCI_SUP_LE_REMOVE_DEV_PER_ADV_LIST 0x10
- #define HCI_SUP_LE_CLEAR_PER_ADV_LIST 0x20
- #define HCI_SUP_LE_READ_PER_ADV_LIST_SIZE 0x40
- #define HCI_SUP_LE_READ_TX_POWER 0x80
- #define HCI_SUP_LE_READ_RF_PATH_COMP 0x01
- #define HCI_SUP_LE_WRITE_RF_PATH_COMP 0x02
- #define HCI_SUP_LE_SET_PRIVACY_MODE 0x04
- #define HCI_SUP_LE_RECEIVER_TEST_V3 0x08
- #define HCI_SUP_LE_TRANSMITTER_TEST_V3 0x10
- #define HCI_SUP_LE_SET_CONNLESS_CTE_TX_PARAMS 0x20
- #define HCI_SUP_LE_SET_CONNLESS_CTE_TX_ENABLE 0x40
- #define HCI_SUP_LE_SET_CONNLESS_IQ_SAMP_ENABLE 0x80
- #define HCI_SUP_LE_SET_CONN_CTE_RX_PARAMS 0x01
- #define HCI_SUP_LE_SET_CONN_CTE_TX_PARAMS 0x02
- #define HCI_SUP_LE_CONN_CTE_REQ_ENABLE 0x04
- #define HCI_SUP_LE_CONN_CTE_RSP_ENABLE 0x08
- #define HCI_SUP_LE_READ_ANTENNA_INFO 0x10
- #define HCI_SUP_LE_SET_PER_ADV_RCV_ENABLE 0x20
- #define HCI_SUP_LE_PER_ADV_SYNC_TRANSFER 0x40
- #define HCI_SUP_LE_PER_ADV_SET_INFO_TRANSFER 0x80
- #define HCI_SUP_LE_SET_PAST_PARAM 0x01
- #define HCI_SUP_LE_SET_DEFAULT_PAST_PARAM 0x02
- #define HCI_SUP_LE_GENERATE_DHKEY_V2 0x04
- #define HCI_SUP_LE_MODIFY_SLEEP_CLK_ACCURACY 0x10
- #define HCI_SUP_LE_READ_BUF_SIZE_V2 0x20
- #define HCI_SUP_LE_READ_ISO_TX_SYNC 0x40

- #define HCI_SUP_LE_SET_CIG_PARAM 0x80
- #define HCI_SUP_LE_SET_CIG_PARAM_TEST 0x01
- #define HCI_SUP_LE_CREATE_CIS 0x02
- #define HCI_SUP_LE_REMOVE_CIG 0x04
- #define HCI_SUP_LE_ACCEPT_CIS_REQ 0x08
- #define HCI_SUP_LE_REJECT_CIS_REQ 0x10
- #define HCI_SUP_LE_CREATE_BIG 0x20
- #define HCI_SUP_LE_CREATE_BIG_TEST 0x40
- #define HCI_SUP_LE_TERMINATE_BIG 0x80
- #define HCI_SUP_LE_BIG_CREATE_SYNC 0x01
- #define HCI_SUP_LE_BIG_TERMINATE_SYNC 0x02
- #define HCI_SUP_LE_REQ_PEER_SCA 0x04
- #define HCI_SUP_LE_SETUP_ISO_DATA_PATH 0x08
- #define HCI_SUP_LE_REMOVE_ISO_DATA_PATH 0x10
- #define HCI_SUP_LE_ISO_TRANSMIT_TEST 0x20
- #define HCI_SUP_LE_ISO_RECEIVE_TEST 0x40
- #define HCI_SUP_LE_ISO_READ_TEST_COUNTERS 0x80
- #define HCI_SUP_LE_ISO_TEST_END 0x01
- #define HCI_SUP_LE_SET_HOST_FEATURE 0x02
- #define HCI_SUP_LE_READ_ISO_LINK_QUALITY 0x04
- #define HCI_SUP_LE_ENH_READ_TX_POWER_LEVEL 0x08
- #define HCI_SUP_LE_READ_REMOTE_TX_POWER_LEVEL 0x01
- #define HCI_SUP_LE_SET_PATH_LOSS_REPORT_PARAM 0x02
- #define HCI_SUP_LE_SET_PATH_LOSS_REPORT_ENABLE 0x04
- #define HCI_SUP_LE_SET_TX_POWER_REPORT_ENABLE 0x08
- #define HCI_SUP_LE_TRANSMITTER_TEST_V4 0x01
- #define HCI_SUP_READ_LOCAL_SUP_CODECS_V2 0x02
- #define HCI_SUP_READ_LOCAL_SUP_CODEC_CAP 0x04
- #define HCI_SUP_READ_LOCAL_SUP_CTR_DLY 0x08
- #define HCI_SUP_CONFIG_DATA_PATH 0x10
- #define HCI_SUP_CMD_LEN 64

Event mask

- #define HCI_EVT_MASK_DISCONNECT_CMPL 0x10
- #define HCI_EVT_MASK_ENC_CHANGE 0x80
- #define HCI_EVT_MASK_READ_REMOTE_VER_INFO_CMPL 0x08
- #define HCI_EVT_MASK_HW_ERROR 0x80
- #define HCI_EVT_MASK_DATA_BUF_OVERFLOW 0x02
- #define HCI_EVT_MASK_ENC_KEY_REFRESH_CMPL 0x80
- #define HCI_EVT_MASK_LE_META 0x20

Event mask page 2

- #define HCI_EVT_MASK_AUTH_PAYLOAD_TIMEOUT 0x80

LE event mask

- #define HCI_EVT_MASK_LE_CONN_CMPL_EVT 0x01
- #define HCI_EVT_MASK_LE_ADV_REPORT_EVT 0x02
- #define HCI_EVT_MASK_LE_CONN_UPDATE_CMPL_EVT 0x04
- #define HCI_EVT_MASK_LE_READ_REMOTE_FEAT_CMPL_EVT 0x08
- #define HCI_EVT_MASK_LE_LTK_REQ_EVT 0x10
- #define HCI_EVT_MASK_LE_REMOTE_CONN_PARAM_REQ_EVT 0x20
- #define HCI_EVT_MASK_LE_DATA_LEN_CHANGE_EVT 0x40
- #define HCI_EVT_MASK_LE_READ_LOCAL_P256_PUB_KEY_CMPL 0x80
- #define HCI_EVT_MASK_LE_GENERATE_DHKEY_CMPL 0x01
- #define HCI_EVT_MASK_LE_ENHANCED_CONN_CMPL_EVT 0x02
- #define HCI_EVT_MASK_LE_DIRECT_ADV_REPORT_EVT 0x04
- #define HCI_EVT_MASK_LE_PHY_UPDATE_CMPL_EVT 0x08
- #define HCI_EVT_MASK_LE_EXT_ADV_REPORT_EVT 0x10
- #define HCI_EVT_MASK_LE_PER_ADV_SYNC_EST_EVT 0x20
- #define HCI_EVT_MASK_LE_PER_ADV_REPORT_EVT 0x40
- #define HCI_EVT_MASK_LE_PER_ADV_SYNC_LOST_EVT 0x80
- #define HCI_EVT_MASK_LE_SCAN_TIMEOUT_EVT 0x01
- #define HCI_EVT_MASK_LE_ADV_SET_TERM_EVT 0x02
- #define HCI_EVT_MASK_LE_SCAN_REQ_RCVD_EVT 0x04
- #define HCI_EVT_MASK_LE_CH_SEL_ALGO_EVT 0x08
- #define HCI_EVT_MASK_LE_CONNLESS_IQ_REPORT_EVT 0x10
- #define HCI_EVT_MASK_LE_CONN_IQ_REPORT_EVT 0x20
- #define HCI_EVT_MASK_LE_CTE_REQ_FAILED_EVT 0x40
- #define HCI_EVT_MASK_LE_PER_SYNC_TRSF_RCVT_EVT 0x80
- #define HCI_EVT_MASK_LE_CIS_EST_EVT 0x01
- #define HCI_EVT_MASK_LE_CIS_REQ_EVT 0x02
- #define HCI_EVT_MASK_LE_CREATE_BIG_CMPL_EVT 0x04
- #define HCI_EVT_MASK_LE_TERMINATE_BIG_CMPL_EVT 0x08
- #define HCI_EVT_MASK_LE_BIG_SYNC_EST_EVT 0x10
- #define HCI_EVT_MASK_LE_BIG_SYNC_LOST_EVT 0x20
- #define HCI_EVT_MASK_LE_PEER_SCA_CMPL_EVT 0x40
- #define HCI_EVT_MASK_LE_PATH_LOSS_REPORT_EVT 0x80
- #define HCI_EVT_MASK_LE_TX_POWER_REPORT_EVT 0x01
- #define HCI_EVT_MASK_LE_BIG_INFO_ADV_RPT_EVT 0x02

LE supported features

- #define HCI_LE_SUP_FEAT_ENCRYPTION 0x0000000000000001
- #define HCI_LE_SUP_FEAT_CONN_PARAM_REQ_PROC 0x0000000000000002
- #define HCI_LE_SUP_FEAT_EXT_REJECT_IND 0x0000000000000004
- #define HCI_LE_SUP_FEAT_SLV_INIT_FEAT_EXCH 0x0000000000000008
- #define HCI_LE_SUP_FEAT_LE_PING 0x0000000000000010
- #define HCI_LE_SUP_FEAT_DATA_LEN_EXT 0x0000000000000020
- #define HCI_LE_SUP_FEAT_PRIVACY 0x0000000000000040
- #define HCI_LE_SUP_FEAT_EXT_SCAN_FILT_POLICY 0x0000000000000080
- #define HCI_LE_SUP_FEAT_LE_2M_PHY 0x0000000000000100
- #define HCI_LE_SUP_FEAT_STABLE_MOD_IDX_TRANSMITTER 0x0000000000000200
- #define HCI_LE_SUP_FEAT_STABLE_MOD_IDX_RECEIVER 0x0000000000000400
- #define HCI_LE_SUP_FEAT_LE_CODED_PHY 0x0000000000000800
- #define HCI_LE_SUP_FEAT_LE_EXT_ADV 0x0000000000001000
- #define HCI_LE_SUP_FEAT_LE_PER_ADV 0x0000000000002000
- #define HCI_LE_SUP_FEAT_CH_SEL_2 0x0000000000004000

- #define HCI_LE_SUP_FEAT_LE_POWER_CLASS_1 0x0000000000008000
- #define HCI_LE_SUP_FEAT_MIN_NUN_USED_CHAN 0x0000000000010000
- #define HCI_LE_SUP_FEAT_CONN_CTE_REQ 0x0000000000020000
- #define HCI_LE_SUP_FEAT_CONN_CTE_RSP 0x0000000000040000
- #define HCI_LE_SUP_FEAT_CONNLESS_CTE_TRANS 0x0000000000080000
- #define HCI_LE_SUP_FEAT_CONNLESS_CTE_RECV 0x0000000000100000
- #define HCI_LE_SUP_FEAT_ANTENNA_SWITCH_AOD 0x0000000000200000
- #define HCI_LE_SUP_FEAT_ANTENNA_SWITCH_AOA 0x0000000000400000
- #define HCI_LE_SUP_FEAT_RECV_CTE 0x0000000000800000
- #define HCI_LE_SUP_FEAT_PAST_SENDER 0x0000000001000000
- #define HCI_LE_SUP_FEAT_PAST_RECIPIENT 0x0000000002000000
- #define HCI_LE_SUP_FEAT_SCA_UPDATE 0x0000000004000000
- #define HCI_LE_SUP_FEAT_REMOTE_PUB_KEY_VALIDATION 0x0000000008000000
- #define HCI_LE_SUP_FEAT_CIS_MASTER 0x0000000010000000
- #define HCI_LE_SUP_FEAT_CIS_SLAVE 0x0000000020000000
- #define HCI_LE_SUP_FEAT_ISO_BROADCASTER 0x0000000040000000
- #define HCI_LE_SUP_FEAT_ISO_SYNC_RECEIVER 0x0000000080000000
- #define HCI_LE_SUP_FEAT_ISO_HOST_SUPPORT 0x0000000100000000
- #define HCI_LE_SUP_FEAT_POWER_CONTROL_REQUEST 0x0000000200000000
- #define HCI_LE_SUP_FEAT_POWER_CHANGE_IND 0x0000000400000000
- #define HCI_LE_SUP_FEAT_PATH_LOSS_MONITOR 0x0000000800000000

LE feature bit positon in FeatureSet stored in the Controller

- #define HCI_LE_FEAT_BIT_ISO_HOST_SUPPORT 32

Advertising command parameters

- #define HCI_ADV_MIN_INTERVAL 0x0020
- #define HCI_ADV_MAX_INTERVAL 0x4000
- #define HCI_ADV_DIRECTED_MAX_DURATION 0x0500
- #define HCI_ADV_TYPE_CONN_UNDIRECT 0x00
- #define HCI_ADV_TYPE_CONN_DIRECT 0x01
- #define HCI_ADV_TYPE_DISC_UNDIRECT 0x02
- #define HCI_ADV_TYPE_NONCONN_UNDIRECT 0x03
- #define HCI_ADV_TYPE_CONN_DIRECT_LO_DUTY 0x04
- #define HCI_ADV_CHAN_37 0x01
- #define HCI_ADV_CHAN_38 0x02
- #define HCI_ADV_CHAN_39 0x04
- #define HCI_ADV_FILT_NONE 0x00
- #define HCI_ADV_FILT_SCAN 0x01
- #define HCI_ADV_FILT_CONN 0x02
- #define HCI_ADV_FILT_ALL 0x03

Scan command parameters

- #define HCI_SCAN_TYPE_PASSIVE 0
- #define HCI_SCAN_TYPE_ACTIVE 1
- #define HCI_SCAN_INTERVAL_MIN 0x0004
- #define HCI_SCAN_INTERVAL_MAX 0x4000
- #define HCI_SCAN_INTERVAL_DEFAULT 0x0010
- #define HCI_SCAN_WINDOW_MIN 0x0004
- #define HCI_SCAN_WINDOW_MAX 0x4000
- #define HCI_SCAN_WINDOW_DEFAULT 0x0010

Connection command parameters

- #define HCI_CONN_INTERVAL_MIN 0x0006
- #define HCI_CONN_INTERVAL_MAX 0x0C80
- #define HCI_CONN_LATENCY_MAX 0x01F3
- #define HCI_SUP_TIMEOUT_MIN 0x000A
- #define HCI_SUP_TIMEOUT_MAX 0x0C80

Connection event parameters

- #define HCI_CLOCK_500PPM 0x00
- #define HCI_CLOCK_250PPM 0x01
- #define HCI_CLOCK_150PPM 0x02
- #define HCI_CLOCK_100PPM 0x03
- #define HCI_CLOCK_75PPM 0x04
- #define HCI_CLOCK_50PPM 0x05
- #define HCI_CLOCK_30PPM 0x06
- #define HCI_CLOCK_20PPM 0x07

Advertising report event parameters

- #define HCI_ADV_CONN_UNDIRECT 0x00
- #define HCI_ADV_CONN_DIRECT 0x01
- #define HCI_ADV_DISC_UNDIRECT 0x02
- #define HCI_ADV_NONCONN_UNDIRECT 0x03
- #define HCI_ADV_SCAN_RESPONSE 0x04

Extended advertising data operations

- #define HCI_ADV_DATA_OP_FRAG_INTER 0x00
- #define HCI_ADV_DATA_OP_FRAG_FIRST 0x01
- #define HCI_ADV_DATA_OP_FRAG_LAST 0x02
- #define HCI_ADV_DATA_OP_COMP_FRAG 0x03
- #define HCI_ADV_DATA_OP_UNCHANGED_DATA 0x04

Advertising data fragment preference

- #define HCI_ADV_DATA_FRAG_PREF_FRAG 0x00
- #define HCI_ADV_DATA_FRAG_PREF_NO_FRAG 0x01

Number of advertising sets

- #define HCI_ADV_NUM_SETS_ALL_DISABLE 0x00

Maximum number of scanning or initiating PHYs

- #define HCI_MAX_NUM_PHYS 3

Advertising PHY values

- #define HCI_ADV_PHY_LE_1M 0x01
- #define HCI_ADV_PHY_LE_2M 0x02
- #define HCI_ADV_PHY_LE_CODED 0x03

Scanner PHY value bits

- #define HCI_SCAN_PHY_LE_1M_BIT (1<<0)
- #define HCI_SCAN_PHY_LE_2M_BIT (1<<1)
- #define HCI_SCAN_PHY_LE_CODED_BIT (1<<2)

Initiator PHY value bits

- #define HCI_INIT_PHY_LE_1M_BIT (1<<0)
- #define HCI_INIT_PHY_LE_2M_BIT (1<<1)
- #define HCI_INIT_PHY_LE_CODED_BIT (1<<2)

Transmitter PHY value bits

- #define HCI_TRANS_PHY_LE_1M_BIT (1<<0)
- #define HCI_TRANS_PHY_LE_2M_BIT (1<<1)
- #define HCI_TRABS_PHY_LE_CODED_BIT (1<<2)

Advertising event properties type bits

- #define HCI_ADV_PROP_CONN_ADV_BIT (1<<0)
- #define HCI_ADV_PROP_SCAN_ADV_BIT (1<<1)
- #define HCI_ADV_PROP_DIRECT_ADV_BIT (1<<2)
- #define HCI_ADV_PROP_CONN_DIRECT_ADV_BIT (1<<3)
- #define HCI_ADV_PROP_USE_LEG_PDU_BIT (1<<4)
- #define HCI_ADV_PROP_OMIT_ADV_ADDR_BIT (1<<5)
- #define HCI_ADV_PROP_INC_TX_PWR_BIT (1<<6)

Advertising event properties for legacy PDUs

- #define HCI_ADV_PROP_LEG_CONN_UNDIRECT 0x13
- #define HCI_ADV_PROP_LEG_CONN_DIRECT 0x1D
- #define HCI_ADV_PROP_LEG_SCAN_UNDIRECT 0x12
- #define HCI_ADV_PROP_LEG_NONCONN_UNDIRECT 0x10
- #define HCI_ADV_PROP_LEG_CONN_DIRECT_LO_DUTY 0x15

Extended advertising report event type bits

- #define HCI_ADV_RPT_CONN_ADV_BIT (1<<0)
- #define HCI_ADV_RPT_SCAN_ADV_BIT (1<<1)
- #define HCI_ADV_RPT_DIRECT_ADV_BIT (1<<2)
- #define HCI_ADV_RPT_SCAN_RSP_BIT (1<<3)
- #define HCI_ADV_RPT_LEG_ADV_BIT (1<<4)
- #define HCI_ADV_RPT_DATA_STATUS_BITS (3<<5)

Advertising report event types for legacy PDUs

- #define HCI_ADV_RPT_LEG_CONN_UNDIRECT 0x13
- #define HCI_ADV_RPT_LEG_CONN_DIRECT 0x15
- #define HCI_ADV_RPT_LEG_SCAN_UNDIRECT 0x12
- #define HCI_ADV_RPT_LEG_NONCONN_UNDIRECT 0x10
- #define HCI_ADV_RPT_LEG_CONN_UNDIRECT_SCAN_RSP 0x1B
- #define HCI_ADV_RPT_LEG_SCAN_UNDIRECT_SCAN_RSP 0x1A

Advertising report data status

- #define HCI_ADV_RPT_DATA_CMPL 0x00
- #define HCI_ADV_RPT_DATA_INCMPL_MORE 0x01
- #define HCI_ADV_RPT_DATA_INCMPL_TRUNC 0x02

Extended advertising report event primary PHY values

- #define HCI_ADV_RPT_PHY_PRIM_LE_1M 0x01
- #define HCI_ADV_RPT_PHY_PRIM_LE_CODED 0x03

Extended advertising report event seconday PHY values

- #define HCI_ADV_RPT_PHY_SEC_NONE 0x00
- #define HCI_ADV_RPT_PHY_SEC_LE_1M 0x01
- #define HCI_ADV_RPT_PHY_SEC_LE_2M 0x02
- #define HCI_ADV_RPT_PHY_SEC_LE_CODED 0x03

Channel selection algorithm used

- #define HCI_CH_SEL_ALGO_1 0x00
- #define HCI_CH_SEL_ALGO_2 0x01

KeyType parameters

- #define HCI_PRIVATE_KEY_GENERATED 0x00
- #define HCI_PRIVATE_KEY_DEBUG 0x01

Minimum number of used channels

- #define HCI_MIN_NUM_OF_USED_CHAN 8

Synchronization timeout for the periodic advertising

- #define HCI_SYNC_MIN_TIMEOUT 0x000A
- #define HCI_SYNC_MAX_TIMEOUT 0x4000

Maximum synchronization skip

- #define HCI_SYNC_MAX_SKIP 0x01F3

Maximum synchronization handle

- #define HCI_SYNC_MAX_HANDLE 0x0EFF

Periodic sync transfer receive mode

- #define HCI_SYNC_TRSF_MODE_OFF 0x00
- #define HCI_SYNC_TRSF_MODE_REP_DISABLED 0x01,
- #define HCI_SYNC_TRSF_MODE_REP_ENABLED 0x02,

Periodic advertising create sync options bits

- #define HCI_OPTIONS_FILT_POLICY_BIT (1<<0)
- #define HCI_OPTIONS_INIT_RPT_ENABLE_BIT (1<<1)

Misc command parameters

- #define HCI_ROLE_MASTER 0
- #define HCI_ROLE_MASTER 0
- #define HCI_ROLE_SLAVE 1
- #define HCI_ROLE_SLAVE 1
- #define HCI_READ_TX_PWR_CURRENT 0
- #define HCI_READ_TX_PWR_MAX 1
- #define HCI_TX_PWR_MIN -30
- #define HCI_TX_PWR_MAX 20
- #define HCI_TX_PWR_NO_PREFERENCE 127
- #define HCI_VERSION 6
- #define HCI_RSSI_MIN -127
- #define HCI_RSSI_MAX 20
- #define HCI_ADDR_TYPE_PUBLIC 0
- #define HCI_ADDR_TYPE_RANDOM 1
- #define HCI_ADDR_TYPE_PUBLIC_IDENTITY 2
- #define HCI_ADDR_TYPE_RANDOM_IDENTITY 3
- #define HCI_ADDR_TYPE_ANONYMOUS 0xFF
- #define HCI_FILT_NONE 0
- #define HCI_FILT_WHITE_LIST 1
- #define HCI_FILT_RES_INIT 2
- #define HCI_FILT_WHITE_LIST_RES_INIT 3
- #define HCI_FILT_PER_ADV_PARAM 0
- #define HCI_FILT_PER_ADV_LIST 1
- #define HCI_PRIV_MODE_NETWORK 0x00
- #define HCI_PRIV_MODE_DEVICE 0x01

PHY types

- #define HCI_PHY_NONE 0x00
- #define HCI_PHY_LE_1M_BIT (1<<0)
- #define HCI_PHY_LE_2M_BIT (1<<1)
- #define HCI_PHY_LE_CODED_BIT (1<<2)

All PHYs preference

- #define HCI_ALL_PHY_ALL_PREFERENCES 0x00
- #define HCI_ALL_PHY_TX_PREFERENCE_BIT (1<<0)
- #define HCI_ALL_PHY_RX_PREFERENCE_BIT (1<<1)

PHY options

- #define HCI_PHY_OPTIONS_NONE 0x00
- #define HCI_PHY_OPTIONS_S2_PREFERRED 0x01
- #define HCI_PHY_OPTIONS_S8_PREFERRED 0x02

CTE Slot Durations

- #define HCI_CTE_SLOT_DURATION_NONE 0x00
- #define HCI_CTE_SLOT_DURATION_1_US 0x01
- #define HCI_CTE_SLOT_DURATION_2_US 0x02

Permitted CTE Type bits

- #define HCI_CTE_TYPE_PERMIT_AOA_RSP_BIT (1<<0)
- #define HCI_CTE_TYPE_PERMIT_AOD_RSP_1_US_BIT (1<<1)
- #define HCI_CTE_TYPE_PERMIT_AOD_RSP_2_US_BIT (1<<2)

Requested CTE Types

- #define HCI_CTE_TYPE_REQ_AOA 0x00
- #define HCI_CTE_TYPE_REQ_AOD_1_US 0x01
- #define HCI_CTE_TYPE_REQ_AOD_2_US 0x02

Bluetooth core specification versions

- #define HCI_VER_BT_CORE_SPEC_4_0 0x06
- #define HCI_VER_BT_CORE_SPEC_4_1 0x07
- #define HCI_VER_BT_CORE_SPEC_4_2 0x08
- #define HCI_VER_BT_CORE_SPEC_5_0 0x09
- #define HCI_VER_BT_CORE_SPEC_5_1 0x0A
- #define HCI_VER_BT_CORE_SPEC_5_2 0x0B

Parameter lengths

- #define HCI_EVT_MASK_LEN 8
- #define HCI_EVT_MASK_PAGE_2_LEN 8
- #define HCI_LE_EVT_MASK_LEN 8
- #define HCI_FEAT_LEN 8
- #define HCI_ADV_DATA_LEN 31
- #define HCI_SCAN_DATA_LEN 31
- #define HCI_EXT_ADV_DATA_LEN 251
- #define HCI_EXT_ADV_CONN_DATA_LEN 191
- #define HCI_PER_ADV_DATA_LEN 252
- #define HCI_EXT_ADV_RPT_DATA_LEN 229
- #define HCI_PER_ADV_RPT_DATA_LEN 247
- #define HCI_CHAN_MAP_LEN 5
- #define HCI_KEY_LEN 16
- #define HCI_ENCRYPT_DATA_LEN 16
- #define HCI_RAND_LEN 8
- #define HCI_LE_STATES_LEN 8
- #define HCI_P256_KEY_LEN 64
- #define HCI_DH_KEY_LEN 32
- #define HCI_BC_LEN 16
- #define HCI_EXT_ADV_RPT_DATA_LEN_OFFSET 23
- #define HCI_PER_ADV_RPT_DATA_LEN_OFFSET 6

Number of Antenna IDs in Switching Pattern

- #define HCI_MIN_NUM_ANTENNA_IDS 2
- #define HCI_MAX_NUM_ANTENNA_IDS 75

IQ Report Sample Counts

- #define HCI_IQ_RPT_SAMPLE_CNT_MIN 9
- #define HCI_IQ_RPT_SAMPLE_CNT_MAX 82
- #define HCI_CONN_IQ_RPT_SAMPLE_CNT_OFFSET 12

CIS Count

- #define HCI_MAX_CIS_COUNT 0x10

BIS Count

- #define HCI_MAX_BIS_COUNT 0x10

CIG IDs

- #define HCI_MIN_CIG_ID 0x00
- #define HCI_MAX_CIG_ID 0xEF

CIS IDs

- #define HCI_MIN_CIS_ID 0x00
- #define HCI_MAX_CIS_ID 0xEF

Packing Scheme

- #define HCI_PACKING_SEQUENTIAL 0x00
- #define HCI_PACKING_INTERLEAVED 0x01

Framing

- #define HCI_FRAMING_UNFRAMED 0x00
- #define HCI_FRAMING_FRAMED 0x01

Slave Clock Accuracy

- #define HCI_MIN_SCA 0x00
- #define HCI_MAX_SCA 0x07

SDU Size

- #define HCI_MIN_SDU_SIZE 0x0000
- #define HCI_MAX_SDU_SIZE 0x0FFF

SDU Interval

- #define HCI_MIN_SDU_INTERV 0x0000FF
- #define HCI_MAX_SDU_INTERV 0x0FFFFF
- #define HCI_DEFAULT_SDU_INTERV 0x004E20

CIS Transport Latency

- #define HCI_MIN_CIS_TRANS_LAT 0x0005
- #define HCI_MAX_CIS_TRANS_LAT 0x0FA0
- #define HCI_DEFAULT_CIS_TRANS_LAT 0x0028

CIS Flush Time

- #define HCI_MIN_CIS_FT 0x01
- #define HCI_MAX_CIS_FT 0xFF

CIS Burst Number

- #define HCI_MIN_CIS_BN 0x00
- #define HCI_MAX_CIS_BN 0x0F

CIS Retransmission Number

- #define HCI_MIN_CIS_RTN 0x00
- #define HCI_MAX_CIS_RTN 0x0F

ISO Data Path Direction

- #define HCI_ISO_DATA_DIR_INPUT 0
- #define HCI_ISO_DATA_DIR_OUTPUT 1

ISO Data Path Direction Bit

- #define HCI_ISO_DATA_PATH_INPUT_BIT (1<<HCI_ISO_DATA_DIR_INPUT)
- #define HCI_ISO_DATA_PATH_OUTPUT_BIT (1<<HCI_ISO_DATA_DIR_OUTPUT)

ISO Data Path ID

- #define HCI_ISO_DATA_PATH_HCI 0x00
- #define HCI_ISO_DATA_PATH_VS 0x01
- #define HCI_ISO_DATA_PATH_DISABLED 0xFF

ISO test packet payload type

- #define HCI_ISO_ISO_PLD_TYPE_ZERO_LEN 0x00
- #define HCI_ISO_ISO_PLD_TYPE_VAR_LEN 0x01
- #define HCI_ISO_ISO_PLD_TYPE_MAX_LEN 0x02

Maximum number of codecs

- #define HCI_MAX_CODEC 5

Maximum length of codec-specific capability data

- #define HCI_CODEC_CAP_DATA_LEN 4

Codec transport types

- #define HCI_CODEC_TRANS_CIS_BIT (1<<2)
- #define HCI_CODEC_TRANS_BIS_BIT (1<<3)

ISO Header Packet Boundary

- #define HCI_ISO_HDR_PB_START_FRAG 0x00
- #define HCI_ISO_HDR_PB_CONT_FRAG 0x01
- #define HCI_ISO_HDR_PB_COMP_FRAG 0x02
- #define HCI_ISO_HDR_PB_END_FRAG 0x03

ISOAL Segmentation Header Start/Continuation Bit

- #define HCI_ISOAL_SEG_HDR_SC_START 0x00
- #define HCI_ISOAL_SEG_HDR_SC_CONT 0x01

Company ID

- #define HCI_ID_PACKETCRAFT 0x07E8
- #define HCI_ID_GREENPEAK 0x0453

  Greenpeak company ID.

Manufacturer location in Local version

- #define HCI_LOCAL_VER_MANUFACTURER_POS 4

Coding Format Assigned Numbers

- #define HCI_ID_LC3 0x01
- #define HCI_ID_VS 0xFF
- #define HCI_CODEC_TRANSPORT_CIS 0x02
- #define HCI_CODEC_TRANSPORT_BIS 0x03

2.1.1 Detailed Description

2.1.2 Macro Definition Documentation

2.1.2.1 HCI_CMD_HDR_LEN

```
#define HCI_CMD_HDR_LEN 3
```

Command packet header length

Definition at line 63 of file hci_defs.h.

2.1.2.2 HCI_ACL_HDR_LEN

```
#define HCI_ACL_HDR_LEN 4
```

ACL packet header length

Definition at line 64 of file hci_defs.h.

2.1.2.3 HCI_ISO_HDR_LEN

```
#define HCI_ISO_HDR_LEN 4
```

ISO packet header length

Definition at line 65 of file hci_defs.h.

2.1.2.4 HCI_EVT_HDR_LEN

```
#define HCI_EVT_HDR_LEN 2
```

Event packet header length

Definition at line 66 of file hci_defs.h.

2.1.2.5 HCI_EVT_PARAM_MAX_LEN

```
#define HCI_EVT_PARAM_MAX_LEN 255
```

Maximum length of event packet parameters

Definition at line 67 of file hci_defs.h.

2.1.2.6 HCI_ACL_DEFAULT_LEN

```
#define HCI_ACL_DEFAULT_LEN 27
```

Default maximum ACL packet length

Definition at line 68 of file hci_defs.h.

2.1.2.7 HCI_PB_FLAG_MASK

```
#define HCI_PB_FLAG_MASK 0x3000
```

ACL packet boundary flag mask

Definition at line 69 of file hci_defs.h.

2.1.2.8 HCI_PB_START_H2C

```
#define HCI_PB_START_H2C 0x0000
```

Packet boundary flag, start, host-to-controller

Definition at line 70 of file hci_defs.h.

2.1.2.9 HCI_PB_CONTINUE

```
#define HCI_PB_CONTINUE 0x1000
```

Packet boundary flag, continue

Definition at line 71 of file hci_defs.h.

2.1.2.10 HCI_PB_START_C2H

```
#define HCI_PB_START_C2H 0x2000
```

Packet boundary flag, start, controller-to-host

Definition at line 72 of file hci_defs.h.

2.1.2.11 HCI_HANDLE_MASK

```
#define HCI_HANDLE_MASK 0x0FFF
```

Mask for handle bits in ACL packet

Definition at line 73 of file hci_defs.h.

2.1.2.12 HCI_HANDLE_NONE

```
#define HCI_HANDLE_NONE 0xFFFF
```

Value for invalid handle

Definition at line 74 of file hci_defs.h.

2.1.2.13 HCI_TS_FLAG_MASK

```
#define HCI_TS_FLAG_MASK (1 << 14)
```

Timestamp flag mask for ISO packets.

Definition at line 77 of file hci_defs.h.

2.1.2.14 HCI_DATA_LOAD_LEN_MASK

```
#define HCI_DATA_LOAD_LEN_MASK 0x3FFF
```

HCI Data load length.

Definition at line 79 of file hci_defs.h.

2.1.2.15 HCI_ISO_DL_MIN_LEN

```
#define HCI_ISO_DL_MIN_LEN 4
```

ISO Data Load header minimum length

Definition at line 82 of file hci_defs.h.

2.1.2.16 HCI_ISO_DL_MAX_LEN

```
#define HCI_ISO_DL_MAX_LEN 8
```

ISO Data Load header maximum length

Definition at line 83 of file hci_defs.h.

2.1.2.17 HCI_ISO_TS_LEN

```
#define HCI_ISO_TS_LEN 4
```

ISO Data Load timestamp length

Definition at line 84 of file hci_defs.h.

2.1.2.18 HCI_ISO_DL_SDU_LEN_MASK

```
#define HCI_ISO_DL_SDU_LEN_MASK 0x0FFF
```

HCI SDU Length mask.

Definition at line 85 of file hci_defs.h.

2.1.2.19 HCI_ISO_DL_PS_MASK

```
#define HCI_ISO_DL_PS_MASK 0xC000
```

HCI Packet status mask.

Definition at line 86 of file hci_defs.h.

2.1.2.20 HCI_CMD_TYPE

```
#define HCI_CMD_TYPE 0x01
```

HCI command packet

Definition at line 93 of file hci_defs.h.

2.1.2.21 HCI_ACL_TYPE

```
#define HCI_ACL_TYPE 0x02
```

HCI ACL data packet

Definition at line 94 of file hci_defs.h.

2.1.2.22 HCI_EVT_TYPE

```
#define HCI_EVT_TYPE 0x04
```

HCI event packet

Definition at line 95 of file hci_defs.h.

2.1.2.23 HCI_ISO_TYPE

```
#define HCI_ISO_TYPE 0x05
```

HCI ISO data packet

Definition at line 96 of file hci_defs.h.

2.1.2.24 HCI_SUCCESS

```
#define HCI_SUCCESS 0x00
```

Success

Definition at line 103 of file hci_defs.h.

2.1.2.25 HCI_ERR_UNKNOWN_CMD

```
#define HCI_ERR_UNKNOWN_CMD 0x01
```

Unknown HCI command

Definition at line 104 of file hci_defs.h.

2.1.2.26 HCI_ERR_UNKNOWN_HANDLE

```
#define HCI_ERR_UNKNOWN_HANDLE 0x02
```

Unknown connection identifier

Definition at line 105 of file hci_defs.h.

2.1.2.27 HCI_ERR_HARDWARE_FAILURE

```
#define HCI_ERR_HARDWARE_FAILURE 0x03
```

Hardware failure

Definition at line 106 of file hci_defs.h.

2.1.2.28 HCI_ERR_PAGE_TIMEOUT

```
#define HCI_ERR_PAGE_TIMEOUT 0x04
```

Page timeout

Definition at line 107 of file hci_defs.h.

2.1.2.29 HCI_ERR_AUTH_FAILURE

```
#define HCI_ERR_AUTH_FAILURE 0x05
```

Authentication failure

Definition at line 108 of file hci_defs.h.

2.1.2.30 HCI_ERR_KEY_MISSING

```
#define HCI_ERR_KEY_MISSING 0x06
```

PIN or key missing

Definition at line 109 of file hci_defs.h.

2.1.2.31 HCI_ERR_MEMORY_EXCEEDED

```
#define HCI_ERR_MEMORY_EXCEEDED 0x07
```

Memory capacity exceeded

Definition at line 110 of file hci_defs.h.

2.1.2.32 HCI_ERR_CONN_TIMEOUT

```
#define HCI_ERR_CONN_TIMEOUT 0x08
```

Connection timeout

Definition at line 111 of file hci_defs.h.

2.1.2.33 HCI_ERR_CONN_LIMIT

```
#define HCI_ERR_CONN_LIMIT 0x09
```

Connection limit exceeded

Definition at line 112 of file hci_defs.h.

2.1.2.34 HCI_ERR_SYNCH_CONN_LIMIT

```
#define HCI_ERR_SYNCH_CONN_LIMIT 0x0A
```

Synchronous connection limit exceeded

Definition at line 113 of file hci_defs.h.

2.1.2.35 HCI_ERR_ACL_CONN_EXISTS

```
#define HCI_ERR_ACL_CONN_EXISTS 0x0B
```

ACL connection already exists

Definition at line 114 of file hci_defs.h.

2.1.2.36 HCI_ERR_CMD_DISALLOWED

```
#define HCI_ERR_CMD_DISALLOWED 0x0C
```

Command disallowed

Definition at line 115 of file hci_defs.h.

2.1.2.37 HCI_ERR_REJ_RESOURCES

```
#define HCI_ERR_REJ_RESOURCES 0x0D
```

Connection rejected limited resources

Definition at line 116 of file hci_defs.h.

2.1.2.38 HCI_ERR_REJ_SECURITY

```
#define HCI_ERR_REJ_SECURITY 0x0E
```

Connection rejected security reasons

Definition at line 117 of file hci_defs.h.

2.1.2.39 HCI_ERR_REJ_BD_ADDR

```
#define HCI_ERR_REJ_BD_ADDR 0x0F
```

Connection rejected unacceptable BD_ADDR

Definition at line 118 of file hci_defs.h.

2.1.2.40 HCI_ERR_ACCEPT_TIMEOUT

```
#define HCI_ERR_ACCEPT_TIMEOUT 0x10
```

Connection accept timeout exceeded

Definition at line 119 of file hci_defs.h.

2.1.2.41 HCI_ERR_UNSUP_FEAT

```
#define HCI_ERR_UNSUP_FEAT 0x11
```

Unsupported feature or parameter value

Definition at line 120 of file hci_defs.h.

2.1.2.42 HCI_ERR_INVALID_PARAM

```
#define HCI_ERR_INVALID_PARAM 0x12
```

Invalid HCI command parameters

Definition at line 121 of file hci_defs.h.

2.1.2.43 HCI_ERR_REMOTE_TERMINATED

```
#define HCI_ERR_REMOTE_TERMINATED 0x13
```

Remote user terminated connection

Definition at line 122 of file hci_defs.h.

2.1.2.44 HCI_ERR_REMOTE_RESOURCES

```
#define HCI_ERR_REMOTE_RESOURCES 0x14
```

Remote device low resources

Definition at line 123 of file hci_defs.h.

2.1.2.45 HCI_ERR_REMOTE_POWER_OFF

```
#define HCI_ERR_REMOTE_POWER_OFF 0x15
```

Remote device power off

Definition at line 124 of file hci_defs.h.

2.1.2.46 HCI_ERR_LOCAL_TERMINATED

```
#define HCI_ERR_LOCAL_TERMINATED 0x16
```

Connection terminated by local host

Definition at line 125 of file hci_defs.h.

2.1.2.47 HCI_ERR_REPEATED_ATTEMPTS

```
#define HCI_ERR_REPEATED_ATTEMPTS 0x17
```

Repeated attempts

Definition at line 126 of file hci_defs.h.

2.1.2.48 HCI_ERR_PAIRING_NOT_ALLOWED

```
#define HCI_ERR_PAIRING_NOT_ALLOWED 0x18
```

Pairing not allowed

Definition at line 127 of file hci_defs.h.

2.1.2.49 HCI_ERR_UNKNOWN_LMP_PDU

```
#define HCI_ERR_UNKNOWN_LMP_PDU 0x19
```

Unknown LMP PDU

Definition at line 128 of file hci_defs.h.

2.1.2.50 HCI_ERR_UNSUP_REMOTE_FEAT

```
#define HCI_ERR_UNSUP_REMOTE_FEAT 0x1A
```

Unsupported remote feature

Definition at line 129 of file hci_defs.h.

2.1.2.51 HCI_ERR_SCO_OFFSET

```
#define HCI_ERR_SCO_OFFSET 0x1B
```

SCO offset rejected

Definition at line 130 of file hci_defs.h.

2.1.2.52 HCI_ERR_SCO_INTERVAL

```
#define HCI_ERR_SCO_INTERVAL 0x1C
```

SCO interval rejected

Definition at line 131 of file hci_defs.h.

2.1.2.53 HCI_ERR_SCO_MODE

```
#define HCI_ERR_SCO_MODE 0x1D
```

SCO air mode rejected

Definition at line 132 of file hci_defs.h.

2.1.2.54 HCI_ERR_LMP_PARAM

```
#define HCI_ERR_LMP_PARAM 0x1E
```

Invalid LMP parameters

Definition at line 133 of file hci_defs.h.

2.1.2.55 HCI_ERR_UNSPECIFIED

```
#define HCI_ERR_UNSPECIFIED 0x1F
```

Unspecified error

Definition at line 134 of file hci_defs.h.

2.1.2.56 HCI_ERR_UNSUP_LMP_PARAM

```
#define HCI_ERR_UNSUP_LMP_PARAM 0x20
```

Unsupported LMP parameter value

Definition at line 135 of file hci_defs.h.

2.1.2.57 HCI_ERR_ROLE_CHANGE

```
#define HCI_ERR_ROLE_CHANGE 0x21
```

Role change not allowed

Definition at line 136 of file hci_defs.h.

2.1.2.58 HCI_ERR_LL_RESP_TIMEOUT

```
#define HCI_ERR_LL_RESP_TIMEOUT 0x22
```

LL response timeout

Definition at line 137 of file hci_defs.h.

2.1.2.59 HCI_ERR_LMP_COLLISION

```
#define HCI_ERR_LMP_COLLISION 0x23
```

LMP error transaction collision

Definition at line 138 of file hci_defs.h.

2.1.2.60 HCI_ERR_LMP_PDU

```
#define HCI_ERR_LMP_PDU 0x24
```

LMP pdu not allowed

Definition at line 139 of file hci_defs.h.

2.1.2.61 HCI_ERR_ENCRYPT_MODE

```
#define HCI_ERR_ENCRYPT_MODE 0x25
```

Encryption mode not acceptable

Definition at line 140 of file hci_defs.h.

2.1.2.62 HCI_ERR_LINK_KEY

```
#define HCI_ERR_LINK_KEY 0x26
```

Link key can not be changed

Definition at line 141 of file hci_defs.h.

2.1.2.63 HCI_ERR_UNSUP_QOS

```
#define HCI_ERR_UNSUP_QOS 0x27
```

Requested qos not supported

Definition at line 142 of file hci_defs.h.

2.1.2.64 HCI_ERR_INSTANT_PASSED

```
#define HCI_ERR_INSTANT_PASSED 0x28
```

Instant passed

Definition at line 143 of file hci_defs.h.

2.1.2.65 HCI_ERR_UNSUP_UNIT_KEY

```
#define HCI_ERR_UNSUP_UNIT_KEY 0x29
```

Pairing with unit key not supported

Definition at line 144 of file hci_defs.h.

2.1.2.66 HCI_ERR_TRANSACT_COLLISION

```
#define HCI_ERR_TRANSACT_COLLISION 0x2A
```

Different transaction collision

Definition at line 145 of file hci_defs.h.

2.1.2.67 HCI_ERR_CHANNEL_CLASS

```
#define HCI_ERR_CHANNEL_CLASS 0x2E
```

Channel classification not supported

Definition at line 146 of file hci_defs.h.

2.1.2.68 HCI_ERR_MEMORY

```
#define HCI_ERR_MEMORY 0x2F
```

Insufficient security

Definition at line 147 of file hci_defs.h.

2.1.2.69 HCI_ERR_PARAMETER_RANGE

```
#define HCI_ERR_PARAMETER_RANGE 0x30
```

Parameter out of mandatory range

Definition at line 148 of file hci_defs.h.

2.1.2.70 HCI_ERR_ROLE_SWITCH_PEND

```
#define HCI_ERR_ROLE_SWITCH_PEND 0x32
```

Role switch pending

Definition at line 149 of file hci_defs.h.

2.1.2.71 HCI_ERR_RESERVED_SLOT

```
#define HCI_ERR_RESERVED_SLOT 0x34
```

Reserved slot violation

Definition at line 150 of file hci_defs.h.

2.1.2.72 HCI_ERR_ROLE_SWITCH

```
#define HCI_ERR_ROLE_SWITCH 0x35
```

Role switch failed

Definition at line 151 of file hci_defs.h.

2.1.2.73 HCI_ERR_INQ_TOO_LARGE

```
#define HCI_ERR_INQ_TOO_LARGE 0x36
```

Extended inquiry response too large

Definition at line 152 of file hci_defs.h.

2.1.2.74 HCI_ERR_UNSUP_SSP

```
#define HCI_ERR_UNSUP_SSP 0x37
```

Secure simple pairing not supported by host

Definition at line 153 of file hci_defs.h.

2.1.2.75 HCI_ERR_HOST_BUSY_PAIRING

```
#define HCI_ERR_HOST_BUSY_PAIRING 0x38
```

Host busy - pairing

Definition at line 154 of file hci_defs.h.

2.1.2.76 HCI_ERR_NO_CHANNEL

```
#define HCI_ERR_NO_CHANNEL 0x39
```

Connection rejected no suitable channel

Definition at line 155 of file hci_defs.h.

2.1.2.77 HCI_ERR_CONTROLLER_BUSY

```
#define HCI_ERR_CONTROLLER_BUSY 0x3A
```

Controller busy

Definition at line 156 of file hci_defs.h.

2.1.2.78 HCI_ERR_CONN_INTERVAL

```
#define HCI_ERR_CONN_INTERVAL 0x3B
```

Unacceptable connection interval

Definition at line 157 of file hci_defs.h.

2.1.2.79 HCI_ERR_ADV_TIMEOUT

```
#define HCI_ERR_ADV_TIMEOUT 0x3C
```

Advertising timeout

Definition at line 158 of file hci_defs.h.

2.1.2.80 HCI_ERR_MIC_FAILURE

```
#define HCI_ERR_MIC_FAILURE 0x3D
```

Connection terminated due to MIC failure

Definition at line 159 of file hci_defs.h.

2.1.2.81 HCI_ERR_CONN_FAIL

```
#define HCI_ERR_CONN_FAIL 0x3E
```

Connection failed to be established

Definition at line 160 of file hci_defs.h.

2.1.2.82 HCI_ERR_MAC_CONN_FAIL

```
#define HCI_ERR_MAC_CONN_FAIL 0x3F
```

MAC connection failed

Definition at line 161 of file hci_defs.h.

2.1.2.83 HCI_ERR_COARSE_CLK_ADJ_REJ

```
#define HCI_ERR_COARSE_CLK_ADJ_REJ 0x40
```

Coarse clock adjustment rejected

Definition at line 162 of file hci_defs.h.

2.1.2.84 HCI_ERR_TYPE0_SUBMAP_NOT_DEF

```
#define HCI_ERR_TYPE0_SUBMAP_NOT_DEF 0x41
```

Type0 submap not defined

Definition at line 163 of file hci_defs.h.

2.1.2.85 HCI_ERR_UNKNOWN_ADV_ID

```
#define HCI_ERR_UNKNOWN_ADV_ID 0x42
```

Unknown advertising identifier

Definition at line 164 of file hci_defs.h.

2.1.2.86 HCI_ERR_LIMIT_REACHED

```
#define HCI_ERR_LIMIT_REACHED 0x43
```

Limit reached

Definition at line 165 of file hci_defs.h.

2.1.2.87 HCI_ERR_OP_CANCELLED_BY_HOST

```
#define HCI_ERR_OP_CANCELLED_BY_HOST 0x44
```

Operation cancelled by host

Definition at line 166 of file hci_defs.h.

2.1.2.88 HCI_ERR_PKT_TOO_LONG

```
#define HCI_ERR_PKT_TOO_LONG 0x45
```

Packet too long

Definition at line 168 of file hci_defs.h.

2.1.2.89 HCI_OGF_NOP

```
#define HCI_OGF_NOP 0x00
```

No operation

Definition at line 175 of file hci_defs.h.

2.1.2.90 HCI_OGF_LINK_CONTROL

```
#define HCI_OGF_LINK_CONTROL 0x01
```

Link control

Definition at line 176 of file hci_defs.h.

2.1.2.91 HCI_OGF_LINK_POLICY

```
#define HCI_OGF_LINK_POLICY 0x02
```

Link policy

Definition at line 177 of file hci_defs.h.

2.1.2.92 HCI_OGF_CONTROLLER

```
#define HCI_OGF_CONTROLLER 0x03
```

Controller and baseband

Definition at line 178 of file hci_defs.h.

2.1.2.93 HCI_OGF_INFORMATIONAL

```
#define HCI_OGF_INFORMATIONAL 0x04
```

Informational parameters

Definition at line 179 of file hci_defs.h.

2.1.2.94 HCI_OGF_STATUS

```
#define HCI_OGF_STATUS 0x05
```

Status parameters

Definition at line 180 of file hci_defs.h.

2.1.2.95 HCI_OGF_TESTING

```
#define HCI_OGF_TESTING 0x06
```

Testing

Definition at line 181 of file hci_defs.h.

2.1.2.96 HCI_OGF_LE_CONTROLLER

```
#define HCI_OGF_LE_CONTROLLER 0x08
```

LE controller

Definition at line 182 of file hci_defs.h.

2.1.2.97 HCI_OGF_VENDOR_SPEC

```
#define HCI_OGF_VENDOR_SPEC 0x3F
```

Vendor specific

Definition at line 183 of file hci_defs.h.

2.1.2.98 HCI_LEN_DISCONNECT_CMPL

```
#define HCI_LEN_DISCONNECT_CMPL 4
```

Disconnect event length.

Definition at line 768 of file hci_defs.h.

2.1.2.99 HCI_LEN_READ_REMOTE_VER_INFO_CMPL

```
#define HCI_LEN_READ_REMOTE_VER_INFO_CMPL 8
```

Read remove version info complete event length.

Definition at line 769 of file hci_defs.h.

2.1.2.100 HCI_LEN_CMD_CMPL

```
#define HCI_LEN_CMD_CMPL 3
```

Command complete event length.

Definition at line 770 of file hci_defs.h.

2.1.2.101 HCI_LEN_CMD_STATUS

```
#define HCI_LEN_CMD_STATUS 4
```

Command status event length.

Definition at line 771 of file hci_defs.h.

2.1.2.102 HCI_LEN_HW_ERR

```
#define HCI_LEN_HW_ERR 1
```

Hardware error event length.

Definition at line 772 of file hci_defs.h.

2.1.2.103 HCI_LEN_NUM_CMPL_PKTS

```
#define HCI_LEN_NUM_CMPL_PKTS(
              numHdls ) (1 + (4 * numHdls))
```

Number of completed packets event length.

Definition at line 773 of file hci_defs.h.

2.1.2.104 HCI_LEN_ENC_CHANGE

```
#define HCI_LEN_ENC_CHANGE 4
```

Encryption change event length.

Definition at line 774 of file hci_defs.h.

2.1.2.105 HCI_LEN_ENC_KEY_REFRESH_CMPL

```
#define HCI_LEN_ENC_KEY_REFRESH_CMPL 3
```

Encryption key refresh complete event length.

Definition at line 775 of file hci_defs.h.

2.1.2.106 HCI_LEN_LE_CONN_CMPL

```
#define HCI_LEN_LE_CONN_CMPL 19
```

Connection complete event length.

Definition at line 776 of file hci_defs.h.

2.1.2.107 HCI_LEN_LE_ADV_RPT_MIN

```
#define HCI_LEN_LE_ADV_RPT_MIN 12
```

Advertising report event minimum length.

Definition at line 777 of file hci_defs.h.

2.1.2.108 HCI_LEN_LE_CONN_UPDATE_CMPL

```
#define HCI_LEN_LE_CONN_UPDATE_CMPL 10
```

Connection update complete event length.

Definition at line 778 of file hci_defs.h.

2.1.2.109 HCI_LEN_LE_READ_REMOTE_FEAT_CMPL

```
#define HCI_LEN_LE_READ_REMOTE_FEAT_CMPL 12
```

Read remote feature event length.

Definition at line 779 of file hci_defs.h.

2.1.2.110 HCI_LEN_LE_LTK_REQ

```
#define HCI_LEN_LE_LTK_REQ 13
```

LTK request event length.

Definition at line 780 of file hci_defs.h.

2.1.2.111 HCI_LEN_LE_REM_CONN_PARAM_REQ

```
#define HCI_LEN_LE_REM_CONN_PARAM_REQ 11
```

Remote connection parameter event length.

Definition at line 782 of file hci_defs.h.

2.1.2.112 HCI_LEN_LE_DATA_LEN_CHANGE

```
#define HCI_LEN_LE_DATA_LEN_CHANGE 11
```

Data length change event length.

Definition at line 783 of file hci_defs.h.

2.1.2.113 HCI_LEN_LE_READ_PUB_KEY_CMPL

```
#define HCI_LEN_LE_READ_PUB_KEY_CMPL 66
```

Read local P256 public key compete event length.

Definition at line 784 of file hci_defs.h.

2.1.2.114 HCI_LEN_LE_GEN_DHKEY_CMPL

```
#define HCI_LEN_LE_GEN_DHKEY_CMPL 34
```

Generate DH key complete event length.

Definition at line 785 of file hci_defs.h.

2.1.2.115 HCI_LEN_LE_ENHANCED_CONN_CMPL

```
#define HCI_LEN_LE_ENHANCED_CONN_CMPL 31
```

Enhanced connection complete event length.

Definition at line 786 of file hci_defs.h.

2.1.2.116 HCI_LEN_LE_DIRECT_ADV_REPORT

```
#define HCI_LEN_LE_DIRECT_ADV_REPORT 18
```

Direct advertising report event length.

Definition at line 787 of file hci_defs.h.

2.1.2.117 HCI_LEN_AUTH_PAYLOAD_TIMEOUT

```
#define HCI_LEN_AUTH_PAYLOAD_TIMEOUT 2
```

Authenticated payload timeout event length.

Definition at line 788 of file hci_defs.h.

2.1.2.118 HCI_LEN_LE_PHY_UPDATE_CMPL [1/2]

```
#define HCI_LEN_LE_PHY_UPDATE_CMPL 6
```

PHY update complete event length.

Definition at line 792 of file hci_defs.h.

2.1.2.119 HCI_LEN_LE_PHY_UPDATE_CMPL [2/2]

```
#define HCI_LEN_LE_PHY_UPDATE_CMPL 6
```

PHY update complete event length.

Definition at line 792 of file hci_defs.h.

2.1.2.120 HCI_LEN_LE_CH_SEL_ALGO

```
#define HCI_LEN_LE_CH_SEL_ALGO 4
```

Channel selection algorithm event length.

Definition at line 791 of file hci_defs.h.

2.1.2.121 HCI_LEN_LE_EXT_ADV_REPORT_MIN

```
#define HCI_LEN_LE_EXT_ADV_REPORT_MIN 26
```

Extended advertising report minimum length.

Definition at line 793 of file hci_defs.h.

2.1.2.122 HCI_LEN_LE_PER_ADV_SYNC_EST

```
#define HCI_LEN_LE_PER_ADV_SYNC_EST 16
```

Periodic advertising sync established event length.

Definition at line 794 of file hci_defs.h.

2.1.2.123 HCI_LEN_LE_PER_ADV_REPORT

```
#define HCI_LEN_LE_PER_ADV_REPORT 8
```

Periodic advertising report event length.

Definition at line 795 of file hci_defs.h.

2.1.2.124 HCI_LEN_LE_PER_ADV_SYNC_LOST

```
#define HCI_LEN_LE_PER_ADV_SYNC_LOST 3
```

Periodic advertising sync lost event length.

Definition at line 796 of file hci_defs.h.

2.1.2.125 HCI_LEN_LE_SCAN_TIMEOUT

```
#define HCI_LEN_LE_SCAN_TIMEOUT 1
```

Scan timeout event length.

Definition at line 797 of file hci_defs.h.

2.1.2.126 HCI_LEN_LE_ADV_SET_TERM

```
#define HCI_LEN_LE_ADV_SET_TERM 6
```

Advertising set terminated event length.

Definition at line 798 of file hci_defs.h.

2.1.2.127 HCI_LEN_LE_SCAN_REQ_RCVD

```
#define HCI_LEN_LE_SCAN_REQ_RCVD 9
```

Scan request received event length.

Definition at line 799 of file hci_defs.h.

2.1.2.128 HCI_LEN_LE_PER_SYNC_TRSF_RCVT

```
#define HCI_LEN_LE_PER_SYNC_TRSF_RCVT 20
```

Periodic advertising sync transfer received event length.

Definition at line 801 of file hci_defs.h.

2.1.2.129 HCI_LEN_LE_CIS_EST

```
#define HCI_LEN_LE_CIS_EST 29
```

CIS established event length.

Definition at line 803 of file hci_defs.h.

2.1.2.130 HCI_LEN_LE_CIS_REQ

```
#define HCI_LEN_LE_CIS_REQ 7
```

CIS request event length.

Definition at line 804 of file hci_defs.h.

2.1.2.131 HCI_LEN_LE_PEER_SCA_CMPL

```
#define HCI_LEN_LE_PEER_SCA_CMPL 5
```

Request peer SCA complete event length.

Definition at line 805 of file hci_defs.h.

2.1.2.132 HCI_LEN_LE_CREATE_BIG_CMPL

```
#define HCI_LEN_LE_CREATE_BIG_CMPL(
             numBis ) (19 + (2 * numBis))
```

Create BIG complete event length.

Definition at line 806 of file hci_defs.h.

2.1.2.133 HCI_LEN_LE_TERMINATE_BIG_CMPL

```
#define HCI_LEN_LE_TERMINATE_BIG_CMPL 3
```

Terminate BIG complete event length.

Definition at line 807 of file hci_defs.h.

2.1.2.134 HCI_LEN_LE_BIG_SYNC_EST

```
#define HCI_LEN_LE_BIG_SYNC_EST(
              numBis ) (15 + (2 * numBis))
```

BIG sync established event length.

Definition at line 808 of file hci_defs.h.

2.1.2.135 HCI_LEN_LE_BIG_SYNC_LOST

```
#define HCI_LEN_LE_BIG_SYNC_LOST 3
```

BIG sync lost event length.

Definition at line 809 of file hci_defs.h.

2.1.2.136 HCI_LEN_LE_POWER_REPORT

```
#define HCI_LEN_LE_POWER_REPORT 9
```

Power reporting event length.

Definition at line 810 of file hci_defs.h.

2.1.2.137 HCI_LEN_LE_PATH_LOSS_ZONE

```
#define HCI_LEN_LE_PATH_LOSS_ZONE 5
```

Path loss reporting event length.

Definition at line 811 of file hci_defs.h.

2.1.2.138 HCI_LEN_LE_BIG_INFO_ADV_REPORT

```
#define HCI_LEN_LE_BIG_INFO_ADV_REPORT 20
```

BIG Info advertising report length.

Definition at line 812 of file hci_defs.h.

2.1.2.139 HCI_SUP_DISCONNECT

```
#define HCI_SUP_DISCONNECT 0x20
```

Byte 0

Definition at line 820 of file hci_defs.h.

2.1.2.140 HCI_SUP_READ_REMOTE_VER_INFO

```
#define HCI_SUP_READ_REMOTE_VER_INFO 0x80
```

Byte 2

Definition at line 821 of file hci_defs.h.

2.1.2.141 HCI_SUP_SET_EVENT_MASK

```
#define HCI_SUP_SET_EVENT_MASK 0x40
```

Byte 5

Definition at line 822 of file hci_defs.h.

2.1.2.142 HCI_SUP_RESET

```
#define HCI_SUP_RESET 0x80
```

Byte 5

Definition at line 823 of file hci_defs.h.

2.1.2.143 HCI_SUP_READ_TX_PWR_LVL

```
#define HCI_SUP_READ_TX_PWR_LVL 0x04
```

Byte 10

Definition at line 824 of file hci_defs.h.

2.1.2.144 HCI_SUP_READ_LOCAL_VER_INFO

```
#define HCI_SUP_READ_LOCAL_VER_INFO 0x08
```

Byte 14

Definition at line 825 of file hci_defs.h.

2.1.2.145 HCI_SUP_READ_LOCAL_SUP_FEAT

```
#define HCI_SUP_READ_LOCAL_SUP_FEAT 0x20
```

Byte 14

Definition at line 826 of file hci_defs.h.

2.1.2.146 HCI_SUP_READ_BD_ADDR

```
#define HCI_SUP_READ_BD_ADDR 0x02
```

Byte 15

Definition at line 827 of file hci_defs.h.

2.1.2.147 HCI_SUP_READ_RSSI

```
#define HCI_SUP_READ_RSSI 0x20
```

Byte 15

Definition at line 828 of file hci_defs.h.

2.1.2.148 HCI_SUP_SET_EVENT_MASK_PAGE2

```
#define HCI_SUP_SET_EVENT_MASK_PAGE2 0x04
```

Byte 22

Definition at line 829 of file hci_defs.h.

2.1.2.149 HCI_SUP_LE_SET_EVENT_MASK

```
#define HCI_SUP_LE_SET_EVENT_MASK 0x01
```

Byte 25

Definition at line 830 of file hci_defs.h.

2.1.2.150 HCI_SUP_LE_READ_BUF_SIZE

```
#define HCI_SUP_LE_READ_BUF_SIZE 0x02
```

Byte 25

Definition at line 831 of file hci_defs.h.

2.1.2.151 HCI_SUP_LE_READ_LOCAL_SUP_FEAT

```
#define HCI_SUP_LE_READ_LOCAL_SUP_FEAT 0x04
```

Byte 25

Definition at line 832 of file hci_defs.h.

2.1.2.152 HCI_SUP_LE_SET_RAND_ADDR

```
#define HCI_SUP_LE_SET_RAND_ADDR 0x10
```

Byte 25

Definition at line 833 of file hci_defs.h.

2.1.2.153 HCI_SUP_LE_SET_ADV_PARAM

```
#define HCI_SUP_LE_SET_ADV_PARAM 0x20
```

Byte 25

Definition at line 834 of file hci_defs.h.

2.1.2.154 HCI_SUP_LE_READ_ADV_TX_POWER

```
#define HCI_SUP_LE_READ_ADV_TX_POWER 0x40
```

Byte 25

Definition at line 835 of file hci_defs.h.

2.1.2.155 HCI_SUP_LE_SET_ADV_DATA

```
#define HCI_SUP_LE_SET_ADV_DATA 0x80
```

Byte 25

Definition at line 836 of file hci_defs.h.

2.1.2.156 HCI_SUP_LE_SET_SCAN_RESP_DATA

```
#define HCI_SUP_LE_SET_SCAN_RESP_DATA 0x01
```

Byte 26

Definition at line 837 of file hci_defs.h.

2.1.2.157 HCI_SUP_LE_SET_ADV_ENABLE

```
#define HCI_SUP_LE_SET_ADV_ENABLE 0x02
```

Byte 26

Definition at line 838 of file hci_defs.h.

2.1.2.158 HCI_SUP_LE_SET_SCAN_PARAM

```
#define HCI_SUP_LE_SET_SCAN_PARAM 0x04
```

Byte 26

Definition at line 839 of file hci_defs.h.

2.1.2.159 HCI_SUP_LE_SET_SCAN_ENABLE

```
#define HCI_SUP_LE_SET_SCAN_ENABLE 0x08
```

Byte 26

Definition at line 840 of file hci_defs.h.

2.1.2.160 HCI_SUP_LE_CREATE_CONN

```
#define HCI_SUP_LE_CREATE_CONN 0x10
```

Byte 26

Definition at line 841 of file hci_defs.h.

2.1.2.161 HCI_SUP_LE_CREATE_CONN_CANCEL

```
#define HCI_SUP_LE_CREATE_CONN_CANCEL 0x20
```

Byte 26

Definition at line 842 of file hci_defs.h.

2.1.2.162 HCI_SUP_LE_READ_WHITE_LIST_SIZE

```
#define HCI_SUP_LE_READ_WHITE_LIST_SIZE 0x40
```

Byte 26

Definition at line 843 of file hci_defs.h.

2.1.2.163 HCI_SUP_LE_CLEAR_WHITE_LIST

```
#define HCI_SUP_LE_CLEAR_WHITE_LIST 0x80
```

Byte 26

Definition at line 844 of file hci_defs.h.

2.1.2.164 HCI_SUP_LE_ADD_DEV_WHITE_LIST

```
#define HCI_SUP_LE_ADD_DEV_WHITE_LIST 0x01
```

Byte 27

Definition at line 845 of file hci_defs.h.

2.1.2.165 HCI_SUP_LE_REMOVE_DEV_WHITE_LIST

```
#define HCI_SUP_LE_REMOVE_DEV_WHITE_LIST 0x02
```

Byte 27

Definition at line 846 of file hci_defs.h.

2.1.2.166 HCI_SUP_LE_CONN_UPDATE

```
#define HCI_SUP_LE_CONN_UPDATE 0x04
```

Byte 27

Definition at line 847 of file hci_defs.h.

2.1.2.167 HCI_SUP_LE_SET_HOST_CHAN_CLASS

```
#define HCI_SUP_LE_SET_HOST_CHAN_CLASS 0x08
```

Byte 27

Definition at line 848 of file hci_defs.h.

2.1.2.168 HCI_SUP_LE_READ_CHAN_MAP

```
#define HCI_SUP_LE_READ_CHAN_MAP 0x10
```

Byte 27

Definition at line 849 of file hci_defs.h.

2.1.2.169 HCI_SUP_LE_READ_REMOTE_FEAT

```
#define HCI_SUP_LE_READ_REMOTE_FEAT 0x20
```

Byte 27

Definition at line 850 of file hci_defs.h.

2.1.2.170 HCI_SUP_LE_ENCRYPT

```
#define HCI_SUP_LE_ENCRYPT 0x40
```

Byte 27

Definition at line 851 of file hci_defs.h.

2.1.2.171 HCI_SUP_LE_RAND

```
#define HCI_SUP_LE_RAND 0x80
```

Byte 27

Definition at line 852 of file hci_defs.h.

2.1.2.172 HCI_SUP_LE_START_ENCRYPTION

```
#define HCI_SUP_LE_START_ENCRYPTION 0x01
```

Byte 28

Definition at line 853 of file hci_defs.h.

2.1.2.173 HCI_SUP_LE_LTK_REQ_REPL

```
#define HCI_SUP_LE_LTK_REQ_REPL 0x02
```

Byte 28

Definition at line 854 of file hci_defs.h.

2.1.2.174 HCI_SUP_LE_LTK_REQ_NEG_REPL

```
#define HCI_SUP_LE_LTK_REQ_NEG_REPL 0x04
```

Byte 28

Definition at line 855 of file hci_defs.h.

2.1.2.175 HCI_SUP_LE_READ_SUP_STATES

```
#define HCI_SUP_LE_READ_SUP_STATES 0x08
```

Byte 28

Definition at line 856 of file hci_defs.h.

2.1.2.176 HCI_SUP_LE_RECEIVER_TEST

```
#define HCI_SUP_LE_RECEIVER_TEST 0x10
```

Byte 28

Definition at line 857 of file hci_defs.h.

2.1.2.177 HCI_SUP_LE_TRANSMITTER_TEST

```
#define HCI_SUP_LE_TRANSMITTER_TEST 0x20
```

Byte 28

Definition at line 858 of file hci_defs.h.

2.1.2.178 HCI_SUP_LE_TEST_END

```
#define HCI_SUP_LE_TEST_END 0x40
```

Byte 28

Definition at line 859 of file hci_defs.h.

2.1.2.179 HCI_SUP_READ_AUTH_PAYLOAD_TO

```
#define HCI_SUP_READ_AUTH_PAYLOAD_TO 0x10
```

Byte 32

Definition at line 860 of file hci_defs.h.

2.1.2.180 HCI_SUP_WRITE_AUTH_PAYLOAD_TO

```
#define HCI_SUP_WRITE_AUTH_PAYLOAD_TO 0x20
```

Byte 32

Definition at line 861 of file hci_defs.h.

2.1.2.181 HCI_SUP_LE_REM_CONN_PARAM_REQ_REPL

```
#define HCI_SUP_LE_REM_CONN_PARAM_REQ_REPL 0x10
```

Byte 33

Definition at line 863 of file hci_defs.h.

2.1.2.182 HCI_SUP_LE_REM_CONN_PARAM_REQ_NEG_REPL

```
#define HCI_SUP_LE_REM_CONN_PARAM_REQ_NEG_REPL 0x20
```

Byte 33

Definition at line 864 of file hci_defs.h.

2.1.2.183 HCI_SUP_LE_SET_DATA_LEN

```
#define HCI_SUP_LE_SET_DATA_LEN 0x40
```

Byte 33

Definition at line 866 of file hci_defs.h.

2.1.2.184 HCI_SUP_LE_READ_DEF_DATA_LEN

```
#define HCI_SUP_LE_READ_DEF_DATA_LEN 0x80
```

Byte 33

Definition at line 867 of file hci_defs.h.

2.1.2.185 HCI_SUP_LE_WRITE_DEF_DATA_LEN

```
#define HCI_SUP_LE_WRITE_DEF_DATA_LEN 0x01
```

Byte 34

Definition at line 868 of file hci_defs.h.

2.1.2.186 HCI_SUP_LE_READ_LOCAL_P256_PUB_KEY

```
#define HCI_SUP_LE_READ_LOCAL_P256_PUB_KEY 0x02
```

Byte 34

Definition at line 869 of file hci_defs.h.

2.1.2.187 HCI_SUP_LE_GENERATE_DHKEY

```
#define HCI_SUP_LE_GENERATE_DHKEY 0x04
```

Byte 34

Definition at line 870 of file hci_defs.h.

2.1.2.188 HCI_SUP_LE_ADD_DEV_RES_LIST_EVT

```
#define HCI_SUP_LE_ADD_DEV_RES_LIST_EVT 0x08
```

Byte 34

Definition at line 871 of file hci_defs.h.

2.1.2.189 HCI_SUP_LE_REMOVE_DEV_RES_LIST

```
#define HCI_SUP_LE_REMOVE_DEV_RES_LIST 0x10
```

Byte 34

Definition at line 872 of file hci_defs.h.

2.1.2.190 HCI_SUP_LE_CLEAR_RES_LIST

```
#define HCI_SUP_LE_CLEAR_RES_LIST 0x20
```

Byte 34

Definition at line 873 of file hci_defs.h.

2.1.2.191 HCI_SUP_LE_READ_RES_LIST_SIZE

```
#define HCI_SUP_LE_READ_RES_LIST_SIZE 0x40
```

Byte 34

Definition at line 874 of file hci_defs.h.

2.1.2.192 HCI_SUP_LE_READ_PEER_RES_ADDR

```
#define HCI_SUP_LE_READ_PEER_RES_ADDR 0x80
```

Byte 34

Definition at line 875 of file hci_defs.h.

2.1.2.193 HCI_SUP_LE_READ_LOCAL_RES_ADDR

```
#define HCI_SUP_LE_READ_LOCAL_RES_ADDR 0x01
```

Byte 35

Definition at line 876 of file hci_defs.h.

2.1.2.194 HCI_SUP_LE_SET_ADDR_RES_ENABLE

```
#define HCI_SUP_LE_SET_ADDR_RES_ENABLE 0x02
```

Byte 35

Definition at line 877 of file hci_defs.h.

2.1.2.195 HCI_SUP_LE_SET_RES_PRIV_ADDR_TO

```
#define HCI_SUP_LE_SET_RES_PRIV_ADDR_TO 0x04
```

Byte 35

Definition at line 878 of file hci_defs.h.

2.1.2.196 HCI_SUP_LE_READ_MAX_DATA_LEN

```
#define HCI_SUP_LE_READ_MAX_DATA_LEN 0x08
```

Byte 35

Definition at line 879 of file hci_defs.h.

2.1.2.197 HCI_SUP_LE_READ_PHY

```
#define HCI_SUP_LE_READ_PHY 0x10
```

Byte 35

Definition at line 881 of file hci_defs.h.

2.1.2.198 HCI_SUP_LE_SET_DEF_PHY

```
#define HCI_SUP_LE_SET_DEF_PHY 0x20
```

Byte 35

Definition at line 882 of file hci_defs.h.

2.1.2.199 HCI_SUP_LE_SET_PHY

```
#define HCI_SUP_LE_SET_PHY 0x40
```

Byte 35

Definition at line 883 of file hci_defs.h.

2.1.2.200 HCI_SUP_LE_ENHANCED_RECEIVER_TEST

```
#define HCI_SUP_LE_ENHANCED_RECEIVER_TEST 0x80
```

Byte 35

Definition at line 884 of file hci_defs.h.

2.1.2.201 HCI_SUP_LE_ENHANCED_TRANSMITTER_TEST

```
#define HCI_SUP_LE_ENHANCED_TRANSMITTER_TEST 0x01
```

Byte 36

Definition at line 885 of file hci_defs.h.

2.1.2.202 HCI_SUP_LE_SET_ADV_SET_RAND_ADDR

```
#define HCI_SUP_LE_SET_ADV_SET_RAND_ADDR 0x02
```

Byte 36

Definition at line 886 of file hci_defs.h.

2.1.2.203 HCI_SUP_LE_SET_EXT_ADV_PARAM

```
#define HCI_SUP_LE_SET_EXT_ADV_PARAM 0x04
```

Byte 36

Definition at line 887 of file hci_defs.h.

2.1.2.204 HCI_SUP_LE_SET_EXT_ADV_DATA

```
#define HCI_SUP_LE_SET_EXT_ADV_DATA 0x08
```

Byte 36

Definition at line 888 of file hci_defs.h.

2.1.2.205 HCI_SUP_LE_SET_EXT_SCAN_RESP_DATA

```
#define HCI_SUP_LE_SET_EXT_SCAN_RESP_DATA 0x10
```

Byte 36

Definition at line 889 of file hci_defs.h.

2.1.2.206 HCI_SUP_LE_SET_EXT_ADV_ENABLE

```
#define HCI_SUP_LE_SET_EXT_ADV_ENABLE 0x20
```

Byte 36

Definition at line 890 of file hci_defs.h.

2.1.2.207 HCI_SUP_LE_READ_MAX_ADV_DATA_LEN

```
#define HCI_SUP_LE_READ_MAX_ADV_DATA_LEN 0x40
```

Byte 36

Definition at line 891 of file hci_defs.h.

2.1.2.208 HCI_SUP_LE_READ_NUM_OF_SUP_ADV_SETS

```
#define HCI_SUP_LE_READ_NUM_OF_SUP_ADV_SETS 0x80
```

Byte 36

Definition at line 892 of file hci_defs.h.

2.1.2.209 HCI_SUP_LE_REMOVE_ADV_SET

```
#define HCI_SUP_LE_REMOVE_ADV_SET 0x01
```

Byte 37

Definition at line 893 of file hci_defs.h.

2.1.2.210 HCI_SUP_LE_CLEAR_ADV_SETS

```
#define HCI_SUP_LE_CLEAR_ADV_SETS 0x02
```

Byte 37

Definition at line 894 of file hci_defs.h.

2.1.2.211 HCI_SUP_LE_SET_PER_ADV_PARAM

```
#define HCI_SUP_LE_SET_PER_ADV_PARAM 0x04
```

Byte 37

Definition at line 895 of file hci_defs.h.

2.1.2.212 HCI_SUP_LE_SET_PER_ADV_DATA

```
#define HCI_SUP_LE_SET_PER_ADV_DATA 0x08
```

Byte 37

Definition at line 896 of file hci_defs.h.

2.1.2.213 HCI_SUP_LE_SET_PER_ADV_ENABLE

```
#define HCI_SUP_LE_SET_PER_ADV_ENABLE 0x10
```

Byte 37

Definition at line 897 of file hci_defs.h.

2.1.2.214 HCI_SUP_LE_SET_EXT_SCAN_PARAM

```
#define HCI_SUP_LE_SET_EXT_SCAN_PARAM 0x20
```

Byte 37

Definition at line 898 of file hci_defs.h.

2.1.2.215 HCI_SUP_LE_SET_EXT_SCAN_ENABLE

```
#define HCI_SUP_LE_SET_EXT_SCAN_ENABLE 0x40
```

Byte 37

Definition at line 899 of file hci_defs.h.

2.1.2.216 HCI_SUP_LE_EXT_CREATE_CONN

```
#define HCI_SUP_LE_EXT_CREATE_CONN 0x80
```

Byte 37

Definition at line 900 of file hci_defs.h.

2.1.2.217 HCI_SUP_LE_PER_ADV_CREATE_SYNC

```
#define HCI_SUP_LE_PER_ADV_CREATE_SYNC 0x01
```

Byte 38

Definition at line 901 of file hci_defs.h.

2.1.2.218 HCI_SUP_LE_PER_ADV_CREATE_SYNC_CANCEL

```
#define HCI_SUP_LE_PER_ADV_CREATE_SYNC_CANCEL 0x02
```

Byte 38

Definition at line 902 of file hci_defs.h.

2.1.2.219 HCI_SUP_LE_PER_ADV_TERMINATE_SYNC

```
#define HCI_SUP_LE_PER_ADV_TERMINATE_SYNC 0x04
```

Byte 38

Definition at line 903 of file hci_defs.h.

2.1.2.220 HCI_SUP_LE_ADD_DEV_PER_ADV_LIST

```
#define HCI_SUP_LE_ADD_DEV_PER_ADV_LIST 0x08
```

Byte 38

Definition at line 904 of file hci_defs.h.

2.1.2.221 HCI_SUP_LE_REMOVE_DEV_PER_ADV_LIST

```
#define HCI_SUP_LE_REMOVE_DEV_PER_ADV_LIST 0x10
```

Byte 38

Definition at line 905 of file hci_defs.h.

2.1.2.222 HCI_SUP_LE_CLEAR_PER_ADV_LIST

```
#define HCI_SUP_LE_CLEAR_PER_ADV_LIST 0x20
```

Byte 38

Definition at line 906 of file hci_defs.h.

2.1.2.223 HCI_SUP_LE_READ_PER_ADV_LIST_SIZE

```
#define HCI_SUP_LE_READ_PER_ADV_LIST_SIZE 0x40
```

Byte 38

Definition at line 907 of file hci_defs.h.

2.1.2.224 HCI_SUP_LE_READ_TX_POWER

```
#define HCI_SUP_LE_READ_TX_POWER 0x80
```

Byte 38

Definition at line 908 of file hci_defs.h.

2.1.2.225 HCI_SUP_LE_READ_RF_PATH_COMP

```
#define HCI_SUP_LE_READ_RF_PATH_COMP 0x01
```

Byte 39

Definition at line 909 of file hci_defs.h.

2.1.2.226 HCI_SUP_LE_WRITE_RF_PATH_COMP

```
#define HCI_SUP_LE_WRITE_RF_PATH_COMP 0x02
```

Byte 39

Definition at line 910 of file hci_defs.h.

2.1.2.227 HCI_SUP_LE_SET_PRIVACY_MODE

```
#define HCI_SUP_LE_SET_PRIVACY_MODE 0x04
```

Byte 39

Definition at line 911 of file hci_defs.h.

2.1.2.228 HCI_SUP_LE_RECEIVER_TEST_V3

```
#define HCI_SUP_LE_RECEIVER_TEST_V3 0x08
```

Byte 39

Definition at line 913 of file hci_defs.h.

2.1.2.229 HCI_SUP_LE_TRANSMITTER_TEST_V3

```
#define HCI_SUP_LE_TRANSMITTER_TEST_V3 0x10
```

Byte 39

Definition at line 914 of file hci_defs.h.

2.1.2.230 HCI_SUP_LE_SET_CONNLESS_CTE_TX_PARAMS

```
#define HCI_SUP_LE_SET_CONNLESS_CTE_TX_PARAMS 0x20
```

Byte 39

Definition at line 915 of file hci_defs.h.

2.1.2.231 HCI_SUP_LE_SET_CONNLESS_CTE_TX_ENABLE

```
#define HCI_SUP_LE_SET_CONNLESS_CTE_TX_ENABLE 0x40
```

Byte 39

Definition at line 916 of file hci_defs.h.

2.1.2.232 HCI_SUP_LE_SET_CONNLESS_IQ_SAMP_ENABLE

```
#define HCI_SUP_LE_SET_CONNLESS_IQ_SAMP_ENABLE 0x80
```

Byte 39

Definition at line 917 of file hci_defs.h.

2.1.2.233 HCI_SUP_LE_SET_CONN_CTE_RX_PARAMS

```
#define HCI_SUP_LE_SET_CONN_CTE_RX_PARAMS 0x01
```

Byte 40

Definition at line 918 of file hci_defs.h.

2.1.2.234 HCI_SUP_LE_SET_CONN_CTE_TX_PARAMS

```
#define HCI_SUP_LE_SET_CONN_CTE_TX_PARAMS 0x02
```

Byte 40

Definition at line 919 of file hci_defs.h.

2.1.2.235 HCI_SUP_LE_CONN_CTE_REQ_ENABLE

```
#define HCI_SUP_LE_CONN_CTE_REQ_ENABLE 0x04
```

Byte 40

Definition at line 920 of file hci_defs.h.

2.1.2.236 HCI_SUP_LE_CONN_CTE_RSP_ENABLE

```
#define HCI_SUP_LE_CONN_CTE_RSP_ENABLE 0x08
```

Byte 40

Definition at line 921 of file hci_defs.h.

2.1.2.237 HCI_SUP_LE_READ_ANTENNA_INFO

```
#define HCI_SUP_LE_READ_ANTENNA_INFO 0x10
```

Byte 40

Definition at line 922 of file hci_defs.h.

2.1.2.238 HCI_SUP_LE_SET_PER_ADV_RCV_ENABLE

```
#define HCI_SUP_LE_SET_PER_ADV_RCV_ENABLE 0x20
```

Byte 40

Definition at line 923 of file hci_defs.h.

2.1.2.239 HCI_SUP_LE_PER_ADV_SYNC_TRANSFER

```
#define HCI_SUP_LE_PER_ADV_SYNC_TRANSFER 0x40
```

Byte 40

Definition at line 924 of file hci_defs.h.

2.1.2.240 HCI_SUP_LE_PER_ADV_SET_INFO_TRANSFER

```
#define HCI_SUP_LE_PER_ADV_SET_INFO_TRANSFER 0x80
```

Byte 40

Definition at line 925 of file hci_defs.h.

2.1.2.241 HCI_SUP_LE_SET_PAST_PARAM

```
#define HCI_SUP_LE_SET_PAST_PARAM 0x01
```

Byte 41

Definition at line 926 of file hci_defs.h.

2.1.2.242 HCI_SUP_LE_SET_DEFAULT_PAST_PARAM

```
#define HCI_SUP_LE_SET_DEFAULT_PAST_PARAM 0x02
```

Byte 41

Definition at line 927 of file hci_defs.h.

2.1.2.243 HCI_SUP_LE_GENERATE_DHKEY_V2

```
#define HCI_SUP_LE_GENERATE_DHKEY_V2 0x04
```

Byte 41

Definition at line 928 of file hci_defs.h.

2.1.2.244 HCI_SUP_LE_MODIFY_SLEEP_CLK_ACCURACY

```
#define HCI_SUP_LE_MODIFY_SLEEP_CLK_ACCURACY 0x10
```

Byte 41

Definition at line 929 of file hci_defs.h.

2.1.2.245 HCI_SUP_LE_READ_BUF_SIZE_V2

```
#define HCI_SUP_LE_READ_BUF_SIZE_V2 0x20
```

Byte 41

Definition at line 931 of file hci_defs.h.

2.1.2.246 HCI_SUP_LE_READ_ISO_TX_SYNC

```
#define HCI_SUP_LE_READ_ISO_TX_SYNC 0x40
```

Byte 41

Definition at line 932 of file hci_defs.h.

2.1.2.247 HCI_SUP_LE_SET_CIG_PARAM

```
#define HCI_SUP_LE_SET_CIG_PARAM 0x80
```

Byte 41

Definition at line 933 of file hci_defs.h.

2.1.2.248 HCI_SUP_LE_SET_CIG_PARAM_TEST

```
#define HCI_SUP_LE_SET_CIG_PARAM_TEST 0x01
```

Byte 42

Definition at line 934 of file hci_defs.h.

2.1.2.249 HCI_SUP_LE_CREATE_CIS

```
#define HCI_SUP_LE_CREATE_CIS 0x02
```

Byte 42

Definition at line 935 of file hci_defs.h.

2.1.2.250 HCI_SUP_LE_REMOVE_CIG

```
#define HCI_SUP_LE_REMOVE_CIG 0x04
```

Byte 42

Definition at line 936 of file hci_defs.h.

2.1.2.251 HCI_SUP_LE_ACCEPT_CIS_REQ

```
#define HCI_SUP_LE_ACCEPT_CIS_REQ 0x08
```

Byte 42

Definition at line 937 of file hci_defs.h.

2.1.2.252 HCI_SUP_LE_REJECT_CIS_REQ

```
#define HCI_SUP_LE_REJECT_CIS_REQ 0x10
```

Byte 42

Definition at line 938 of file hci_defs.h.

2.1.2.253 HCI_SUP_LE_CREATE_BIG

```
#define HCI_SUP_LE_CREATE_BIG 0x20
```

Byte 42

Definition at line 939 of file hci_defs.h.

2.1.2.254 HCI_SUP_LE_CREATE_BIG_TEST

```
#define HCI_SUP_LE_CREATE_BIG_TEST 0x40
```

Byte 42

Definition at line 940 of file hci_defs.h.

2.1.2.255 HCI_SUP_LE_TERMINATE_BIG

```
#define HCI_SUP_LE_TERMINATE_BIG 0x80
```

Byte 42

Definition at line 941 of file hci_defs.h.

2.1.2.256 HCI_SUP_LE_BIG_CREATE_SYNC

```
#define HCI_SUP_LE_BIG_CREATE_SYNC 0x01
```

Byte 43

Definition at line 942 of file hci_defs.h.

2.1.2.257 HCI_SUP_LE_BIG_TERMINATE_SYNC

```
#define HCI_SUP_LE_BIG_TERMINATE_SYNC 0x02
```

Byte 43

Definition at line 943 of file hci_defs.h.

2.1.2.258 HCI_SUP_LE_REQ_PEER_SCA

```
#define HCI_SUP_LE_REQ_PEER_SCA 0x04
```

Byte 43

Definition at line 944 of file hci_defs.h.

2.1.2.259 HCI_SUP_LE_SETUP_ISO_DATA_PATH

```
#define HCI_SUP_LE_SETUP_ISO_DATA_PATH 0x08
```

Byte 43

Definition at line 945 of file hci_defs.h.

2.1.2.260 HCI_SUP_LE_REMOVE_ISO_DATA_PATH

```
#define HCI_SUP_LE_REMOVE_ISO_DATA_PATH 0x10
```

Byte 43

Definition at line 946 of file hci_defs.h.

2.1.2.261 HCI_SUP_LE_ISO_TRANSMIT_TEST

```
#define HCI_SUP_LE_ISO_TRANSMIT_TEST 0x20
```

Byte 43

Definition at line 947 of file hci_defs.h.

2.1.2.262 HCI_SUP_LE_ISO_RECEIVE_TEST

```
#define HCI_SUP_LE_ISO_RECEIVE_TEST 0x40
```

Byte 43

Definition at line 948 of file hci_defs.h.

2.1.2.263 HCI_SUP_LE_ISO_READ_TEST_COUNTERS

```
#define HCI_SUP_LE_ISO_READ_TEST_COUNTERS 0x80
```

Byte 43

Definition at line 949 of file hci_defs.h.

2.1.2.264 HCI_SUP_LE_ISO_TEST_END

```
#define HCI_SUP_LE_ISO_TEST_END 0x01
```

Byte 44

Definition at line 950 of file hci_defs.h.

2.1.2.265 HCI_SUP_LE_SET_HOST_FEATURE

```
#define HCI_SUP_LE_SET_HOST_FEATURE 0x02
```

Byte 44

Definition at line 951 of file hci_defs.h.

2.1.2.266 HCI_SUP_LE_READ_ISO_LINK_QUALITY

```
#define HCI_SUP_LE_READ_ISO_LINK_QUALITY 0x04
```

Byte 44

Definition at line 952 of file hci_defs.h.

2.1.2.267 HCI_SUP_LE_ENH_READ_TX_POWER_LEVEL

```
#define HCI_SUP_LE_ENH_READ_TX_POWER_LEVEL 0x08
```

Byte 44

Definition at line 953 of file hci_defs.h.

2.1.2.268 HCI_SUP_LE_READ_REMOTE_TX_POWER_LEVEL

```
#define HCI_SUP_LE_READ_REMOTE_TX_POWER_LEVEL 0x01
```

Byte 44

Definition at line 954 of file hci_defs.h.

2.1.2.269 HCI_SUP_LE_SET_PATH_LOSS_REPORT_PARAM

```
#define HCI_SUP_LE_SET_PATH_LOSS_REPORT_PARAM 0x02
```

Byte 44

Definition at line 955 of file hci_defs.h.

2.1.2.270 HCI_SUP_LE_SET_PATH_LOSS_REPORT_ENABLE

```
#define HCI_SUP_LE_SET_PATH_LOSS_REPORT_ENABLE 0x04
```

Byte 44

Definition at line 956 of file hci_defs.h.

2.1.2.271 HCI_SUP_LE_SET_TX_POWER_REPORT_ENABLE

```
#define HCI_SUP_LE_SET_TX_POWER_REPORT_ENABLE 0x08
```

Byte 44

Definition at line 957 of file hci_defs.h.

2.1.2.272 HCI_SUP_LE_TRANSMITTER_TEST_V4

```
#define HCI_SUP_LE_TRANSMITTER_TEST_V4 0x01
```

Byte 45

Definition at line 958 of file hci_defs.h.

2.1.2.273 HCI_SUP_READ_LOCAL_SUP_CODECS_V2

```
#define HCI_SUP_READ_LOCAL_SUP_CODECS_V2 0x02
```

Byte 45

Definition at line 959 of file hci_defs.h.

2.1.2.274 HCI_SUP_READ_LOCAL_SUP_CODEC_CAP

```
#define HCI_SUP_READ_LOCAL_SUP_CODEC_CAP 0x04
```

Byte 45

Definition at line 960 of file hci_defs.h.

2.1.2.275 HCI_SUP_READ_LOCAL_SUP_CTR_DLY

```
#define HCI_SUP_READ_LOCAL_SUP_CTR_DLY 0x08
```

Byte 45

Definition at line 961 of file hci_defs.h.

2.1.2.276 HCI_SUP_CONFIG_DATA_PATH

```
#define HCI_SUP_CONFIG_DATA_PATH 0x10
```

Byte 45

Definition at line 962 of file hci_defs.h.

2.1.2.277 HCI_SUP_CMD_LEN

```
#define HCI_SUP_CMD_LEN 64
```

Byte length of support cmd field.

Definition at line 964 of file hci_defs.h.

2.1.2.278 HCI_EVT_MASK_DISCONNECT_CMPL

```
#define HCI_EVT_MASK_DISCONNECT_CMPL 0x10
```

Byte 0

Definition at line 972 of file hci_defs.h.

2.1.2.279 HCI_EVT_MASK_ENC_CHANGE

```
#define HCI_EVT_MASK_ENC_CHANGE 0x80
```

Byte 0

Definition at line 973 of file hci_defs.h.

2.1.2.280 HCI_EVT_MASK_READ_REMOTE_VER_INFO_CMPL

```
#define HCI_EVT_MASK_READ_REMOTE_VER_INFO_CMPL 0x08
```

Byte 1

Definition at line 974 of file hci_defs.h.

2.1.2.281 HCI_EVT_MASK_HW_ERROR

```
#define HCI_EVT_MASK_HW_ERROR 0x80
```

Byte 1

Definition at line 975 of file hci_defs.h.

2.1.2.282 HCI_EVT_MASK_DATA_BUF_OVERFLOW

```
#define HCI_EVT_MASK_DATA_BUF_OVERFLOW 0x02
```

Byte 3

Definition at line 976 of file hci_defs.h.

2.1.2.283 HCI_EVT_MASK_ENC_KEY_REFRESH_CMPL

```
#define HCI_EVT_MASK_ENC_KEY_REFRESH_CMPL 0x80
```

Byte 5

Definition at line 977 of file hci_defs.h.

2.1.2.284 HCI_EVT_MASK_LE_META

```
#define HCI_EVT_MASK_LE_META 0x20
```

Byte 7

Definition at line 978 of file hci_defs.h.

2.1.2.285 HCI_EVT_MASK_AUTH_PAYLOAD_TIMEOUT

```
#define HCI_EVT_MASK_AUTH_PAYLOAD_TIMEOUT 0x80
```

Byte 2

Definition at line 985 of file hci_defs.h.

2.1.2.286 HCI_EVT_MASK_LE_CONN_CMPL_EVT

```
#define HCI_EVT_MASK_LE_CONN_CMPL_EVT 0x01
```

Byte 0

Definition at line 992 of file hci_defs.h.

2.1.2.287 HCI_EVT_MASK_LE_ADV_REPORT_EVT

```
#define HCI_EVT_MASK_LE_ADV_REPORT_EVT 0x02
```

Byte 0

Definition at line 993 of file hci_defs.h.

2.1.2.288 HCI_EVT_MASK_LE_CONN_UPDATE_CMPL_EVT

```
#define HCI_EVT_MASK_LE_CONN_UPDATE_CMPL_EVT 0x04
```

Byte 0

Definition at line 994 of file hci_defs.h.

2.1.2.289 HCI_EVT_MASK_LE_READ_REMOTE_FEAT_CMPL_EVT

```
#define HCI_EVT_MASK_LE_READ_REMOTE_FEAT_CMPL_EVT 0x08
```

Byte 0

Definition at line 995 of file hci_defs.h.

2.1.2.290 HCI_EVT_MASK_LE_LTK_REQ_EVT

```
#define HCI_EVT_MASK_LE_LTK_REQ_EVT 0x10
```

Byte 0

Definition at line 996 of file hci_defs.h.

2.1.2.291 HCI_EVT_MASK_LE_REMOTE_CONN_PARAM_REQ_EVT

```
#define HCI_EVT_MASK_LE_REMOTE_CONN_PARAM_REQ_EVT 0x20
```

Byte 0

Definition at line 998 of file hci_defs.h.

2.1.2.292 HCI_EVT_MASK_LE_DATA_LEN_CHANGE_EVT

```
#define HCI_EVT_MASK_LE_DATA_LEN_CHANGE_EVT 0x40
```

Byte 0

Definition at line 1000 of file hci_defs.h.

2.1.2.293 HCI_EVT_MASK_LE_READ_LOCAL_P256_PUB_KEY_CMPL

```
#define HCI_EVT_MASK_LE_READ_LOCAL_P256_PUB_KEY_CMPL 0x80
```

Byte 0

Definition at line 1001 of file hci_defs.h.

2.1.2.294 HCI_EVT_MASK_LE_GENERATE_DHKEY_CMPL

```
#define HCI_EVT_MASK_LE_GENERATE_DHKEY_CMPL 0x01
```

Byte 1

Definition at line 1002 of file hci_defs.h.

2.1.2.295 HCI_EVT_MASK_LE_ENHANCED_CONN_CMPL_EVT

```
#define HCI_EVT_MASK_LE_ENHANCED_CONN_CMPL_EVT 0x02
```

Byte 1

Definition at line 1003 of file hci_defs.h.

2.1.2.296 HCI_EVT_MASK_LE_DIRECT_ADV_REPORT_EVT

```
#define HCI_EVT_MASK_LE_DIRECT_ADV_REPORT_EVT 0x04
```

Byte 1

Definition at line 1004 of file hci_defs.h.

2.1.2.297 HCI_EVT_MASK_LE_PHY_UPDATE_CMPL_EVT

```
#define HCI_EVT_MASK_LE_PHY_UPDATE_CMPL_EVT 0x08
```

Byte 1

Definition at line 1006 of file hci_defs.h.

2.1.2.298 HCI_EVT_MASK_LE_EXT_ADV_REPORT_EVT

```
#define HCI_EVT_MASK_LE_EXT_ADV_REPORT_EVT 0x10
```

Byte 1

Definition at line 1007 of file hci_defs.h.

2.1.2.299 HCI_EVT_MASK_LE_PER_ADV_SYNC_EST_EVT

```
#define HCI_EVT_MASK_LE_PER_ADV_SYNC_EST_EVT 0x20
```

Byte 1

Definition at line 1008 of file hci_defs.h.

2.1.2.300 HCI_EVT_MASK_LE_PER_ADV_REPORT_EVT

```
#define HCI_EVT_MASK_LE_PER_ADV_REPORT_EVT 0x40
```

Byte 1

Definition at line 1009 of file hci_defs.h.

2.1.2.301 HCI_EVT_MASK_LE_PER_ADV_SYNC_LOST_EVT

```
#define HCI_EVT_MASK_LE_PER_ADV_SYNC_LOST_EVT 0x80
```

Byte 1

Definition at line 1010 of file hci_defs.h.

2.1.2.302 HCI_EVT_MASK_LE_SCAN_TIMEOUT_EVT

```
#define HCI_EVT_MASK_LE_SCAN_TIMEOUT_EVT 0x01
```

Byte 2

Definition at line 1011 of file hci_defs.h.

2.1.2.303 HCI_EVT_MASK_LE_ADV_SET_TERM_EVT

```
#define HCI_EVT_MASK_LE_ADV_SET_TERM_EVT 0x02
```

Byte 2

Definition at line 1012 of file hci_defs.h.

2.1.2.304 HCI_EVT_MASK_LE_SCAN_REQ_RCVD_EVT

```
#define HCI_EVT_MASK_LE_SCAN_REQ_RCVD_EVT 0x04
```

Byte 2

Definition at line 1013 of file hci_defs.h.

2.1.2.305 HCI_EVT_MASK_LE_CH_SEL_ALGO_EVT

```
#define HCI_EVT_MASK_LE_CH_SEL_ALGO_EVT 0x08
```

Byte 2 (Bit 19)

Definition at line 1014 of file hci_defs.h.

2.1.2.306 HCI_EVT_MASK_LE_CONNLESS_IQ_REPORT_EVT

```
#define HCI_EVT_MASK_LE_CONNLESS_IQ_REPORT_EVT 0x10
```

Byte 2

Definition at line 1016 of file hci_defs.h.

2.1.2.307 HCI_EVT_MASK_LE_CONN_IQ_REPORT_EVT

```
#define HCI_EVT_MASK_LE_CONN_IQ_REPORT_EVT 0x20
```

Byte 2

Definition at line 1017 of file hci_defs.h.

2.1.2.308 HCI_EVT_MASK_LE_CTE_REQ_FAILED_EVT

```
#define HCI_EVT_MASK_LE_CTE_REQ_FAILED_EVT 0x40
```

Byte 2

Definition at line 1018 of file hci_defs.h.

2.1.2.309 HCI_EVT_MASK_LE_PER_SYNC_TRSF_RCVT_EVT

```
#define HCI_EVT_MASK_LE_PER_SYNC_TRSF_RCVT_EVT 0x80
```

Byte 2 (Bit 23)

Definition at line 1019 of file hci_defs.h.

2.1.2.310 HCI_EVT_MASK_LE_CIS_EST_EVT

```
#define HCI_EVT_MASK_LE_CIS_EST_EVT 0x01
```

Byte 3 (Bit 24)

Definition at line 1021 of file hci_defs.h.

2.1.2.311 HCI_EVT_MASK_LE_CIS_REQ_EVT

```
#define HCI_EVT_MASK_LE_CIS_REQ_EVT 0x02
```

Byte 3

Definition at line 1022 of file hci_defs.h.

2.1.2.312 HCI_EVT_MASK_LE_CREATE_BIG_CMPL_EVT

```
#define HCI_EVT_MASK_LE_CREATE_BIG_CMPL_EVT 0x04
```

Byte 3

Definition at line 1023 of file hci_defs.h.

2.1.2.313 HCI_EVT_MASK_LE_TERMINATE_BIG_CMPL_EVT

```
#define HCI_EVT_MASK_LE_TERMINATE_BIG_CMPL_EVT 0x08
```

Byte 3

Definition at line 1024 of file hci_defs.h.

2.1.2.314 HCI_EVT_MASK_LE_BIG_SYNC_EST_EVT

```
#define HCI_EVT_MASK_LE_BIG_SYNC_EST_EVT 0x10
```

Byte 3

Definition at line 1025 of file hci_defs.h.

2.1.2.315 HCI_EVT_MASK_LE_BIG_SYNC_LOST_EVT

```
#define HCI_EVT_MASK_LE_BIG_SYNC_LOST_EVT 0x20
```

Byte 3

Definition at line 1026 of file hci_defs.h.

2.1.2.316 HCI_EVT_MASK_LE_PEER_SCA_CMPL_EVT

```
#define HCI_EVT_MASK_LE_PEER_SCA_CMPL_EVT 0x40
```

Byte 3

Definition at line 1027 of file hci_defs.h.

2.1.2.317 HCI_EVT_MASK_LE_PATH_LOSS_REPORT_EVT

```
#define HCI_EVT_MASK_LE_PATH_LOSS_REPORT_EVT 0x80
```

Byte 3

Definition at line 1028 of file hci_defs.h.

2.1.2.318 HCI_EVT_MASK_LE_TX_POWER_REPORT_EVT

```
#define HCI_EVT_MASK_LE_TX_POWER_REPORT_EVT 0x01
```

Byte 4 (Bit 32)

Definition at line 1030 of file hci_defs.h.

2.1.2.319 HCI_EVT_MASK_LE_BIG_INFO_ADV_RPT_EVT

```
#define HCI_EVT_MASK_LE_BIG_INFO_ADV_RPT_EVT 0x02
```

Byte 4

Definition at line 1031 of file hci_defs.h.

2.1.2.320 HCI_LE_SUP_FEAT_ENCRYPTION

```
#define HCI_LE_SUP_FEAT_ENCRYPTION 0x0000000000000001
```

Encryption supported

Definition at line 1040 of file hci_defs.h.

2.1.2.321 HCI_LE_SUP_FEAT_CONN_PARAM_REQ_PROC

```
#define HCI_LE_SUP_FEAT_CONN_PARAM_REQ_PROC 0x0000000000000002
```

Connection Parameters Request Procedure supported

Definition at line 1042 of file hci_defs.h.

2.1.2.322 HCI_LE_SUP_FEAT_EXT_REJECT_IND

```
#define HCI_LE_SUP_FEAT_EXT_REJECT_IND 0x0000000000000004
```

Extended Reject Indication supported

Definition at line 1043 of file hci_defs.h.

2.1.2.323 HCI_LE_SUP_FEAT_SLV_INIT_FEAT_EXCH

```
#define HCI_LE_SUP_FEAT_SLV_INIT_FEAT_EXCH 0x0000000000000008
```

Slave-Initiated Features Exchange supported

Definition at line 1044 of file hci_defs.h.

2.1.2.324 HCI_LE_SUP_FEAT_LE_PING

```
#define HCI_LE_SUP_FEAT_LE_PING 0x0000000000000010
```

LE Ping supported

Definition at line 1045 of file hci_defs.h.

2.1.2.325 HCI_LE_SUP_FEAT_DATA_LEN_EXT

```
#define HCI_LE_SUP_FEAT_DATA_LEN_EXT 0x0000000000000020
```

Data Length Extension supported

Definition at line 1047 of file hci_defs.h.

2.1.2.326 HCI_LE_SUP_FEAT_PRIVACY

```
#define HCI_LE_SUP_FEAT_PRIVACY 0x0000000000000040
```

LL Privacy supported

Definition at line 1048 of file hci_defs.h.

2.1.2.327 HCI_LE_SUP_FEAT_EXT_SCAN_FILT_POLICY

```
#define HCI_LE_SUP_FEAT_EXT_SCAN_FILT_POLICY 0x0000000000000080
```

Extended Scan Filter Policy supported

Definition at line 1049 of file hci_defs.h.

2.1.2.328 HCI_LE_SUP_FEAT_LE_2M_PHY

```
#define HCI_LE_SUP_FEAT_LE_2M_PHY 0x0000000000000100
```

LE 2M PHY supported

Definition at line 1051 of file hci_defs.h.

2.1.2.329 HCI_LE_SUP_FEAT_STABLE_MOD_IDX_TRANSMITTER

```
#define HCI_LE_SUP_FEAT_STABLE_MOD_IDX_TRANSMITTER 0x0000000000000200
```

Stable Modulation Index - Transmitter supported

Definition at line 1052 of file hci_defs.h.

2.1.2.330 HCI_LE_SUP_FEAT_STABLE_MOD_IDX_RECEIVER

```
#define HCI_LE_SUP_FEAT_STABLE_MOD_IDX_RECEIVER 0x0000000000000400
```

Stable Modulation Index - Receiver supported

Definition at line 1053 of file hci_defs.h.

2.1.2.331 HCI_LE_SUP_FEAT_LE_CODED_PHY

```
#define HCI_LE_SUP_FEAT_LE_CODED_PHY 0x0000000000000800
```

LE Coded PHY supported

Definition at line 1054 of file hci_defs.h.

2.1.2.332 HCI_LE_SUP_FEAT_LE_EXT_ADV

```
#define HCI_LE_SUP_FEAT_LE_EXT_ADV 0x0000000000001000
```

LE Extended Advertising supported

Definition at line 1055 of file hci_defs.h.

2.1.2.333 HCI_LE_SUP_FEAT_LE_PER_ADV

```
#define HCI_LE_SUP_FEAT_LE_PER_ADV 0x0000000000002000
```

LE Periodic Advertising supported

Definition at line 1056 of file hci_defs.h.

2.1.2.334 HCI_LE_SUP_FEAT_CH_SEL_2

```
#define HCI_LE_SUP_FEAT_CH_SEL_2 0x0000000000004000
```

Channel Selection Algorithm #2 supported

Definition at line 1057 of file hci_defs.h.

2.1.2.335 HCI_LE_SUP_FEAT_LE_POWER_CLASS_1

```
#define HCI_LE_SUP_FEAT_LE_POWER_CLASS_1 0x0000000000008000
```

LE Power Class 1 supported

Definition at line 1058 of file hci_defs.h.

2.1.2.336 HCI_LE_SUP_FEAT_MIN_NUN_USED_CHAN

```
#define HCI_LE_SUP_FEAT_MIN_NUN_USED_CHAN 0x0000000000010000
```

Minimum Number of Used Channels Procedure supported

Definition at line 1059 of file hci_defs.h.

2.1.2.337 HCI_LE_SUP_FEAT_CONN_CTE_REQ

```
#define HCI_LE_SUP_FEAT_CONN_CTE_REQ 0x0000000000020000
```

Connection CTE Request supported

Definition at line 1061 of file hci_defs.h.

2.1.2.338 HCI_LE_SUP_FEAT_CONN_CTE_RSP

```
#define HCI_LE_SUP_FEAT_CONN_CTE_RSP 0x0000000000040000
```

Connection CTE Response supported

Definition at line 1062 of file hci_defs.h.

2.1.2.339 HCI_LE_SUP_FEAT_CONNLESS_CTE_TRANS

```
#define HCI_LE_SUP_FEAT_CONNLESS_CTE_TRANS 0x0000000000080000
```

Connectionless CTE Transmitter supported

Definition at line 1063 of file hci_defs.h.

2.1.2.340 HCI_LE_SUP_FEAT_CONNLESS_CTE_RECV

```
#define HCI_LE_SUP_FEAT_CONNLESS_CTE_RECV 0x0000000000100000
```

Connectionless CTE Receiver supported

Definition at line 1064 of file hci_defs.h.

2.1.2.341 HCI_LE_SUP_FEAT_ANTENNA_SWITCH_AOD

```
#define HCI_LE_SUP_FEAT_ANTENNA_SWITCH_AOD 0x0000000000200000
```

Anetenna Switching during CTE Transmission (AoD) supported

Definition at line 1065 of file hci_defs.h.

2.1.2.342 HCI_LE_SUP_FEAT_ANTENNA_SWITCH_AOA

```
#define HCI_LE_SUP_FEAT_ANTENNA_SWITCH_AOA 0x0000000000400000
```

Anetenna Switching during CTE Reception (AoA) supported

Definition at line 1066 of file hci_defs.h.

2.1.2.343 HCI_LE_SUP_FEAT_RECV_CTE

```
#define HCI_LE_SUP_FEAT_RECV_CTE 0x0000000000800000
```

Receive Constant Tone Extension supported

Definition at line 1067 of file hci_defs.h.

2.1.2.344 HCI_LE_SUP_FEAT_PAST_SENDER

```
#define HCI_LE_SUP_FEAT_PAST_SENDER 0x0000000001000000
```

Periodic Advertising Sync Transfer Sender supported

Definition at line 1068 of file hci_defs.h.

2.1.2.345 HCI_LE_SUP_FEAT_PAST_RECIPIENT

```
#define HCI_LE_SUP_FEAT_PAST_RECIPIENT 0x0000000002000000
```

Periodic Advertising Sync Transfer Recipient supported

Definition at line 1069 of file hci_defs.h.

2.1.2.346 HCI_LE_SUP_FEAT_SCA_UPDATE

```
#define HCI_LE_SUP_FEAT_SCA_UPDATE 0x0000000004000000
```

Sleep Clock Accuracy Update supported

Definition at line 1070 of file hci_defs.h.

2.1.2.347 HCI_LE_SUP_FEAT_REMOTE_PUB_KEY_VALIDATION

```
#define HCI_LE_SUP_FEAT_REMOTE_PUB_KEY_VALIDATION 0x0000000008000000
```

Remote Public Key Validation supported

Definition at line 1071 of file hci_defs.h.

2.1.2.348 HCI_LE_SUP_FEAT_CIS_MASTER

```
#define HCI_LE_SUP_FEAT_CIS_MASTER 0x0000000010000000
```

Connected Isochronous Master Role supported

Definition at line 1073 of file hci_defs.h.

2.1.2.349 HCI_LE_SUP_FEAT_CIS_SLAVE

```
#define HCI_LE_SUP_FEAT_CIS_SLAVE 0x0000000020000000
```

Connected Isochronous Slave Role supported

Definition at line 1074 of file hci_defs.h.

2.1.2.350 HCI_LE_SUP_FEAT_ISO_BROADCASTER

```
#define HCI_LE_SUP_FEAT_ISO_BROADCASTER 0x0000000040000000
```

Isochronous Broadcaster Role supported

Definition at line 1075 of file hci_defs.h.

2.1.2.351 HCI_LE_SUP_FEAT_ISO_SYNC_RECEIVER

```
#define HCI_LE_SUP_FEAT_ISO_SYNC_RECEIVER 0x0000000080000000
```

Isochronous Synchronized Receiver Role supported

Definition at line 1076 of file hci_defs.h.

2.1.2.352 HCI_LE_SUP_FEAT_ISO_HOST_SUPPORT

```
#define HCI_LE_SUP_FEAT_ISO_HOST_SUPPORT 0x0000000100000000
```

Host support for ISO Channels

Definition at line 1077 of file hci_defs.h.

2.1.2.353 HCI_LE_SUP_FEAT_POWER_CONTROL_REQUEST

```
#define HCI_LE_SUP_FEAT_POWER_CONTROL_REQUEST 0x0000000200000000
```

Power control requests supported

Definition at line 1078 of file hci_defs.h.

2.1.2.354 HCI_LE_SUP_FEAT_POWER_CHANGE_IND

```
#define HCI_LE_SUP_FEAT_POWER_CHANGE_IND 0x0000000400000000
```

Power control power change indication supported

Definition at line 1079 of file hci_defs.h.

2.1.2.355 HCI_LE_SUP_FEAT_PATH_LOSS_MONITOR

```
#define HCI_LE_SUP_FEAT_PATH_LOSS_MONITOR 0x0000000800000000
```

Path loss monitoring supported

Definition at line 1080 of file hci_defs.h.

2.1.2.356 HCI_LE_FEAT_BIT_ISO_HOST_SUPPORT

```
#define HCI_LE_FEAT_BIT_ISO_HOST_SUPPORT 32
```

Host support for ISO Channels

Definition at line 1087 of file hci_defs.h.

2.1.2.357 HCI_ADV_MIN_INTERVAL

```
#define HCI_ADV_MIN_INTERVAL 0x0020
```

Minimum advertising interval

Definition at line 1094 of file hci_defs.h.

2.1.2.358 HCI_ADV_MAX_INTERVAL

```
#define HCI_ADV_MAX_INTERVAL 0x4000
```

Maximum advertising interval

Definition at line 1095 of file hci_defs.h.

2.1.2.359 HCI_ADV_DIRECTED_MAX_DURATION

```
#define HCI_ADV_DIRECTED_MAX_DURATION 0x0500
```

Maximum high duty cycle connectable directed advertising duration

Definition at line 1096 of file hci_defs.h.

2.1.2.360 HCI_ADV_TYPE_CONN_UNDIRECT

```
#define HCI_ADV_TYPE_CONN_UNDIRECT 0x00
```

Connectable undirected advertising

Definition at line 1097 of file hci_defs.h.

2.1.2.361 HCI_ADV_TYPE_CONN_DIRECT

```
#define HCI_ADV_TYPE_CONN_DIRECT 0x01
```

Connectable directed high duty cycle advertising

Definition at line 1098 of file hci_defs.h.

2.1.2.362 HCI_ADV_TYPE_DISC_UNDIRECT

```
#define HCI_ADV_TYPE_DISC_UNDIRECT 0x02
```

Discoverable undirected advertising

Definition at line 1099 of file hci_defs.h.

2.1.2.363 HCI_ADV_TYPE_NONCONN_UNDIRECT

```
#define HCI_ADV_TYPE_NONCONN_UNDIRECT 0x03
```

Nonconnectable undirected advertising

Definition at line 1100 of file hci_defs.h.

2.1.2.364 HCI_ADV_TYPE_CONN_DIRECT_LO_DUTY

```
#define HCI_ADV_TYPE_CONN_DIRECT_LO_DUTY 0x04
```

Connectable directed low duty cycle advertising

Definition at line 1101 of file hci_defs.h.

2.1.2.365 HCI_ADV_CHAN_37

```
#define HCI_ADV_CHAN_37 0x01
```

Advertising channel 37

Definition at line 1102 of file hci_defs.h.

2.1.2.366 HCI_ADV_CHAN_38

```
#define HCI_ADV_CHAN_38 0x02
```

Advertising channel 38

Definition at line 1103 of file hci_defs.h.

2.1.2.367 HCI_ADV_CHAN_39

```
#define HCI_ADV_CHAN_39 0x04
```

Advertising channel 39

Definition at line 1104 of file hci_defs.h.

2.1.2.368 HCI_ADV_FILT_NONE

```
#define HCI_ADV_FILT_NONE 0x00
```

No scan request or connection filtering

Definition at line 1105 of file hci_defs.h.

2.1.2.369 HCI_ADV_FILT_SCAN

```
#define HCI_ADV_FILT_SCAN 0x01
```

White list filters scan requests

Definition at line 1106 of file hci_defs.h.

2.1.2.370 HCI_ADV_FILT_CONN

```
#define HCI_ADV_FILT_CONN 0x02
```

White list filters connections

Definition at line 1107 of file hci_defs.h.

2.1.2.371 HCI_ADV_FILT_ALL

```
#define HCI_ADV_FILT_ALL 0x03
```

White list filters scan req. and conn.

Definition at line 1108 of file hci_defs.h.

2.1.2.372 HCI_SCAN_TYPE_PASSIVE

```
#define HCI_SCAN_TYPE_PASSIVE 0
```

Passive scan

Definition at line 1115 of file hci_defs.h.

2.1.2.373 HCI_SCAN_TYPE_ACTIVE

```
#define HCI_SCAN_TYPE_ACTIVE 1
```

Active scan

Definition at line 1116 of file hci_defs.h.

2.1.2.374 HCI_SCAN_INTERVAL_MIN

```
#define HCI_SCAN_INTERVAL_MIN 0x0004
```

Minimum scan interval

Definition at line 1117 of file hci_defs.h.

2.1.2.375 HCI_SCAN_INTERVAL_MAX

```
#define HCI_SCAN_INTERVAL_MAX 0x4000
```

Maximum scan interval

Definition at line 1118 of file hci_defs.h.

2.1.2.376 HCI_SCAN_INTERVAL_DEFAULT

```
#define HCI_SCAN_INTERVAL_DEFAULT 0x0010
```

Default scan interval

Definition at line 1119 of file hci_defs.h.

2.1.2.377 HCI_SCAN_WINDOW_MIN

```
#define HCI_SCAN_WINDOW_MIN 0x0004
```

Minimum scan window

Definition at line 1120 of file hci_defs.h.

2.1.2.378 HCI_SCAN_WINDOW_MAX

```
#define HCI_SCAN_WINDOW_MAX 0x4000
```

Maximum scan window

Definition at line 1121 of file hci_defs.h.

2.1.2.379 HCI_SCAN_WINDOW_DEFAULT

```
#define HCI_SCAN_WINDOW_DEFAULT 0x0010
```

Default scan window

Definition at line 1122 of file hci_defs.h.

2.1.2.380 HCI_CONN_INTERVAL_MIN

```
#define HCI_CONN_INTERVAL_MIN 0x0006
```

Minimum connection interval

Definition at line 1129 of file hci_defs.h.

2.1.2.381 HCI_CONN_INTERVAL_MAX

```
#define HCI_CONN_INTERVAL_MAX 0x0C80
```

Maximum connection interval

Definition at line 1130 of file hci_defs.h.

2.1.2.382 HCI_CONN_LATENCY_MAX

```
#define HCI_CONN_LATENCY_MAX 0x01F3
```

Maximum connection latency

Definition at line 1131 of file hci_defs.h.

2.1.2.383 HCI_SUP_TIMEOUT_MIN

```
#define HCI_SUP_TIMEOUT_MIN 0x000A
```

Minimum supervision timeout

Definition at line 1132 of file hci_defs.h.

2.1.2.384 HCI_SUP_TIMEOUT_MAX

```
#define HCI_SUP_TIMEOUT_MAX 0x0C80
```

Maximum supervision timeout

Definition at line 1133 of file hci_defs.h.

2.1.2.385 HCI_ROLE_MASTER [1/2]

```
#define HCI_ROLE_MASTER 0
```

Role is master

Definition at line 1393 of file hci_defs.h.

2.1.2.386 HCI_ROLE_MASTER [2/2]

```
#define HCI_ROLE_MASTER 0
```

Role is master

Definition at line 1393 of file hci_defs.h.

2.1.2.387 HCI_ROLE_SLAVE [1/2]

```
#define HCI_ROLE_SLAVE 1
```

Role is slave

Definition at line 1394 of file hci_defs.h.

2.1.2.388 HCI_ROLE_SLAVE [2/2]

```
#define HCI_ROLE_SLAVE 1
```

Role is slave

Definition at line 1394 of file hci_defs.h.

2.1.2.389 HCI_CLOCK_500PPM

```
#define HCI_CLOCK_500PPM 0x00
```

500 ppm clock accuracy

Definition at line 1142 of file hci_defs.h.

2.1.2.390 HCI_CLOCK_250PPM

```
#define HCI_CLOCK_250PPM 0x01
```

250 ppm clock accuracy

Definition at line 1143 of file hci_defs.h.

2.1.2.391 HCI_CLOCK_150PPM

```
#define HCI_CLOCK_150PPM 0x02
```

150 ppm clock accuracy

Definition at line 1144 of file hci_defs.h.

2.1.2.392 HCI_CLOCK_100PPM

```
#define HCI_CLOCK_100PPM 0x03
```

100 ppm clock accuracy

Definition at line 1145 of file hci_defs.h.

2.1.2.393 HCI_CLOCK_75PPM

```
#define HCI_CLOCK_75PPM 0x04
```

75 ppm clock accuracy

Definition at line 1146 of file hci_defs.h.

2.1.2.394 HCI_CLOCK_50PPM

```
#define HCI_CLOCK_50PPM 0x05
```

50 ppm clock accuracy

Definition at line 1147 of file hci_defs.h.

2.1.2.395 HCI_CLOCK_30PPM

```
#define HCI_CLOCK_30PPM 0x06
```

30 ppm clock accuracy

Definition at line 1148 of file hci_defs.h.

2.1.2.396 HCI_CLOCK_20PPM

```
#define HCI_CLOCK_20PPM 0x07
```

20 ppm clock accuracy

Definition at line 1149 of file hci_defs.h.

2.1.2.397 HCI_ADV_CONN_UNDIRECT

```
#define HCI_ADV_CONN_UNDIRECT 0x00
```

Connectable undirected advertising

Definition at line 1156 of file hci_defs.h.

2.1.2.398 HCI_ADV_CONN_DIRECT

```
#define HCI_ADV_CONN_DIRECT 0x01
```

Connectable directed advertising

Definition at line 1157 of file hci_defs.h.

2.1.2.399 HCI_ADV_DISC_UNDIRECT

```
#define HCI_ADV_DISC_UNDIRECT 0x02
```

Discoverable undirected advertising

Definition at line 1158 of file hci_defs.h.

2.1.2.400 HCI_ADV_NONCONN_UNDIRECT

```
#define HCI_ADV_NONCONN_UNDIRECT 0x03
```

Non-connectable undirected advertising

Definition at line 1159 of file hci_defs.h.

2.1.2.401 HCI_ADV_SCAN_RESPONSE

```
#define HCI_ADV_SCAN_RESPONSE 0x04
```

Scan response

Definition at line 1160 of file hci_defs.h.

2.1.2.402 HCI_ADV_DATA_OP_FRAG_INTER

```
#define HCI_ADV_DATA_OP_FRAG_INTER 0x00
```

Intermediate fragment

Definition at line 1167 of file hci_defs.h.

2.1.2.403 HCI_ADV_DATA_OP_FRAG_FIRST

```
#define HCI_ADV_DATA_OP_FRAG_FIRST 0x01
```

First fragment

Definition at line 1168 of file hci_defs.h.

2.1.2.404 HCI_ADV_DATA_OP_FRAG_LAST

```
#define HCI_ADV_DATA_OP_FRAG_LAST 0x02
```

Last fragment

Definition at line 1169 of file hci_defs.h.

2.1.2.405 HCI_ADV_DATA_OP_COMP_FRAG

```
#define HCI_ADV_DATA_OP_COMP_FRAG 0x03
```

Complete extended advertising data

Definition at line 1170 of file hci_defs.h.

2.1.2.406 HCI_ADV_DATA_OP_UNCHANGED_DATA

```
#define HCI_ADV_DATA_OP_UNCHANGED_DATA 0x04
```

Unchanged data (just update Advertising DID)

Definition at line 1171 of file hci_defs.h.

2.1.2.407 HCI_ADV_DATA_FRAG_PREF_FRAG

```
#define HCI_ADV_DATA_FRAG_PREF_FRAG 0x00
```

Controller may fragment all Host advertising data

Definition at line 1178 of file hci_defs.h.

2.1.2.408 HCI_ADV_DATA_FRAG_PREF_NO_FRAG

```
#define HCI_ADV_DATA_FRAG_PREF_NO_FRAG 0x01
```

Controller should not fragment nor minimize fragmentation of Host advertising data

Definition at line 1179 of file hci_defs.h.

2.1.2.409 HCI_ADV_NUM_SETS_ALL_DISABLE

```
#define HCI_ADV_NUM_SETS_ALL_DISABLE 0x00
```

Disable all advertising sets

Definition at line 1186 of file hci_defs.h.

2.1.2.410 HCI_MAX_NUM_PHYS

```
#define HCI_MAX_NUM_PHYS 3
```

Maximum number of scanning or initiating PHYs

Definition at line 1193 of file hci_defs.h.

2.1.2.411 HCI_ADV_PHY_LE_1M

```
#define HCI_ADV_PHY_LE_1M 0x01
```

LE 1M PHY

Definition at line 1200 of file hci_defs.h.

2.1.2.412 HCI_ADV_PHY_LE_2M

```
#define HCI_ADV_PHY_LE_2M 0x02
```

LE 2M PHY

Definition at line 1201 of file hci_defs.h.

2.1.2.413 HCI_ADV_PHY_LE_CODED

```
#define HCI_ADV_PHY_LE_CODED 0x03
```

LE Coded PHY

Definition at line 1202 of file hci_defs.h.

2.1.2.414 HCI_SCAN_PHY_LE_1M_BIT

```
#define HCI_SCAN_PHY_LE_1M_BIT (1<<0)
```

LE 1M PHY

Definition at line 1209 of file hci_defs.h.

2.1.2.415 HCI_SCAN_PHY_LE_2M_BIT

```
#define HCI_SCAN_PHY_LE_2M_BIT (1<<1)
```

LE 2M PHY

Definition at line 1210 of file hci_defs.h.

2.1.2.416 HCI_SCAN_PHY_LE_CODED_BIT

```
#define HCI_SCAN_PHY_LE_CODED_BIT (1<<2)
```

LE Coded PHY

Definition at line 1211 of file hci_defs.h.

2.1.2.417 HCI_INIT_PHY_LE_1M_BIT

```
#define HCI_INIT_PHY_LE_1M_BIT (1<<0)
```

LE 1M PHY

Definition at line 1218 of file hci_defs.h.

2.1.2.418 HCI_INIT_PHY_LE_2M_BIT

```
#define HCI_INIT_PHY_LE_2M_BIT (1<<1)
```

LE 2M PHY

Definition at line 1219 of file hci_defs.h.

2.1.2.419 HCI_INIT_PHY_LE_CODED_BIT

```
#define HCI_INIT_PHY_LE_CODED_BIT (1<<2)
```

LE Coded PHY

Definition at line 1220 of file hci_defs.h.

2.1.2.420 HCI_TRANS_PHY_LE_1M_BIT

```
#define HCI_TRANS_PHY_LE_1M_BIT (1<<0)
```

LE 1M PHY

Definition at line 1227 of file hci_defs.h.

2.1.2.421 HCI_TRANS_PHY_LE_2M_BIT

```
#define HCI_TRANS_PHY_LE_2M_BIT (1<<1)
```

LE 2M PHY

Definition at line 1228 of file hci_defs.h.

2.1.2.422 HCI_TRABS_PHY_LE_CODED_BIT

```
#define HCI_TRABS_PHY_LE_CODED_BIT (1<<2)
```

LE Coded PHY

Definition at line 1229 of file hci_defs.h.

2.1.2.423 HCI_ADV_PROP_CONN_ADV_BIT

```
#define HCI_ADV_PROP_CONN_ADV_BIT (1<<0)
```

Connectable advertising bit

Definition at line 1236 of file hci_defs.h.

2.1.2.424 HCI_ADV_PROP_SCAN_ADV_BIT

```
#define HCI_ADV_PROP_SCAN_ADV_BIT (1<<1)
```

Scannable advertising bit

Definition at line 1237 of file hci_defs.h.

2.1.2.425 HCI_ADV_PROP_DIRECT_ADV_BIT

```
#define HCI_ADV_PROP_DIRECT_ADV_BIT (1<<2)
```

Directed advertising bit

Definition at line 1238 of file hci_defs.h.

2.1.2.426 HCI_ADV_PROP_CONN_DIRECT_ADV_BIT

```
#define HCI_ADV_PROP_CONN_DIRECT_ADV_BIT (1<<3)
```

High duty cycle connectable directed advertising bit

Definition at line 1239 of file hci_defs.h.

2.1.2.427 HCI_ADV_PROP_USE_LEG_PDU_BIT

```
#define HCI_ADV_PROP_USE_LEG_PDU_BIT (1<<4)
```

Use legacy advertising PDUs bit

Definition at line 1240 of file hci_defs.h.

2.1.2.428 HCI_ADV_PROP_OMIT_ADV_ADDR_BIT

```
#define HCI_ADV_PROP_OMIT_ADV_ADDR_BIT (1<<5)
```

Omit advertiser's address from all PDUs (anonymous advertising) bit

Definition at line 1241 of file hci_defs.h.

2.1.2.429 HCI_ADV_PROP_INC_TX_PWR_BIT

```
#define HCI_ADV_PROP_INC_TX_PWR_BIT (1<<6)
```

Include TxPower in extended header of advertising PDU bit

Definition at line 1242 of file hci_defs.h.

2.1.2.430 HCI_ADV_PROP_LEG_CONN_UNDIRECT

```
#define HCI_ADV_PROP_LEG_CONN_UNDIRECT 0x13
```

Connectable and scannable undirected advertising (00010011b)

Definition at line 1249 of file hci_defs.h.

2.1.2.431 HCI_ADV_PROP_LEG_CONN_DIRECT

```
#define HCI_ADV_PROP_LEG_CONN_DIRECT 0x1D
```

Connectable directed high duty cycle advertising (00011101b)

Definition at line 1250 of file hci_defs.h.

2.1.2.432 HCI_ADV_PROP_LEG_SCAN_UNDIRECT

```
#define HCI_ADV_PROP_LEG_SCAN_UNDIRECT 0x12
```

Scannable undirected advertising (00010010b)

Definition at line 1251 of file hci_defs.h.

2.1.2.433 HCI_ADV_PROP_LEG_NONCONN_UNDIRECT

```
#define HCI_ADV_PROP_LEG_NONCONN_UNDIRECT 0x10
```

Non-connectable and non-scannable undirected advertising (00010000b)

Definition at line 1252 of file hci_defs.h.

2.1.2.434 HCI_ADV_PROP_LEG_CONN_DIRECT_LO_DUTY

```
#define HCI_ADV_PROP_LEG_CONN_DIRECT_LO_DUTY 0x15
```

Connectable directed low duty cycle advertising (00010101b)

Definition at line 1253 of file hci_defs.h.

2.1.2.435 HCI_ADV_RPT_CONN_ADV_BIT

```
#define HCI_ADV_RPT_CONN_ADV_BIT (1<<0)
```

Connectable advertising event bit

Definition at line 1260 of file hci_defs.h.

2.1.2.436 HCI_ADV_RPT_SCAN_ADV_BIT

```
#define HCI_ADV_RPT_SCAN_ADV_BIT (1<<1)
```

Scannable advertising event bit

Definition at line 1261 of file hci_defs.h.

2.1.2.437 HCI_ADV_RPT_DIRECT_ADV_BIT

```
#define HCI_ADV_RPT_DIRECT_ADV_BIT (1<<2)
```

Directed advertising event bit

Definition at line 1262 of file hci_defs.h.

2.1.2.438 HCI_ADV_RPT_SCAN_RSP_BIT

```
#define HCI_ADV_RPT_SCAN_RSP_BIT (1<<3)
```

Scan response event bit

Definition at line 1263 of file hci_defs.h.

2.1.2.439 HCI_ADV_RPT_LEG_ADV_BIT

```
#define HCI_ADV_RPT_LEG_ADV_BIT (1<<4)
```

Legacy advertising PDU event bit

Definition at line 1264 of file hci_defs.h.

2.1.2.440 HCI_ADV_RPT_DATA_STATUS_BITS

```
#define HCI_ADV_RPT_DATA_STATUS_BITS (3<<5)
```

Data status bits

Definition at line 1265 of file hci_defs.h.

2.1.2.441 HCI_ADV_RPT_LEG_CONN_UNDIRECT

```
#define HCI_ADV_RPT_LEG_CONN_UNDIRECT 0x13
```

Connectable and scannable undirected advertising (0010011b)

Definition at line 1272 of file hci_defs.h.

2.1.2.442 HCI_ADV_RPT_LEG_CONN_DIRECT

```
#define HCI_ADV_RPT_LEG_CONN_DIRECT 0x15
```

Connectable directed advertising (0010101b)

Definition at line 1273 of file hci_defs.h.

2.1.2.443 HCI_ADV_RPT_LEG_SCAN_UNDIRECT

```
#define HCI_ADV_RPT_LEG_SCAN_UNDIRECT 0x12
```

Scannable undirected advertising (0010010b)

Definition at line 1274 of file hci_defs.h.

2.1.2.444 HCI_ADV_RPT_LEG_NONCONN_UNDIRECT

```
#define HCI_ADV_RPT_LEG_NONCONN_UNDIRECT 0x10
```

Non-connectable and non-scannable undirected advertising (0010000b)

Definition at line 1275 of file hci_defs.h.

2.1.2.445 HCI_ADV_RPT_LEG_CONN_UNDIRECT_SCAN_RSP

```
#define HCI_ADV_RPT_LEG_CONN_UNDIRECT_SCAN_RSP 0x1B
```

Scan response to connectable and scannable undirected advertising (0011011b)

Definition at line 1276 of file hci_defs.h.

2.1.2.446 HCI_ADV_RPT_LEG_SCAN_UNDIRECT_SCAN_RSP

```
#define HCI_ADV_RPT_LEG_SCAN_UNDIRECT_SCAN_RSP 0x1A
```

Scan response to scannable undirected advertising (0011010b)

Definition at line 1277 of file hci_defs.h.

2.1.2.447 HCI_ADV_RPT_DATA_CMPL

```
#define HCI_ADV_RPT_DATA_CMPL 0x00
```

Data complete

Definition at line 1284 of file hci_defs.h.

2.1.2.448 HCI_ADV_RPT_DATA_INCMPL_MORE

```
#define HCI_ADV_RPT_DATA_INCMPL_MORE 0x01
```

Data incomplete, more date to come

Definition at line 1285 of file hci_defs.h.

2.1.2.449 HCI_ADV_RPT_DATA_INCMPL_TRUNC

```
#define HCI_ADV_RPT_DATA_INCMPL_TRUNC 0x02
```

Data incomplete, data truncated, no more date to come

Definition at line 1286 of file hci_defs.h.

2.1.2.450 HCI_ADV_RPT_PHY_PRIM_LE_1M

```
#define HCI_ADV_RPT_PHY_PRIM_LE_1M 0x01
```

Advertiser PHY is LE 1M

Definition at line 1293 of file hci_defs.h.

2.1.2.451 HCI_ADV_RPT_PHY_PRIM_LE_CODED

```
#define HCI_ADV_RPT_PHY_PRIM_LE_CODED 0x03
```

Advertiser PHY is LE Coded

Definition at line 1294 of file hci_defs.h.

2.1.2.452 HCI_ADV_RPT_PHY_SEC_NONE

```
#define HCI_ADV_RPT_PHY_SEC_NONE 0x00
```

No packets on seconday advertising channel

Definition at line 1301 of file hci_defs.h.

2.1.2.453 HCI_ADV_RPT_PHY_SEC_LE_1M

```
#define HCI_ADV_RPT_PHY_SEC_LE_1M 0x01
```

Advertiser PHY is LE 1M

Definition at line 1302 of file hci_defs.h.

2.1.2.454 HCI_ADV_RPT_PHY_SEC_LE_2M

```
#define HCI_ADV_RPT_PHY_SEC_LE_2M 0x02
```

Advertiser PHY is LE 2M

Definition at line 1303 of file hci_defs.h.

2.1.2.455 HCI_ADV_RPT_PHY_SEC_LE_CODED

```
#define HCI_ADV_RPT_PHY_SEC_LE_CODED 0x03
```

Advertiser PHY is LE Coded

Definition at line 1304 of file hci_defs.h.

2.1.2.456 HCI_CH_SEL_ALGO_1

```
#define HCI_CH_SEL_ALGO_1 0x00
```

LE channel selection algorithm #1 used

Definition at line 1311 of file hci_defs.h.

2.1.2.457 HCI_CH_SEL_ALGO_2

```
#define HCI_CH_SEL_ALGO_2 0x01
```

LE channel selection algorithm #2 used

Definition at line 1312 of file hci_defs.h.

2.1.2.458 HCI_PRIVATE_KEY_GENERATED

```
#define HCI_PRIVATE_KEY_GENERATED 0x00
```

Use generated private key

Definition at line 1319 of file hci_defs.h.

2.1.2.459 HCI_PRIVATE_KEY_DEBUG

```
#define HCI_PRIVATE_KEY_DEBUG 0x01
```

Use debug private key

Definition at line 1320 of file hci_defs.h.

2.1.2.460 HCI_MIN_NUM_OF_USED_CHAN

```
#define HCI_MIN_NUM_OF_USED_CHAN 8
```

Minimum number of used channels

Definition at line 1328 of file hci_defs.h.

2.1.2.461 HCI_SYNC_MIN_TIMEOUT

```
#define HCI_SYNC_MIN_TIMEOUT 0x000A
```

Minimum synchronization timeout

Definition at line 1335 of file hci_defs.h.

2.1.2.462 HCI_SYNC_MAX_TIMEOUT

```
#define HCI_SYNC_MAX_TIMEOUT 0x4000
```

Maximum synchronization timeout

Definition at line 1336 of file hci_defs.h.

2.1.2.463 HCI_SYNC_MAX_SKIP

```
#define HCI_SYNC_MAX_SKIP 0x01F3
```

Maximum synchronization skip

Definition at line 1343 of file hci_defs.h.

2.1.2.464 HCI_SYNC_MAX_HANDLE

```
#define HCI_SYNC_MAX_HANDLE 0x0EFF
```

Maximum synchronization handle

Definition at line 1350 of file hci_defs.h.

2.1.2.465 HCI_SYNC_TRSF_MODE_OFF

```
#define HCI_SYNC_TRSF_MODE_OFF 0x00
```

Periodic sync transfer receive is disabled

Definition at line 1357 of file hci_defs.h.

2.1.2.466 HCI_SYNC_TRSF_MODE_REP_DISABLED

```
#define HCI_SYNC_TRSF_MODE_REP_DISABLED 0x01,
```

Periodic sync transfer receive is enabled, report event is disabled

Definition at line 1358 of file hci_defs.h.

2.1.2.467 HCI_SYNC_TRSF_MODE_REP_ENABLED

```
#define HCI_SYNC_TRSF_MODE_REP_ENABLED 0x02,
```

Periodic sync transfer receive is enabled, report event is enabled

Definition at line 1359 of file hci_defs.h.

2.1.2.468 HCI_OPTIONS_FILT_POLICY_BIT

```
#define HCI_OPTIONS_FILT_POLICY_BIT (1<<0)
```

filter policy bit

Definition at line 1366 of file hci_defs.h.

2.1.2.469 HCI_OPTIONS_INIT_RPT_ENABLE_BIT

```
#define HCI_OPTIONS_INIT_RPT_ENABLE_BIT (1<<1)
```

initial periodic advertisement reporting bit

Definition at line 1367 of file hci_defs.h.

2.1.2.470 HCI_READ_TX_PWR_CURRENT

```
#define HCI_READ_TX_PWR_CURRENT 0
```

Read current tx power

Definition at line 1374 of file hci_defs.h.

2.1.2.471 HCI_READ_TX_PWR_MAX

```
#define HCI_READ_TX_PWR_MAX 1
```

Read maximum tx power

Definition at line 1375 of file hci_defs.h.

2.1.2.472 HCI_TX_PWR_MIN

```
#define HCI_TX_PWR_MIN -30
```

Minimum tx power dBm

Definition at line 1376 of file hci_defs.h.

2.1.2.473 HCI_TX_PWR_MAX

```
#define HCI_TX_PWR_MAX 20
```

Maximum tx power dBm

Definition at line 1377 of file hci_defs.h.

2.1.2.474 HCI_TX_PWR_NO_PREFERENCE

```
#define HCI_TX_PWR_NO_PREFERENCE 127
```

Tx power no preference

Definition at line 1378 of file hci_defs.h.

2.1.2.475 HCI_VERSION

```
#define HCI_VERSION 6
```

HCI specification version

Definition at line 1379 of file hci_defs.h.

2.1.2.476 HCI_RSSI_MIN

```
#define HCI_RSSI_MIN -127
```

Minimum RSSI dBm

Definition at line 1380 of file hci_defs.h.

2.1.2.477 HCI_RSSI_MAX

```
#define HCI_RSSI_MAX 20
```

Maximum RSSI dBm

Definition at line 1381 of file hci_defs.h.

2.1.2.478 HCI_ADDR_TYPE_PUBLIC

```
#define HCI_ADDR_TYPE_PUBLIC 0
```

Public device address

Definition at line 1382 of file hci_defs.h.

2.1.2.479 HCI_ADDR_TYPE_RANDOM

```
#define HCI_ADDR_TYPE_RANDOM 1
```

Random device address

Definition at line 1383 of file hci_defs.h.

2.1.2.480 HCI_ADDR_TYPE_PUBLIC_IDENTITY

```
#define HCI_ADDR_TYPE_PUBLIC_IDENTITY 2
```

Public identity address

Definition at line 1384 of file hci_defs.h.

2.1.2.481 HCI_ADDR_TYPE_RANDOM_IDENTITY

```
#define HCI_ADDR_TYPE_RANDOM_IDENTITY 3
```

Random identity address

Definition at line 1385 of file hci_defs.h.

2.1.2.482 HCI_ADDR_TYPE_ANONYMOUS

```
#define HCI_ADDR_TYPE_ANONYMOUS 0xFF
```

Anonymous device address

Definition at line 1386 of file hci_defs.h.

2.1.2.483 HCI_FILT_NONE

```
#define HCI_FILT_NONE 0
```

Accept all advertising packets

Definition at line 1387 of file hci_defs.h.

2.1.2.484 HCI_FILT_WHITE_LIST

```
#define HCI_FILT_WHITE_LIST 1
```

Accept from While List only

Definition at line 1388 of file hci_defs.h.

2.1.2.485 HCI_FILT_RES_INIT

```
#define HCI_FILT_RES_INIT 2
```

Accept directed advertisements with RPAs

Definition at line 1389 of file hci_defs.h.

2.1.2.486 HCI_FILT_WHITE_LIST_RES_INIT

```
#define HCI_FILT_WHITE_LIST_RES_INIT 3
```

Accept from White List or directed advertisements with RPAs

Definition at line 1390 of file hci_defs.h.

2.1.2.487 HCI_FILT_PER_ADV_PARAM

```
#define HCI_FILT_PER_ADV_PARAM 0
```

Listen to advertiser specified by create sync command parameters

Definition at line 1391 of file hci_defs.h.

2.1.2.488 HCI_FILT_PER_ADV_LIST

```
#define HCI_FILT_PER_ADV_LIST 1
```

Listen to advertiser from Periodic Advertiser List only

Definition at line 1392 of file hci_defs.h.

2.1.2.489 HCI_PRIV_MODE_NETWORK

```
#define HCI_PRIV_MODE_NETWORK 0x00
```

Network privacy mode (default)

Definition at line 1395 of file hci_defs.h.

2.1.2.490 HCI_PRIV_MODE_DEVICE

```
#define HCI_PRIV_MODE_DEVICE 0x01
```

Device privacy mode

Definition at line 1396 of file hci_defs.h.

2.1.2.491 HCI_PHY_NONE

```
#define HCI_PHY_NONE 0x00
```

No selected PHY

Definition at line 1403 of file hci_defs.h.

2.1.2.492 HCI_PHY_LE_1M_BIT

```
#define HCI_PHY_LE_1M_BIT (1<<0)
```

LE 1M PHY

Definition at line 1404 of file hci_defs.h.

2.1.2.493 HCI_PHY_LE_2M_BIT

```
#define HCI_PHY_LE_2M_BIT (1<<1)
```

LE 2M PHY

Definition at line 1405 of file hci_defs.h.

2.1.2.494 HCI_PHY_LE_CODED_BIT

```
#define HCI_PHY_LE_CODED_BIT (1<<2)
```

LE Coded PHY

Definition at line 1406 of file hci_defs.h.

2.1.2.495 HCI_ALL_PHY_ALL_PREFERENCES

```
#define HCI_ALL_PHY_ALL_PREFERENCES 0x00
```

All PHY preferences

Definition at line 1413 of file hci_defs.h.

2.1.2.496 HCI_ALL_PHY_TX_PREFERENCE_BIT

```
#define HCI_ALL_PHY_TX_PREFERENCE_BIT (1<<0)
```

Tx PHY preference

Definition at line 1414 of file hci_defs.h.

2.1.2.497 HCI_ALL_PHY_RX_PREFERENCE_BIT

```
#define HCI_ALL_PHY_RX_PREFERENCE_BIT (1<<1)
```

Rx PHY preference

Definition at line 1415 of file hci_defs.h.

2.1.2.498 HCI_PHY_OPTIONS_NONE

```
#define HCI_PHY_OPTIONS_NONE 0x00
```

No preferences

Definition at line 1422 of file hci_defs.h.

2.1.2.499 HCI_PHY_OPTIONS_S2_PREFERRED

```
#define HCI_PHY_OPTIONS_S2_PREFERRED 0x01
```

S=2 coding preferred when transmitting on LE Coded PHY

Definition at line 1423 of file hci_defs.h.

2.1.2.500 HCI_PHY_OPTIONS_S8_PREFERRED

```
#define HCI_PHY_OPTIONS_S8_PREFERRED 0x02
```

S=8 coding preferred when transmitting on LE Coded PHY

Definition at line 1424 of file hci_defs.h.

2.1.2.501 HCI_CTE_SLOT_DURATION_NONE

```
#define HCI_CTE_SLOT_DURATION_NONE 0x00
```

No switching and sampling

Definition at line 1431 of file hci_defs.h.

2.1.2.502 HCI_CTE_SLOT_DURATION_1_US

```
#define HCI_CTE_SLOT_DURATION_1_US 0x01
```

Switching and sampling slots are 1 us each

Definition at line 1432 of file hci_defs.h.

2.1.2.503 HCI_CTE_SLOT_DURATION_2_US

```
#define HCI_CTE_SLOT_DURATION_2_US 0x02
```

Switching and sampling slots are 2 us each

Definition at line 1433 of file hci_defs.h.

2.1.2.504 HCI_CTE_TYPE_PERMIT_AOA_RSP_BIT

```
#define HCI_CTE_TYPE_PERMIT_AOA_RSP_BIT (1<<0)
```

Allow AoA Constant Tone Extension Response

Definition at line 1440 of file hci_defs.h.

2.1.2.505 HCI_CTE_TYPE_PERMIT_AOD_RSP_1_US_BIT

```
#define HCI_CTE_TYPE_PERMIT_AOD_RSP_1_US_BIT (1<<1)
```

Allow AoD Constant Tone Extension Response with 1 us slots

Definition at line 1441 of file hci_defs.h.

2.1.2.506 HCI_CTE_TYPE_PERMIT_AOD_RSP_2_US_BIT

```
#define HCI_CTE_TYPE_PERMIT_AOD_RSP_2_US_BIT (1<<2)
```

Allow AoD Constant Tone Extension Response with 2 us slots

Definition at line 1442 of file hci_defs.h.

2.1.2.507 HCI_CTE_TYPE_REQ_AOA

```
#define HCI_CTE_TYPE_REQ_AOA 0x00
```

AoA Constant Tone Extension

Definition at line 1449 of file hci_defs.h.

2.1.2.508 HCI_CTE_TYPE_REQ_AOD_1_US

```
#define HCI_CTE_TYPE_REQ_AOD_1_US 0x01
```

AoD Constant Tone Extension with 1 us slots

Definition at line 1450 of file hci_defs.h.

2.1.2.509 HCI_CTE_TYPE_REQ_AOD_2_US

```
#define HCI_CTE_TYPE_REQ_AOD_2_US 0x02
```

AoD Constant Tone Extension with 2 us slots

Definition at line 1451 of file hci_defs.h.

2.1.2.510 HCI_VER_BT_CORE_SPEC_4_0

```
#define HCI_VER_BT_CORE_SPEC_4_0 0x06
```

Bluetooth core specification 4.0

Definition at line 1458 of file hci_defs.h.

2.1.2.511 HCI_VER_BT_CORE_SPEC_4_1

```
#define HCI_VER_BT_CORE_SPEC_4_1 0x07
```

Bluetooth core specification 4.1

Definition at line 1459 of file hci_defs.h.

2.1.2.512 HCI_VER_BT_CORE_SPEC_4_2

```
#define HCI_VER_BT_CORE_SPEC_4_2 0x08
```

Bluetooth core specification 4.2

Definition at line 1460 of file hci_defs.h.

2.1.2.513 HCI_VER_BT_CORE_SPEC_5_0

```
#define HCI_VER_BT_CORE_SPEC_5_0 0x09
```

Bluetooth core specification 5.0

Definition at line 1461 of file hci_defs.h.

2.1.2.514 HCI_VER_BT_CORE_SPEC_5_1

```
#define HCI_VER_BT_CORE_SPEC_5_1 0x0A
```

Bluetooth core specification 5.1

Definition at line 1462 of file hci_defs.h.

2.1.2.515 HCI_VER_BT_CORE_SPEC_5_2

```
#define HCI_VER_BT_CORE_SPEC_5_2 0x0B
```

Bluetooth core specification 5.2

Definition at line 1463 of file hci_defs.h.

2.1.2.516 HCI_EVT_MASK_LEN

```
#define HCI_EVT_MASK_LEN 8
```

Length of event mask byte array

Definition at line 1470 of file hci_defs.h.

2.1.2.517 HCI_EVT_MASK_PAGE_2_LEN

```
#define HCI_EVT_MASK_PAGE_2_LEN 8
```

Length of event mask page 2 byte array

Definition at line 1471 of file hci_defs.h.

2.1.2.518 HCI_LE_EVT_MASK_LEN

```
#define HCI_LE_EVT_MASK_LEN 8
```

Length of LE event mask byte array

Definition at line 1472 of file hci_defs.h.

2.1.2.519 HCI_FEAT_LEN

```
#define HCI_FEAT_LEN 8
```

Length of features byte array

Definition at line 1473 of file hci_defs.h.

2.1.2.520 HCI_ADV_DATA_LEN

```
#define HCI_ADV_DATA_LEN 31
```

Length of advertising data

Definition at line 1474 of file hci_defs.h.

2.1.2.521 HCI_SCAN_DATA_LEN

```
#define HCI_SCAN_DATA_LEN 31
```

Length of scan response data

Definition at line 1475 of file hci_defs.h.

2.1.2.522 HCI_EXT_ADV_DATA_LEN

```
#define HCI_EXT_ADV_DATA_LEN 251
```

Length of extended advertising data

Definition at line 1476 of file hci_defs.h.

2.1.2.523 HCI_EXT_ADV_CONN_DATA_LEN

```
#define HCI_EXT_ADV_CONN_DATA_LEN 191
```

Length of extended connectable advertising data

Definition at line 1477 of file hci_defs.h.

2.1.2.524 HCI_PER_ADV_DATA_LEN

```
#define HCI_PER_ADV_DATA_LEN 252
```

Length of periodic advertising data

Definition at line 1478 of file hci_defs.h.

2.1.2.525 HCI_EXT_ADV_RPT_DATA_LEN

```
#define HCI_EXT_ADV_RPT_DATA_LEN 229
```

Length of extended advertising report data

Definition at line 1479 of file hci_defs.h.

2.1.2.526 HCI_PER_ADV_RPT_DATA_LEN

```
#define HCI_PER_ADV_RPT_DATA_LEN 247
```

Length of periodic advertising report data

Definition at line 1480 of file hci_defs.h.

2.1.2.527 HCI_CHAN_MAP_LEN

```
#define HCI_CHAN_MAP_LEN 5
```

Length of channel map byte array

Definition at line 1481 of file hci_defs.h.

2.1.2.528 HCI_KEY_LEN

```
#define HCI_KEY_LEN 16
```

Length of encryption key

Definition at line 1482 of file hci_defs.h.

2.1.2.529 HCI_ENCRYPT_DATA_LEN

```
#define HCI_ENCRYPT_DATA_LEN 16
```

Length of data used in encryption

Definition at line 1483 of file hci_defs.h.

2.1.2.530 HCI_RAND_LEN

```
#define HCI_RAND_LEN 8
```

Length of random number

Definition at line 1484 of file hci_defs.h.

2.1.2.531 HCI_LE_STATES_LEN

```
#define HCI_LE_STATES_LEN 8
```

Length of LE states byte array

Definition at line 1485 of file hci_defs.h.

2.1.2.532 HCI_P256_KEY_LEN

```
#define HCI_P256_KEY_LEN 64
```

Length of P256 key

Definition at line 1486 of file hci_defs.h.

2.1.2.533 HCI_DH_KEY_LEN

```
#define HCI_DH_KEY_LEN 32
```

Length of DH Key

Definition at line 1487 of file hci_defs.h.

2.1.2.534 HCI_BC_LEN

```
#define HCI_BC_LEN 16
```

Broadcast code length

Definition at line 1488 of file hci_defs.h.

2.1.2.535 HCI_EXT_ADV_RPT_DATA_LEN_OFFSET

```
#define HCI_EXT_ADV_RPT_DATA_LEN_OFFSET 23
```

Length field offset of extended advertising report data

Definition at line 1490 of file hci_defs.h.

2.1.2.536 HCI_PER_ADV_RPT_DATA_LEN_OFFSET

```
#define HCI_PER_ADV_RPT_DATA_LEN_OFFSET 6
```

Length field offset of periodic advertising report data

Definition at line 1491 of file hci_defs.h.

2.1.2.537 HCI_MIN_NUM_ANTENNA_IDS

```
#define HCI_MIN_NUM_ANTENNA_IDS 2
```

Minimum number of Antenna IDs in switching pattern

Definition at line 1498 of file hci_defs.h.

2.1.2.538 HCI_MAX_NUM_ANTENNA_IDS

```
#define HCI_MAX_NUM_ANTENNA_IDS 75
```

Maximum number of Antenna IDs in switching pattern

Definition at line 1499 of file hci_defs.h.

2.1.2.539 HCI_IQ_RPT_SAMPLE_CNT_MIN

```
#define HCI_IQ_RPT_SAMPLE_CNT_MIN 9
```

Minimum number of sample pairs in IQ report

Definition at line 1506 of file hci_defs.h.

2.1.2.540 HCI_IQ_RPT_SAMPLE_CNT_MAX

```
#define HCI_IQ_RPT_SAMPLE_CNT_MAX 82
```

Maximum number of sample pairs in IQ report

Definition at line 1507 of file hci_defs.h.

2.1.2.541 HCI_CONN_IQ_RPT_SAMPLE_CNT_OFFSET

```
#define HCI_CONN_IQ_RPT_SAMPLE_CNT_OFFSET 12
```

Sample count field offset of connection IQ report

Definition at line 1509 of file hci_defs.h.

2.1.2.542 HCI_MAX_CIS_COUNT

```
#define HCI_MAX_CIS_COUNT 0x10
```

Maximum count for CIS

Definition at line 1516 of file hci_defs.h.

2.1.2.543 HCI_MAX_BIS_COUNT

```
#define HCI_MAX_BIS_COUNT 0x10
```

Maximum count for BIS

Definition at line 1523 of file hci_defs.h.

2.1.2.544 HCI_MIN_CIG_ID

```
#define HCI_MIN_CIG_ID 0x00
```

Minimum value for CIG ID.

Definition at line 1530 of file hci_defs.h.

2.1.2.545 HCI_MAX_CIG_ID

```
#define HCI_MAX_CIG_ID 0xEF
```

Maximum value for CIG ID.

Definition at line 1531 of file hci_defs.h.

2.1.2.546 HCI_MIN_CIS_ID

```
#define HCI_MIN_CIS_ID 0x00
```

Minimum value for CIS ID.

Definition at line 1538 of file hci_defs.h.

2.1.2.547 HCI_MAX_CIS_ID

```
#define HCI_MAX_CIS_ID 0xEF
```

Maximum value for CIS ID.

Definition at line 1539 of file hci_defs.h.

2.1.2.548 HCI_PACKING_SEQUENTIAL

```
#define HCI_PACKING_SEQUENTIAL 0x00
```

Sequential

Definition at line 1546 of file hci_defs.h.

2.1.2.549 HCI_PACKING_INTERLEAVED

```
#define HCI_PACKING_INTERLEAVED 0x01
```

Interleaved

Definition at line 1547 of file hci_defs.h.

2.1.2.550 HCI_FRAMING_UNFRAMED

```
#define HCI_FRAMING_UNFRAMED 0x00
```

Unframed

Definition at line 1554 of file hci_defs.h.

2.1.2.551 HCI_FRAMING_FRAMED

```
#define HCI_FRAMING_FRAMED 0x01
```

Framed

Definition at line 1555 of file hci_defs.h.

2.1.2.552 HCI_MIN_SCA

```
#define HCI_MIN_SCA 0x00
```

Minimum value for SCA.

Definition at line 1562 of file hci_defs.h.

2.1.2.553 HCI_MAX_SCA

```
#define HCI_MAX_SCA 0x07
```

Maximum value for SCA.

Definition at line 1563 of file hci_defs.h.

2.1.2.554 HCI_MIN_SDU_SIZE

```
#define HCI_MIN_SDU_SIZE 0x0000
```

Minimum value for SDU size.

Definition at line 1569 of file hci_defs.h.

2.1.2.555 HCI_MAX_SDU_SIZE

```
#define HCI_MAX_SDU_SIZE 0x0FFF
```

Maximum value for SDU size.

Definition at line 1570 of file hci_defs.h.

2.1.2.556 HCI_MIN_SDU_INTERV

```
#define HCI_MIN_SDU_INTERV 0x0000FF
```

Minimum value for SDU interval.

Definition at line 1577 of file hci_defs.h.

2.1.2.557 HCI_MAX_SDU_INTERV

```
#define HCI_MAX_SDU_INTERV 0x0FFFFF
```

Maximum value for SDU interval.

Definition at line 1578 of file hci_defs.h.

2.1.2.558 HCI_DEFAULT_SDU_INTERV

```
#define HCI_DEFAULT_SDU_INTERV 0x004E20
```

Default value for SDU interval.

Definition at line 1579 of file hci_defs.h.

2.1.2.559 HCI_MIN_CIS_TRANS_LAT

```
#define HCI_MIN_CIS_TRANS_LAT 0x0005
```

Minimum value for CIS transport latency.

Definition at line 1586 of file hci_defs.h.

2.1.2.560 HCI_MAX_CIS_TRANS_LAT

```
#define HCI_MAX_CIS_TRANS_LAT 0x0FA0
```

Maximum value for CIS transport latency.

Definition at line 1587 of file hci_defs.h.

2.1.2.561 HCI_DEFAULT_CIS_TRANS_LAT

```
#define HCI_DEFAULT_CIS_TRANS_LAT 0x0028
```

Default value for CIS transport latency.

Definition at line 1588 of file hci_defs.h.

2.1.2.562 HCI_MIN_CIS_FT

```
#define HCI_MIN_CIS_FT 0x01
```

Minimum value for CIS flush time.

Definition at line 1595 of file hci_defs.h.

2.1.2.563 HCI_MAX_CIS_FT

```
#define HCI_MAX_CIS_FT 0xFF
```

Maximum value for CIS flush time.

Definition at line 1596 of file hci_defs.h.

2.1.2.564 HCI_MIN_CIS_BN

```
#define HCI_MIN_CIS_BN 0x00
```

Minimum value for CIS burst number.

Definition at line 1603 of file hci_defs.h.

2.1.2.565 HCI_MAX_CIS_BN

```
#define HCI_MAX_CIS_BN 0x0F
```

Maximum value for CIS burst number.

Definition at line 1604 of file hci_defs.h.

2.1.2.566 HCI_MIN_CIS_RTN

```
#define HCI_MIN_CIS_RTN 0x00
```

Minimum value for CIS retransmission number.

Definition at line 1611 of file hci_defs.h.

2.1.2.567 HCI_MAX_CIS_RTN

```
#define HCI_MAX_CIS_RTN 0x0F
```

Maximum value for CIS retransmission number.

Definition at line 1612 of file hci_defs.h.

2.1.2.568 HCI_ISO_DATA_DIR_INPUT

```
#define HCI_ISO_DATA_DIR_INPUT 0
```

Input (Host to Controller) data path.

Definition at line 1619 of file hci_defs.h.

2.1.2.569 HCI_ISO_DATA_DIR_OUTPUT

```
#define HCI_ISO_DATA_DIR_OUTPUT 1
```

Output (Controller to Host) data path.

Definition at line 1620 of file hci_defs.h.

2.1.2.570 HCI_ISO_DATA_PATH_INPUT_BIT

```
#define HCI_ISO_DATA_PATH_INPUT_BIT (1<<HCI_ISO_DATA_DIR_INPUT)
```

Data path input bit.

Definition at line 1627 of file hci_defs.h.

2.1.2.571 HCI_ISO_DATA_PATH_OUTPUT_BIT

```
#define HCI_ISO_DATA_PATH_OUTPUT_BIT (1<<HCI_ISO_DATA_DIR_OUTPUT)
```

Data path output bit.

Definition at line 1628 of file hci_defs.h.

2.1.2.572 HCI_ISO_DATA_PATH_HCI

```
#define HCI_ISO_DATA_PATH_HCI 0x00
```

HCI data path.

Definition at line 1635 of file hci_defs.h.

2.1.2.573 HCI_ISO_DATA_PATH_VS

```
#define HCI_ISO_DATA_PATH_VS 0x01
```

Vendor Specific.

Definition at line 1636 of file hci_defs.h.

2.1.2.574 HCI_ISO_DATA_PATH_DISABLED

```
#define HCI_ISO_DATA_PATH_DISABLED 0xFF
```

Data path is disabled.

Definition at line 1637 of file hci_defs.h.

2.1.2.575 HCI_ISO_ISO_PLD_TYPE_ZERO_LEN

```
#define HCI_ISO_ISO_PLD_TYPE_ZERO_LEN 0x00
```

Zero length payload.

Definition at line 1644 of file hci_defs.h.

2.1.2.576 HCI_ISO_ISO_PLD_TYPE_VAR_LEN

```
#define HCI_ISO_ISO_PLD_TYPE_VAR_LEN 0x01
```

Variable length payload.

Definition at line 1645 of file hci_defs.h.

2.1.2.577 HCI_ISO_ISO_PLD_TYPE_MAX_LEN

```
#define HCI_ISO_ISO_PLD_TYPE_MAX_LEN 0x02
```

Maximum length payload.

Definition at line 1646 of file hci_defs.h.

2.1.2.578 HCI_MAX_CODEC

```
#define HCI_MAX_CODEC 5
```

Maximum number of codecs to read from the Controller.

Definition at line 1653 of file hci_defs.h.

2.1.2.579 HCI_CODEC_CAP_DATA_LEN

```
#define HCI_CODEC_CAP_DATA_LEN 4
```

Maximum length of codec-specific capability data.

Definition at line 1660 of file hci_defs.h.

2.1.2.580 HCI_CODEC_TRANS_CIS_BIT

```
#define HCI_CODEC_TRANS_CIS_BIT (1<<2)
```

Codec supported over LE CIS.

Definition at line 1667 of file hci_defs.h.

2.1.2.581 HCI_CODEC_TRANS_BIS_BIT

```
#define HCI_CODEC_TRANS_BIS_BIT (1<<3)
```

Codec supported over LE BIS.

Definition at line 1668 of file hci_defs.h.

2.1.2.582 HCI_ISO_HDR_PB_START_FRAG

```
#define HCI_ISO_HDR_PB_START_FRAG 0x00
```

Start fragment of a fragmented SDU.

Definition at line 1675 of file hci_defs.h.

2.1.2.583 HCI_ISO_HDR_PB_CONT_FRAG

```
#define HCI_ISO_HDR_PB_CONT_FRAG 0x01
```

Continuation fragment of a fragmented SDU.

Definition at line 1676 of file hci_defs.h.

2.1.2.584 HCI_ISO_HDR_PB_COMP_FRAG

```
#define HCI_ISO_HDR_PB_COMP_FRAG 0x02
```

Complete SDU.

Definition at line 1677 of file hci_defs.h.

2.1.2.585 HCI_ISO_HDR_PB_END_FRAG

```
#define HCI_ISO_HDR_PB_END_FRAG 0x03
```

The end fragment of a fragmented SDU.

Definition at line 1678 of file hci_defs.h.

2.1.2.586 HCI_ISOAL_SEG_HDR_SC_START

```
#define HCI_ISOAL_SEG_HDR_SC_START 0x00
```

ISOAL segmentation header start bit.

Definition at line 1685 of file hci_defs.h.

2.1.2.587 HCI_ISOAL_SEG_HDR_SC_CONT

```
#define HCI_ISOAL_SEG_HDR_SC_CONT 0x01
```

ISOAL segmentation header continue bit.

Definition at line 1686 of file hci_defs.h.

2.1.2.588 HCI_ID_PACKETCRAFT

```
#define HCI_ID_PACKETCRAFT 0x07E8
```

Packetcraft Inc. company ID

Definition at line 1693 of file hci_defs.h.

2.1.2.589 HCI_LOCAL_VER_MANUFACTURER_POS

```
#define HCI_LOCAL_VER_MANUFACTURER_POS 4
```

Manufacturer location in local version

Definition at line 1701 of file hci_defs.h.

2.1.2.590 HCI_ID_LC3

```
#define HCI_ID_LC3 0x01
```

LC3 ID

Definition at line 1708 of file hci_defs.h.

2.1.2.591 HCI_ID_VS

```
#define HCI_ID_VS 0xFF
```

Vendor specific ID

Definition at line 1709 of file hci_defs.h.

2.1.2.592 HCI_CODEC_TRANSPORT_CIS

```
#define HCI_CODEC_TRANSPORT_CIS 0x02
```

Codec supported over LE CIS

Definition at line 1716 of file hci_defs.h.

2.1.2.593 HCI_CODEC_TRANSPORT_BIS

```
#define HCI_CODEC_TRANSPORT_BIS 0x03
```

Codec supported over LE BIS

Definition at line 1717 of file hci_defs.h.

2.2 WSF Utility API

Data Structures

- struct terminalCommand_t

  Terminal command.

Macros

- #define BDA_ADDR_LEN 6

  BD address length.
- #define BDA_ADDR_STR_LEN (BDA_ADDR_LEN ∗ 2)

  BD address string length.
- #define BDA_ADDR_IS_RPA(bda) (((bda)[5] & 0xC0) == 0x40)

  BDA RPA check.
- #define BDA_ADDR_IS_NRPA(bda) (((bda)[5] & 0xC0) == 0x00)

  BDA NRPA check.
- #define BDA_ADDR_IS_STATIC(bda) (((bda)[5] & 0xC0) == 0xC0)

  BDA static random check.
- #define BDA64_ADDR_IS_RPA(bda64) ((((bda64) >> 40) & 0xC0) == 0x40)

  BDA64 RPA check.
- #define BDA64_ADDR_IS_NRPA(bda64) ((((bda64) >> 40) & 0xC0) == 0x00)

  BDA64 NRPA check.
- #define BDA64_ADDR_IS_STATIC(bda64) ((((bda64) >> 40) & 0xC0) == 0xC0)

  BDA64 static random check.
- #define CALC128_LEN 16

  128-bit integer length in bytes
- #define PRINT_ATTRIBUTE(a, b)

  Print function attributes.
- #define TERMINAL_MAX_ARGC 8u

  Maximum number of arguments to any command.
- #define TERMINAL_MAX_COMMAND_LEN 100u

  Maximum length of command line.
- #define TERMINAL_PRINTF_MAX_LEN 256u

  Maximum length of any printed output.
- #define TERMINAL_STRING_PROMPT "> "

  Prompt string.
- #define TERMINAL_STRING_ERROR "ERROR: "

  Error prefix.
- #define TERMINAL_STRING_USAGE "USAGE: "

  Usage prefix.
- #define TERMINAL_STRING_NEW_LINE "\r\n"

  New line string.
- #define WSTR_IS_HEX_FORMAT(c)
- #define WSTR_IS_BIN_FORMAT(c) (((char)(c)[0] == '0') && ((char)(c)[1] == 'b'))

Typedefs

- typedef uint8_t bdAddr_t[BDA_ADDR_LEN]

  BD address data type.
- typedef uint8_t(∗ terminalHandler_t) (uint32_t argc, char ∗∗argv)

  Handler for a terminal command.
- typedef bool_t(∗ terminalUartTx_t) (const uint8_t ∗pBuf, uint32_t len)

  Handler for transmit.

Enumerations

- enum {
  TERMINAL_ERROR_OK = 0,
  TERMINAL_ERROR_BAD_ARGUMENTS = 1,
  TERMINAL_ERROR_TOO_FEW_ARGUMENTS = 2,
  TERMINAL_ERROR_TOO_MANY_ARGUMENTS = 3,
  TERMINAL_ERROR_EXEC = 4 }

  Terminal command error codes.

Functions

- void BdaCpy (uint8_t ∗pDst, const uint8_t ∗pSrc)

  Copy a BD address from source to destination.
- bool_t BdaCmp (const uint8_t ∗pAddr1, const uint8_t ∗pAddr2)

  Compare two BD addresses.
- uint8_t ∗ BdaClr (uint8_t ∗pDst)

  Set a BD address to all zeros.
- bool_t BdaIsZeros (const uint8_t ∗pAddr)

  Check if a BD address is all zeros.
- char ∗ Bda2Str (const uint8_t ∗pAddr)

  Convert a BD address to a string.
- uint64_t BstreamToBda64 (const uint8_t ∗p)

  Convert bstream to BDA64.
- uint64_t BstreamToUint64 (const uint8_t ∗p)

  Convert bstream to uint64_t.
- void Bda64ToBstream (uint8_t ∗p, uint64_t bda)

  Convert BDA64 to bstream.
- void Uint64ToBstream (uint8_t ∗p, uint64_t n)

  Convert uint64_t to bstream.
- void Calc128Cpy (uint8_t ∗pDst, uint8_t ∗pSrc)

  Copy a 128-bit integer from source to destination.
- void Calc128Cpy64 (uint8_t ∗pDst, uint8_t ∗pSrc)

  Copy a 64-bit integer from source to destination.
- void Calc128Xor (uint8_t ∗pDst, uint8_t ∗pSrc)

  Exclusive-or two 128-bit integers and return the result in pDst.
- uint32_t CalcCrc32 (uint32_t crcInit, uint32_t len, const uint8_t ∗pBuf)

  Calculate the CRC-32 of the given buffer.
- uint32_t PrintVsn (char ∗pStr, uint32_t size, const char ∗pFmt, va_list ap) PRINT_ATTRIBUTE(3

  Print a trace message.
- void TerminalInit (wsfHandlerId_t handlerId)

Initialize terminal.

- void TerminalRegisterUartTxFunc (terminalUartTx_t uartTxFunc)

  Register the UART Tx Function for the platform.
- void TerminalRegisterCommand (terminalCommand_t ∗pCommand)

  Register command with terminal.
- void TerminalHandler (wsfEventMask_t event, wsfMsgHdr_t ∗pMsg)

  Handler for terminal messages.
- void TerminalRx (uint8_t dataByte)

  Called by application when a data byte is received.
- void TerminalTxStr (const char ∗pStr)

  Called by application to transmit string.
- void TerminalTxChar (char c)

  Called by application to transmit character.
- void TerminalTxPrint (const char ∗pStr,...)

  Called by application to print formatted data.
- void TerminalTx (const uint8_t ∗pData, uint16_t len)

  Application function to transmit data..
- void WstrnCpy (char ∗pBuf, const char ∗pData, uint8_t n)

  Copies a string up to a given length.
- void WStrReverseCpy (uint8_t ∗pBuf1, const uint8_t ∗pBuf2, uint16_t len)

  Byte by byte reverse and copy a buffer.
- void WStrReverse (uint8_t ∗pBuf, uint8_t len)

  Byte by byte reverse a buffer.
- void WStrFormatHex (char ∗pBuf, uint32_t val, uint8_t len)

  Format a hex value.
- void WStrHexToArray (const char ∗pStr, uint8_t ∗pBuf, uint16_t len)

  Convert a formatted string to int array, zero out space after the string length.

Variables

- const uint8_t calc128Zeros [CALC128_LEN]

  128-bit zero value

Macros for converting a little endian byte buffer to integers.

- #define BYTES_TO_INT16(n, p) {n = ((int16_t)(p)[0] + ((int16_t)(p)[1] << 8));}

  convert little endian byte buffer to int16_t.
- #define BYTES_TO_UINT16(n, p) {n = ((uint16_t)(p)[0] + ((uint16_t)(p)[1] << 8));}

  convert little endian byte buffer to uint16_t.
- #define BYTES_TO_UINT24(n, p)

  convert little endian byte buffer to uint24_t.
- #define BYTES_TO_UINT32(n, p)

  convert little endian byte buffer to uint32_t.
- #define BYTES_TO_UINT40(n, p)

  convert little endian byte buffer to uint40_t.
- #define BYTES_TO_UINT64(n, p)

  convert little endian byte buffer to uint64_t.

Macros for converting a big endian byte buffer to integers.

- #define BYTES_BE_TO_UINT16(n, p) {n = ((uint16_t)(p)[1] + ((uint16_t)(p)[0] << 8));}

  convert big endian byte buffer to uint16_t.
- #define BYTES_BE_TO_UINT24(n, p)

  convert big endian byte buffer to 24-bit uint32_t (MSB 0).
- #define BYTES_BE_TO_UINT32(n, p)

  convert big endian byte buffer to uint32_t.

Macros for converting little endian integers to array of bytes

- #define UINT16_TO_BYTES(n) ((uint8_t) (n)), ((uint8_t)((n) >> 8))

  convert little endian uint16_t to array of bytes.
- #define UINT32_TO_BYTES(n) ((uint8_t) (n)), ((uint8_t)((n) >> 8)), ((uint8_t)((n) >> 16)), ((uint8_t)((n) >> 24))

  convert little endian uint32_t to array of bytes.

Macros for converting big endian integers to array of bytes

- #define UINT16_TO_BE_BYTES(n) ((uint8_t)((n) >> 8)), ((uint8_t) (n))

  convert big endian uint16_t to array of bytes.
- #define UINT24_TO_BE_BYTES(n) ((uint8_t)((n) >> 16)), ((uint8_t)((n) >> 8)), ((uint8_t) (n))

  convert 24-bit big endian uint32_t (MSB 0) to array of bytes.
- #define UINT32_TO_BE_BYTES(n) ((uint8_t)((n) >> 24)), ((uint8_t)((n) >> 16)), ((uint8_t)((n) >> 8)), ((uint8_t) (n))

  convert big endian uint32_t to array of bytes.

Macros for converting little endian integers to single bytes

- #define UINT16_TO_BYTE0(n) ((uint8_t) (n))

  convert little endian uint16_t to byte 0.
- #define UINT16_TO_BYTE1(n) ((uint8_t) ((n) >> 8))

  convert little endian uint16_t to byte 1.
- #define UINT32_TO_BYTE0(n) ((uint8_t) (n))

  convert little endian uint32_t to byte 0.
- #define UINT32_TO_BYTE1(n) ((uint8_t) ((n) >> 8))

  convert little endian uint32_t to byte 1.
- #define UINT32_TO_BYTE2(n) ((uint8_t) ((n) >> 16))

  convert little endian uint32_t to byte 2.
- #define UINT32_TO_BYTE3(n) ((uint8_t) ((n) >> 24))

  convert little endian uint32_t to byte 3.

Macros for converting a little endian byte stream to integers, with increment.

- #define BSTREAM_TO_INT8(n, p) {n = (int8_t)(∗(p)++);}

  convert little endian byte stream to uint8_t, incrementing one byte.
- #define BSTREAM_TO_UINT8(n, p) {n = (uint8_t)(∗(p)++);}

  convert little endian byte stream to int8_t, incrementing one byte.
- #define BSTREAM_TO_INT16(n, p) {BYTES_TO_INT16(n, p); p += 2;}

  convert little endian byte stream to int16_t, incrementing two bytes.
- #define BSTREAM_TO_UINT16(n, p) {BYTES_TO_UINT16(n, p); p += 2;}

  convert little endian byte stream to uint16_t, incrementing two bytes.
- #define BSTREAM_TO_UINT24(n, p) {BYTES_TO_UINT24(n, p); p += 3;}

  convert little endian byte stream to uint24_t, incrementing three bytes.
- #define BSTREAM_TO_UINT32(n, p) {BYTES_TO_UINT32(n, p); p += 4;}

  convert little endian byte stream to uint32_t, incrementing four bytes.
- #define BSTREAM_TO_UINT40(n, p) {BYTES_TO_UINT40(n, p); p += 5;}

  convert little endian byte stream to uint40_t, incrementing five bytes.
- #define BSTREAM_TO_UINT64(n, p) {n = BstreamToUint64(p); p += 8;}

  convert little endian byte stream to uint64_t, incrementing eigth bytes.
- #define BSTREAM_TO_BDA(bda, p) {BdaCpy(bda, p); p += BDA_ADDR_LEN;}

  convert little endian byte stream to six byte Bluetooth device address, incrementing six bytes.
- #define BSTREAM_TO_BDA64(bda, p) {bda = BstreamToBda64(p); p += BDA_ADDR_LEN;}

  convert little endian byte stream to eight byte Bluetooth device address, incrementing eight bytes.

Macros for converting a big endian byte stream to integers, with increment.

- #define BSTREAM_BE_TO_UINT16(n, p) {BYTES_BE_TO_UINT16(n, p); p += 2;}

  convert big endian byte stream to uint16_t, incrementing one byte.
- #define BSTREAM_BE_TO_UINT24(n, p) {BYTES_BE_TO_UINT24(n, p); p += 3;}

  convert big endian byte stream to 24-bit uint32_t (MSB 0), incrementing one byte.

Macros for converting integers to a little endian byte stream, with increment.

- #define UINT8_TO_BSTREAM(p, n) {∗(p)++ = (uint8_t)(n);}

  convert uint8_t to little endian byte stream, incrementing one byte.
- #define UINT16_TO_BSTREAM(p, n) {∗(p)++ = (uint8_t)(n); ∗(p)++ = (uint8_t)((n) >> 8);}

  convert uint16_t to little endian byte stream, incrementing two bytes.
- #define UINT24_TO_BSTREAM(p, n)

  convert uint24_t to little endian byte stream, incrementing three bytes.
- #define UINT32_TO_BSTREAM(p, n)

  convert uint32_t to little endian byte stream, incrementing four bytes.
- #define UINT40_TO_BSTREAM(p, n)

  convert uint40_t to little endian byte stream, incrementing five bytes.
- #define UINT64_TO_BSTREAM(p, n) {Uint64ToBstream(p, n); p += sizeof(uint64_t);}

  convert uint64_t to little endian byte stream, incrementing eight bytes.
- #define BDA_TO_BSTREAM(p, bda) {BdaCpy(p, bda); p += BDA_ADDR_LEN;}

  convert six byte Bluetooth device address to little endian byte stream, incrementing six bytes.
- #define BDA64_TO_BSTREAM(p, bda) {Bda64ToBstream(p, bda); p += BDA_ADDR_LEN;}

  convert eight byte Bluetooth device address to little endian byte stream, incrementing eigth bytes.

Macros for converting integers to a big endian byte stream, with increment.

- #define UINT16_TO_BE_BSTREAM(p, n) {∗(p)++ = (uint8_t)((n) >> 8); ∗(p)++ = (uint8_t)(n);}

  convert uint16_t to big endian byte stream, incrementing one byte.
- #define UINT32_TO_BE_BSTREAM(p, n)

  convert uint32_t to big endian byte stream, incrementing one byte.

Macros for converting integers to a little endian byte stream, without increment.

- #define UINT16_TO_BUF(p, n) {(p)[0] = (uint8_t)(n); (p)[1] = (uint8_t)((n) >> 8);}

  convert uint16_t to little endian byte stream.
- #define UINT24_TO_BUF(p, n)

  convert uint24_t to little endian byte stream.
- #define UINT32_TO_BUF(p, n)

  convert uint32_t to little endian byte stream.
- #define UINT40_TO_BUF(p, n)

  convert uint40_t to little endian byte stream.

Macros for converting integers to a big endian byte stream, without increment.

- #define UINT16_TO_BE_BUF(p, n) {(p)[0] = (uint8_t)((n) >> 8); (p)[1] = (uint8_t)(n);}

  convert uint16_t to big endian byte stream.
- #define UINT24_TO_BE_BUF(p, n)

  convert 24-bit uint32_t (MSB 0) to big endian byte stream.
- #define UINT32_TO_BE_BUF(p, n)

  convert uint32_t to big endian byte stream.

Macros for comparing a little endian byte buffer to integers.

- #define BYTES_UINT16_CMP(p, n) ((p)[1] == UINT16_TO_BYTE1(n) && (p)[0] == UINT16_TO_BYTE0(n))

  compare 2 byte little endian buffer with a uint16_t.

Macros for IEEE FLOAT type: exponent = byte 3, mantissa = bytes 2-0

- #define FLT_TO_UINT32(m, e) ((m) | ((int32_t)(e) ∗ 16777216))

  Convert float to uint32.
- #define UINT32_TO_FLT(m, e, n) {m = UINT32_TO_FLT_M(n); e = UINT32_TO_FLT_E(n);}

  Convert uint32_t to float.
- #define UINT32_TO_FLT_M(n)

  Convert uint32_t to float mantissa component.
- #define UINT32_TO_FLT_E(n) ((int8_t)(n >> 24))

  Convert uint32_t to float exponent component.

Macros for IEEE SFLOAT type: exponent = bits 15-12, mantissa = bits 11-0

- #define SFLT_TO_UINT16(m, e) ((m) | (0xF000 & ((int16_t)(e) ∗ 4096)))

  Convert sfloat to uint16_t.
- #define UINT16_TO_SFLT(m, e, n) {m = UINT16_TO_SFLT_M(n); e = UINT16_TO_SFLT_E(n);}

  Convert uint16_t to sfloat.
- #define UINT16_TO_SFLT_M(n)

  Convert uint16_T to sfloat mantissa component.
- #define UINT16_TO_SFLT_E(n)

  Convert uint16_T to sfloat exponent component.

2.2.1 Detailed Description

2.2.2 Macro Definition Documentation

2.2.2.1 BYTES_TO_UINT24

```
#define BYTES_TO_UINT24(
                n,
                p )
```

Value:

```
{n = ((uint32_t)(p)[0] + ((uint32_t)(p)[1] << 8) + \
                                        ((uint32_t)(p)[2] << 16));}
```

convert little endian byte buffer to uint24_t.

Definition at line 74 of file bstream.h.

2.2.2.2 BYTES_TO_UINT32

```
#define BYTES_TO_UINT32(
                n,
                p )
```

Value:

```
{n = ((uint32_t)(p)[0] + ((uint32_t)(p)[1] << 8) + \
                                        ((uint32_t)(p)[2] << 16) + ((uint32_t)(p)[3] << 24));}
```

convert little endian byte buffer to uint32_t.

Definition at line 77 of file bstream.h.

2.2.2.3 BYTES_TO_UINT40

```
#define BYTES_TO_UINT40(
              n,
              p )
```

Value:

```
{n = ((uint64_t)(p)[0] + ((uint64_t)(p)[1] << 8) + \
                                        ((uint64_t)(p)[2] << 16) + ((uint64_t)(p)[3] << 24) + \
                                        ((uint64_t)(p)[4] << 32));}
```

convert little endian byte buffer to uint40_t.

Definition at line 80 of file bstream.h.

2.2.2.4 BYTES_TO_UINT64

```
#define BYTES_TO_UINT64(
              n,
              p )
```

Value:

```
{n = ((uint64_t)(p)[0] + ((uint64_t)(p)[1] << 8) + \
                                        ((uint64_t)(p)[2] << 16) + ((uint64_t)(p)[3] << 24) + \
                                        ((uint64_t)(p)[4] << 32) + ((uint64_t)(p)[5] << 40) + \
                                        ((uint64_t)(p)[6] << 48) + ((uint64_t)(p)[7] << 56));}
```

convert little endian byte buffer to uint64_t.

Definition at line 84 of file bstream.h.

2.2.2.5 BYTES_BE_TO_UINT24

```
#define BYTES_BE_TO_UINT24(
              n,
              p )
```

Value:

```
{n = ((uint32_t)(p)[2] + ((uint32_t)(p)[1] << 8) + \
                                           ((uint32_t)(p)[0] << 16));}
```

convert big endian byte buffer to 24-bit uint32_t (MSB 0).

Definition at line 97 of file bstream.h.

2.2.2.6 BYTES_BE_TO_UINT32

```
#define BYTES_BE_TO_UINT32(
              n,
              p )
```

Value:

```
{n = ((uint32_t)(p)[3] + ((uint32_t)(p)[2] << 8) + \
                                        ((uint32_t)(p)[1] << 16) + ((uint32_t)(p)[0] << 24));}
```

convert big endian byte buffer to uint32_t.

Definition at line 100 of file bstream.h.

2.2.2.7 UINT24_TO_BSTREAM

```
#define UINT24_TO_BSTREAM(
              p,
              n )
```

Value:

```
{*(p)++ = (uint8_t)(n); *(p)++ = (uint8_t)((n) >> 8); \
                                 *(p)++ = (uint8_t)((n) >> 16);}
```

convert uint24_t to little endian byte stream, incrementing three bytes.

Definition at line 190 of file bstream.h.

2.2.2.8 UINT32_TO_BSTREAM

```
#define UINT32_TO_BSTREAM(
              p,
              n )
```

Value:

```
{*(p)++ = (uint8_t)(n); *(p)++ = (uint8_t)((n) >> 8); \
                                 *(p)++ = (uint8_t)((n) >> 16); *(p)++ = (uint8_t)((n) >> 24);}
```

convert uint32_t to little endian byte stream, incrementing four bytes.

Definition at line 193 of file bstream.h.

2.2.2.9 UINT40_TO_BSTREAM

```
#define UINT40_TO_BSTREAM(
              p,
              n )
```

Value:

```
{*(p)++ = (uint8_t)(n); *(p)++ = (uint8_t)((n) >> 8); \
                              *(p)++ = (uint8_t)((n) >> 16); *(p)++ = (uint8_t)((n) >> 24); \
                              *(p)++ = (uint8_t)((n) >> 32);}
```

convert uint40_t to little endian byte stream, incrementing five bytes.

Definition at line 196 of file bstream.h.

2.2.2.10 UINT32_TO_BE_BSTREAM

```
#define UINT32_TO_BE_BSTREAM(
              p,
              n )
```

Value:

```
{*(p)++ = (uint8_t)((n) >> 24); *(p)++ = (uint8_t)((n) >> 16);\
                                   *(p)++ = (uint8_t)((n) >> 8); *(p)++ = (uint8_t)(n);}
```

convert uint32_t to big endian byte stream, incrementing one byte.

Definition at line 214 of file bstream.h.

2.2.2.11 UINT24_TO_BUF

```
#define UINT24_TO_BUF(
              p,
              n )
```

Value:

```
{(p)[0] = (uint8_t)(n); (p)[1] = (uint8_t)((n) >> 8); \
                              (p)[2] = (uint8_t)((n) >> 16);}
```

convert uint24_t to little endian byte stream.

Definition at line 225 of file bstream.h.

2.2.2.12 UINT32_TO_BUF

```
#define UINT32_TO_BUF(
             p,
             n )
```

Value:

```
{(p)[0] = (uint8_t)(n); (p)[1] = (uint8_t)((n) >> 8); \
                              (p)[2] = (uint8_t)((n) >> 16); (p)[3] = (uint8_t)((n) >> 24);}
```

convert uint32_t to little endian byte stream.

Definition at line 228 of file bstream.h.

2.2.2.13 UINT40_TO_BUF

```
#define UINT40_TO_BUF(
             p,
             n )
```

Value:

```
{(p)[0] = (uint8_t)(n); (p)[1] = (uint8_t)((n) >> 8); \
                              (p)[2] = (uint8_t)((n) >> 16); (p)[3] = (uint8_t)((n) >> 24);\
                              (p)[4] = (uint8_t)((n) >> 32);}
```

convert uint40_t to little endian byte stream.

Definition at line 231 of file bstream.h.

2.2.2.14 UINT24_TO_BE_BUF

```
#define UINT24_TO_BE_BUF(
             p,
             n )
```

Value:

```
{(p)[0] = (uint8_t)((n) >> 16); (p)[1] = (uint8_t)((n) >> 8); \
                              (p)[2] = (uint8_t)(n);}
```

convert 24-bit uint32_t (MSB 0) to big endian byte stream.

Definition at line 243 of file bstream.h.

2.2.2.15 UINT32_TO_BE_BUF

```
#define UINT32_TO_BE_BUF(
              p,
              n )
```

Value:

```
{(p)[0] = (uint8_t)((n) >> 24); (p)[1] = (uint8_t)((n) >> 16); \
                              (p)[2] = (uint8_t)((n) >> 8); (p)[3] = (uint8_t)(n);}
```

convert uint32_t to big endian byte stream.

Definition at line 246 of file bstream.h.

2.2.2.16 UINT32_TO_FLT_M

```
#define UINT32_TO_FLT_M(
              n )
```

Value:

```
((((n) & 0x00FFFFFF) >= 0x00800000) ? \
                               ((int32_t)(((n) | 0xFF000000))) : ((int32_t)((n) & 0x00FFFFFF)))
```

Convert uint32_t to float mantissa component.

Definition at line 267 of file bstream.h.

2.2.2.17 UINT16_TO_SFLT_M

```
#define UINT16_TO_SFLT_M(
              n )
```

Value:

```
((((n) & 0x0FFF) >= 0x0800) ? \
                               ((int16_t)(((n) | 0xF000))) : ((int16_t)((n) & 0x0FFF)))
```

Convert uint16_T to sfloat mantissa component.

Definition at line 282 of file bstream.h.

2.2.2.18 UINT16_TO_SFLT_E

```
#define UINT16_TO_SFLT_E(
              n )
```

Value:

```
(((n >> 12) >= 0x0008) ? \
                              ((int8_t)(((n >> 12) | 0xF0))) : ((int8_t)(n >> 12)))
```

Convert uint16_T to sfloat exponent component.

Definition at line 285 of file bstream.h.

2.2.2.19 WSTR_IS_HEX_FORMAT

```
#define WSTR_IS_HEX_FORMAT(
              c )
```

Value:

```
(((char)(c)[0] == '0') && \
                           (((char)(c)[1] == 'x') || ((char)(c)[1] == 'X')))
```

HEX format '0x' or '0X' check

Definition at line 40 of file wstr.h.

2.2.2.20 WSTR_IS_BIN_FORMAT

```
#define WSTR_IS_BIN_FORMAT(
              c ) (((char)(c)[0] == '0') && ((char)(c)[1] == 'b'))
```

Binary format '0b' check

Definition at line 44 of file wstr.h.

2.2.3 Typedef Documentation

2.2.3.1 terminalHandler_t

```
typedef uint8_t(* terminalHandler_t) (uint32_t argc, char **argv)
```

Handler for a terminal command.

Parameters

argc	The number of arguments passed to the command.
argv	The array of arguments; the 0th argument is the command.

Returns

Error code.

Definition at line 103 of file terminal.h.

2.2.3.2 terminalUartTx_t

```
typedef bool_t(* terminalUartTx_t) (const uint8_t *pBuf, uint32_t len)
```

Handler for transmit.

Parameters

pBuf	Buffer to transmit.
len	Number of bytes to transmit.

Definition at line 113 of file terminal.h.

2.2.4 Enumeration Type Documentation

2.2.4.1 anonymous enum

```
anonymous enum
```

Terminal command error codes.

Enumerator

TERMINAL_ERROR_OK	Command completed.
TERMINAL_ERROR_BAD_ARGUMENTS	ERROR: Invalid argument(s)
TERMINAL_ERROR_TOO_FEW_ARGUMENTS	ERROR: Too few arguments.
TERMINAL_ERROR_TOO_MANY_ARGUMENTS	ERROR: Too many arguments.
TERMINAL_ERROR_EXEC	Command completed with execution error.

Definition at line 80 of file terminal.h.

```
81 {
   TERMINAL_ERROR_OK                   = 0,  /*!< \brief Command completed. */
   TERMINAL_ERROR_BAD_ARGUMENTS        = 1,  /*!< \brief ERROR: Invalid
     argument(s) */
   TERMINAL_ERROR_TOO_FEW_ARGUMENTS    = 2,  /*!< \brief ERROR: Too few
     arguments */
   TERMINAL_ERROR_TOO_MANY_ARGUMENTS   = 3,  /*!< \brief ERROR: Too many
     arguments */
   TERMINAL_ERROR_EXEC                 = 4   /*!< \brief Command completed with execution
     error. */
};
```

2.2.5 Function Documentation

2.2.5.1 BdaCpy()

```
void BdaCpy (
             uint8_t * pDst,
             const uint8_t * pSrc )
```

Copy a BD address from source to destination.

Parameters

pDst	Pointer to destination.
pSrc	Pointer to source.

2.2.5.2 BdaCmp()

```
bool_t BdaCmp (
             const uint8_t * pAddr1,
             const uint8_t * pAddr2 )
```

Compare two BD addresses.

Parameters

pAddr1	First address.
pAddr2	Second address.

Returns

TRUE if addresses match, FALSE otherwise.

2.2.5.3 BdaClr()

```
uint8_t* BdaClr (
             uint8_t * pDst )
```

Set a BD address to all zeros.

Parameters

pDst	Pointer to destination.

Returns

pDst + BDA_ADDR_LEN

2.2.5.4 BdaIsZeros()

```
bool_t BdaIsZeros (
             const uint8_t * pAddr )
```

Check if a BD address is all zeros.

Parameters

pAddr	Pointer to address.

Returns

TRUE if address is all zeros, FALSE otherwise.

2.2.5.5 Bda2Str()

```
char* Bda2Str (
             const uint8_t * pAddr )
```

Convert a BD address to a string.

Parameters

pAddr	Pointer to BD address.

Returns

Pointer to string.

2.2.5.6 BstreamToBda64()

```
uint64_t BstreamToBda64 (
            const uint8_t * p )
```

Convert bstream to BDA64.

Parameters

p	Bstream pointer.

Returns

Resulting BDA64 number.

2.2.5.7 BstreamToUint64()

```
uint64_t BstreamToUint64 (
            const uint8_t * p )
```

Convert bstream to uint64_t.

Parameters

p	Bstream pointer.

Returns

Resulting uint64_t number.

2.2.5.8 Bda64ToBstream()

```
void Bda64ToBstream (
            uint8_t * p,
            uint64_t bda )
```

Convert BDA64 to bstream.

Parameters

p	Bstream pointer.
bda	uint64_t BDA.

2.2.5.9 Uint64ToBstream()

```
void Uint64ToBstream (
             uint8_t * p,
             uint64_t n )
```

Convert uint64_t to bstream.

Parameters

p	Bstream pointer.
n	uint64_t number.

2.2.5.10 Calc128Cpy()

```
void Calc128Cpy (
             uint8_t * pDst,
             uint8_t * pSrc )
```

Copy a 128-bit integer from source to destination.

Parameters

pDst	Pointer to destination.
pSrc	Pointer to source.

2.2.5.11 Calc128Cpy64()

```
void Calc128Cpy64 (
             uint8_t * pDst,
             uint8_t * pSrc )
```

Copy a 64-bit integer from source to destination.

Parameters

pDst	Pointer to destination.
pSrc	Pointer to source.

2.2.5.12 Calc128Xor()

```
void Calc128Xor (
             uint8_t * pDst,
             uint8_t * pSrc )
```

Exclusive-or two 128-bit integers and return the result in pDst.

Parameters

pDst	Pointer to destination.
pSrc	Pointer to source.

2.2.5.13 CalcCrc32()

```
uint32_t CalcCrc32 (
             uint32_t crcInit,
             uint32_t len,
             const uint8_t * pBuf )
```

Calculate the CRC-32 of the given buffer.

Parameters

crcInit	Initial value of the CRC.
len	Length of the buffer.
pBuf	Buffer to compute the CRC.

This routine was originally generated with crcmod.py using the following parameters:

- polynomial 0x104C11DB7
- bit reverse algorithm

2.2.5.14 PrintVsn()

```
uint32_t PrintVsn (
             char * pStr,
             uint32_t size,
             const char * pFmt,
             va_list ap )
```

Print a trace message.

Parameters

pStr	Storage for formatted string.
size	Maximum number of characters to store.
pFmt	Format string.
ap	Arguments.

Returns

Number of characters stored.

2.2.5.15 TerminalInit()

```
void TerminalInit (
            wsfHandlerId_t handlerId )
```

Initialize terminal.

Parameters

handler↩ Id	Handler ID for TerminalHandler().

2.2.5.16 TerminalRegisterUartTxFunc()

```
void TerminalRegisterUartTxFunc (
            terminalUartTx_t uartTxFunc )
```

Register the UART Tx Function for the platform.

Parameters

uartTxFunc	UART Tx callback function.

2.2.5.17 TerminalRegisterCommand()

```
void TerminalRegisterCommand (
            terminalCommand_t * pCommand )
```

Register command with terminal.

Parameters

pCommand	Command.

2.2.5.18 TerminalHandler()

```
void TerminalHandler (
             wsfEventMask_t event,
             wsfMsgHdr_t * pMsg )
```

Handler for terminal messages.

Parameters

event	WSF event mask.
pMsg	WSF message.

2.2.5.19 TerminalRx()

```
void TerminalRx (
             uint8_t dataByte )
```

Called by application when a data byte is received.

Parameters

dataByte	received byte

2.2.5.20 TerminalTxStr()

```
void TerminalTxStr (
             const char * pStr )
```

Called by application to transmit string.

Parameters

pStr	String.

2.2.5.21 TerminalTxChar()

```
void TerminalTxChar (
             char c )
```

Called by application to transmit character.

Parameters

c	Character.

2.2.5.22 TerminalTxPrint()

```
void TerminalTxPrint (
             const char * pStr,
              ...  )
```

Called by application to print formatted data.

Parameters

pStr	Message format string
...	Additional arguments, printf-style

2.2.5.23 TerminalTx()

```
void TerminalTx (
             const uint8_t * pData,
             uint16_t len )
```

Application function to transmit data..

Parameters

pData	Data.
len	Length of data, in bytes.

2.2.5.24 WstrnCpy()

```
void WstrnCpy (
             char * pBuf,
             const char * pData,
             uint8_t n )
```

Copies a string up to a given length.

Parameters

pBuf	Pointer to buffer to copy to.
pData	Pointer to the string to copy.
n	Size of pBuf in bytes.

Returns

none.

2.2.5.25 WStrReverseCpy()

```
void WStrReverseCpy (
             uint8_t * pBuf1,
             const uint8_t * pBuf2,
             uint16_t len )
```

Byte by byte reverse and copy a buffer.

Parameters

pBuf1	Buffer to hold reversed copy.
pBuf2	Buffer to copy.
len	Size of pBuf1 and pBuf2 in bytes.

2.2.5.26 WStrReverse()

```
void WStrReverse (
             uint8_t * pBuf,
             uint8_t len )
```

Byte by byte reverse a buffer.

Parameters

pBuf	Buffer to reverse.
len	size of pBuf in bytes.

2.2.5.27 WStrFormatHex()

```
void WStrFormatHex (
             char * pBuf,
             uint32_t val,
             uint8_t len )
```

Format a hex value.

Parameters

pBuf	Storage for string representation of value.
val	Value.
len	Length of value, in bits.

2.2.5.28 WStrHexToArray()

```
void WStrHexToArray (
             const char * pStr,
             uint8_t * pBuf,
             uint16_t len )
```

Convert a formatted string to int array, zero out space after the string length.

Parameters

pStr	Pointer to the string to convert.
pBuf	Pointer to destination buffer.
len	Size of pBuf in bytes.

Returns

none.

2.3 WSF Assert API

Macros

- #define WSF_ASSERT_ENABLED FALSE

  Enable assertion statements.
- #define WSF_ASSERT(expr) (void)(expr);

  Run-time assert macro. The assert executes when the expression is FALSE.
- #define WSF_CT_ASSERT(expr) extern char wsf_ct_assert[(expr) ? 1 : -1]

  Compile-time assert macro. This macro causes a compiler error when the expression is FALSE. Note that this macro is generally used at file scope to test constant expressions. Errors may result of it is used in executing code.

Functions

- void **WsfAssert** (const char ∗pFile, uint16_t line)
- uint16_t WsfAssertNum (void)

  Get number of asserts.
- void WsfAssertTrapEnable (bool_t enaAssertTrap)

  Enable assert trap.
- void WsfAssertRegister (void(∗cback)(void))

  Register assert handler.

2.3.1 Detailed Description

2.3.2 Macro Definition Documentation

2.3.2.1 WSF_ASSERT

```
#define WSF_ASSERT(
              expr ) (void)(expr);
```

Run-time assert macro. The assert executes when the expression is FALSE.

Parameters

expr	Boolean expression to be tested.

Definition at line 131 of file wsf_assert.h.

2.3.2.2 WSF_CT_ASSERT

```
#define WSF_CT_ASSERT(
              expr ) extern char wsf_ct_assert[(expr) ?  1 :  -1]
```

Compile-time assert macro. This macro causes a compiler error when the expression is FALSE. Note that this macro is generally used at file scope to test constant expressions. Errors may result of it is used in executing code.

Parameters

expr	Boolean expression to be tested.

Definition at line 143 of file wsf_assert.h.

2.3.3 Function Documentation

2.3.3.1 WsfAssertNum()

```
uint16_t WsfAssertNum (
            void  )
```

Get number of asserts.

Returns

Number of asserts.

2.3.3.2 WsfAssertTrapEnable()

```
void WsfAssertTrapEnable (
            bool_t enaAssertTrap )
```

Enable assert trap.

Parameters

enaAssertTrap	TRUE to enable assert trap.

2.3.3.3 WsfAssertRegister()

```
void WsfAssertRegister (
            void(*)(void) cback )
```

Register assert handler.

Parameters

cback	Callback called upon assert condition.

Returns

None

2.4 WSF Buffer API

Data Structures

- struct wsfBufPoolDesc_t

  Buffer pool descriptor structure.
- struct WsfBufPoolStat_t

  Pool statistics.
- struct wsfBufDiagAllocFail_t

  WSF buffer diagnostics - buffer allocation failure.
- struct WsfBufDiag_t

  WSF buffer diagnostics message.

Macros

- #define WSF_BUF_FREE_CHECK_ASSERT TRUE

  Check if trying to free a buffer that is already free.
- #define WSF_BUF_ALLOC_BEST_FIT_FAIL_ASSERT FALSE

  Assert on best-fit buffer allocation failure.
- #define WSF_BUF_ALLOC_FAIL_ASSERT FALSE

  Assert on buffer allocation failure.
- #define WSF_BUF_STATS_HIST FALSE

  Buffer histogram stats.
- #define WSF_BUF_STATS_MAX_LEN 128

  Length of the buffer statistics array.
- #define WSF_BUF_STATS_MAX_POOL 32

  Max number of pools can allocate.
- #define WSF_BUF_ALLOC_FAILED 1

  Failure Codes.
- #define WSF_BUF_STATS FALSE

  Enable buffer allocation statistics.

Typedefs

- typedef void(∗ WsfBufDiagCback_t) (WsfBufDiag_t ∗pInfo)

  Callback providing WSF buffer diagnostic messages.

Functions

- uint32_t WsfBufCalcSize (uint8_t numPools, wsfBufPoolDesc_t ∗pDesc)

  Calculate size required by the buffer pool.
- uint32_t WsfBufInit (uint32_t bufMemLen, uint8_t ∗pBufMem, uint8_t numPools, wsfBufPoolDesc_t ∗pDesc)

  Initialize the buffer pool service. This function should only be called once upon system initialization.
- bool_t CheckWsfBufAlloc (uint16_t len)

  Verify whether a buffer with required length is available.
- void ∗ WsfBufAlloc (uint16_t len)

  Allocate a buffer.
- void WsfBufFree (void ∗pBuf)

Free a buffer.

- uint8_t ∗ WsfBufGetAllocStats (void)

  Diagnostic function to get the buffer allocation statistics.
- uint8_t ∗ WsfBufGetPoolOverFlowStats (void)

  Diagnostic function to get the number of overflow times for each pool.
- uint8_t WsfBufGetNumPool (void)

  Get number of pools.
- void WsfBufGetPoolStats (WsfBufPoolStat_t ∗pStat, uint8_t numPool)

  Get statistics for each pool.
- void WsfBufDiagRegister (WsfBufDiagCback_t callback)

  Called to register the buffer diagnostics callback function.
- uint32_t WsfBufNumOutstanding (void)

  Get the number of outstanding memory pool buffers.

2.4.1 Detailed Description

2.4.2 Typedef Documentation

2.4.2.1 WsfBufDiagCback_t

```
typedef void(∗ WsfBufDiagCback_t) (WsfBufDiag_t ∗pInfo)
```

Callback providing WSF buffer diagnostic messages.

Parameters

pInfo	Diagnostics message.

Definition at line 158 of file wsf_buf.h.

2.4.3 Function Documentation

2.4.3.1 WsfBufCalcSize()

```
uint32_t WsfBufCalcSize (
            uint8_t numPools,
            wsfBufPoolDesc_t ∗ pDesc )
```

Calculate size required by the buffer pool.

Parameters

numPools	Number of buffer pools.
pDesc	Array of buffer pool descriptors, one for each pool.

Returns

Amount of pBufMem used.

2.4.3.2 WsfBufInit()

```
uint32_t WsfBufInit (
            uint32_t bufMemLen,
            uint8_t * pBufMem,
            uint8_t numPools,
            wsfBufPoolDesc_t * pDesc )
```

Initialize the buffer pool service. This function should only be called once upon system initialization.

Parameters

bufMemLen	Length in bytes of memory pointed to by pBufMem.
pBufMem	Memory in which to store the pools used by the buffer pool service.
numPools	Number of buffer pools.
pDesc	Array of buffer pool descriptors, one for each pool.

Returns

Amount of pBufMem used or 0 for failures.

2.4.3.3 CheckWsfBufAlloc()

```
bool_t CheckWsfBufAlloc (
            uint16_t len )
```

Verify whether a buffer with required length is available.

Parameters

len	Length of buffer to allocate.

Returns

True if buffer is available, false if buffer allocation is not possible.

2.4.3.4 WsfBufAlloc()

```
void* WsfBufAlloc (
             uint16_t len )
```

Allocate a buffer.

Parameters

len	Length of buffer to allocate.

Returns

Pointer to allocated buffer or NULL if allocation fails.

2.4.3.5 WsfBufFree()

```
void WsfBufFree (
             void * pBuf )
```

Free a buffer.

Parameters

pBuf	Buffer to free.

2.4.3.6 WsfBufGetAllocStats()

```
uint8_t* WsfBufGetAllocStats (
             void  )
```

Diagnostic function to get the buffer allocation statistics.

Returns

Buffer allocation statistics array.

2.4.3.7 WsfBufGetPoolOverFlowStats()

```
uint8_t* WsfBufGetPoolOverFlowStats (
             void  )
```

Diagnostic function to get the number of overflow times for each pool.

Returns

Overflow times statistics array

2.4.3.8 WsfBufGetNumPool()

```
uint8_t WsfBufGetNumPool (
             void  )
```

Get number of pools.

Returns

Number of pools.

2.4.3.9 WsfBufGetPoolStats()

```
void WsfBufGetPoolStats (
             WsfBufPoolStat_t * pStat,
             uint8_t numPool )
```

Get statistics for each pool.

Parameters

pStat	Buffer to store the statistics.
numPool	Number of pool elements.

Returns

Pool statistics.

2.4.3.10 WsfBufDiagRegister()

```
void WsfBufDiagRegister (
             WsfBufDiagCback_t callback )
```

Called to register the buffer diagnostics callback function.

Parameters

callback	Pointer to the callback function.

2.4.3.11 WsfBufNumOutstanding()

```
uint32_t WsfBufNumOutstanding (
            void  )
```

Get the number of outstanding memory pool buffers.

Returns

The number of outstanding buffers

2.5 WSF Buffer IO API

Typedefs

- typedef void(∗ WsfBufIoUartRxCback_t) (uint8_t rxByte)

  Buffer IO UART Rx callback.

Functions

- uint32_t WsfBufIoUartInit (void ∗pBuf, uint32_t size)

  Initialize the platform UART.
- void WsfBufIoUartRegister (WsfBufIoUartRxCback_t rxCback)

  Register the platform UART RX callback.
- bool_t WsfBufIoWrite (const uint8_t ∗pBuf, uint32_t len)

  Transmit buffer on platform UART.

2.5.1 Detailed Description

2.5.2 Function Documentation

2.5.2.1 WsfBufIoUartInit()

```
uint32_t WsfBufIoUartInit (
            void * pBuf,
            uint32_t size )
```

Initialize the platform UART.

Parameters

pBuf	Tx Buffer pointer.
size	Length of buffer.

Returns

memory used.

2.5.2.2 WsfBufIoUartRegister()

```
void WsfBufIoUartRegister (
            WsfBufIoUartRxCback_t rxCback )
```

Register the platform UART RX callback.

Parameters

in	*Callback*	function for UART RX.

2.5.2.3 WsfBufIoWrite()

```
bool_t WsfBufIoWrite (
            const uint8_t * pBuf,
            uint32_t len )
```

Transmit buffer on platform UART.

Parameters

pBuf	Buffer to transmit.
len	Length of buffer in octets.

2.6 WSF Critical Section API

Macros

- #define WSF_CS_STATS FALSE

  Use CS statistics hooks.
- #define WSF_CS_INIT(cs)

  Initialize critical section. This macro may define a variable.
- #define WSF_CS_ENTER(cs) WsfCsEnter()

  Enter a critical section.
- #define WSF_CS_EXIT(cs) WsfCsExit()

  Exit a critical section.

Functions

- uint32_t WsfCsStatsGetCsWaterMark (void)

  Get critical section duration watermark level.
- void WsfCsEnter (void)

  Enter a critical section.
- void WsfCsExit (void)

  Exit a critical section.

2.6.1 Detailed Description

2.6.2 Macro Definition Documentation

2.6.2.1 WSF_CS_INIT

```
#define WSF_CS_INIT(
              cs )
```

Initialize critical section. This macro may define a variable.

Parameters

cs	Critical section variable to be defined.

Definition at line 57 of file wsf_cs.h.

2.6.2.2 WSF_CS_ENTER

```
#define WSF_CS_ENTER(
              cs ) WsfCsEnter()
```

Enter a critical section.

Parameters

cs	Critical section variable.

Definition at line 68 of file wsf_cs.h.

2.6.2.3 WSF_CS_EXIT

```
#define WSF_CS_EXIT(
              cs ) WsfCsExit()
```

Exit a critical section.

Parameters

cs	Critical section variable.

Definition at line 79 of file wsf_cs.h.

2.6.3 Function Documentation

2.6.3.1 WsfCsStatsGetCsWaterMark()

```
uint32_t WsfCsStatsGetCsWaterMark (
             void  )
```

Get critical section duration watermark level.

Returns

Critical section duration watermark level.

2.7 WSF Embedded File System API

Data Structures

- struct wsfEsfAttributes_t

  File attributes data type.
- struct wsfEfsControl_t

  File control block data type.
- struct wsfEfsFileInfo_t

  File Listing Information.
- struct wsfEfsMedia_t

  Media Control data type.

Macros

- #define WSF_EFS_MAX_FILES 6

  Max Number of Files and Media.
- #define WSF_EFS_MAX_MEDIA 4

  Max Number of Media.
- #define WSF_EFS_FILE_OFFSET_ANY 0xFFFFFFFF

  Offset to WsfEfsAddFile indicating any file offset can be used.
- #define WSF_EFS_NAME_LEN 16

  File name length in bytes.
- #define WSF_EFS_VERSION_LEN 16

  File version length in bytes.
- #define WSF_EFS_USER_CMD 0x80

  Media Specific Command Identifiers reserved for applications begin at 0x80.

Typedefs

- typedef uint16_t wsfEfsHandle_t

  File handle data type.
- typedef uint8_t wsfMediaInitFunc_t(void)

  Media Init function, called when media is registered.
- typedef uint8_t wsfMediaEraseFunc_t(uint32_t address, uint32_t size)

  Media Erase function.
- typedef uint8_t wsfMediaReadFunc_t(uint8_t ∗pBuf, uint32_t address, uint32_t size)

  Media Read function.
- typedef uint8_t wsfMediaWriteFunc_t(const uint8_t ∗pBuf, uint32_t address, uint32_t size)

  Media Write function.
- typedef uint8_t wsfMediaHandleCmdFunc_t(uint8_t cmd, uint32_t param)

  Media Specific Command handler.
- typedef const wsfEfsMedia_t ∗ pWsfEfsMedia_t

  Pointer to Media Control data type.

Functions

- void WsfEfsInit (void)

  Initialise the embedded file system.
- wsfEfsHandle_t WsfEfsAddFile (uint32_t maxSize, uint8_t media, wsfEsfAttributes_t ∗pAttr, uint32_t offset)

  Create a file in the embedded file system.
- uint8_t WsfEfsRemoveFile (wsfEfsHandle_t handle)

  Deletes a file in the embedded file system.
- uint8_t WsfEfsErase (wsfEfsHandle_t handle)

  Clears the contents of a file without deleting the file.
- uint8_t WsfEfsGetAttributes (wsfEfsHandle_t handle, wsfEsfAttributes_t ∗pAttr)

  Gets the attributes of a file.
- uint8_t WsfEfsSetAttributes (wsfEfsHandle_t handle, wsfEsfAttributes_t ∗pInfo)

  Updates the attributes of a file.
- uint16_t WsfEfsGet (wsfEfsHandle_t handle, uint32_t offset, uint8_t ∗pBuffer, uint16_t len)

  Copies data from a file.
- uint16_t WsfEfsPut (wsfEfsHandle_t handle, uint32_t offset, const uint8_t ∗pBuffer, uint16_t len)

  Writes data to a file.
- uint8_t WsfEfsRegisterMedia (const wsfEfsMedia_t ∗pMediaCtrl, uint8_t mediaID)

  Registers a File Storage Medium with the Embedded File System.
- wsfEfsControl_t ∗ WsfEfsGetFileByHandle (wsfEfsHandle_t handle)

  Returns the file control block for the given handle.
- char ∗ WsfEfsGetFileName (wsfEfsHandle_t handle)

  Get the name of a file.
- char ∗ WsfEfsGetFileVersion (wsfEfsHandle_t handle)

  Get the version of a file.
- uint32_t WsfEfsGetFileSize (wsfEfsHandle_t handle)

  Get the size of a file.
- uint32_t WsfEfsGetFileMaxSize (wsfEfsHandle_t handle)

  Get the number of bytes of memory reserved for use by a file.
- uint8_t WsfEfsGetFileType (wsfEfsHandle_t handle)

  Get the type of a file.
- uint16_t WsfEfsGetFilePermissions (wsfEfsHandle_t handle)

  Get the permissions of a file.
- uint8_t WsfEfsMediaSpecificCommand (wsfEfsHandle_t handle, uint8_t cmd, uint32_t param)

  Execute a media specific command on a file.

Status Codes

- #define WSF_EFS_SUCCESS 0

  Success.
- #define WSF_EFS_FAILURE 1

  Failure.
- #define WSF_EFS_CBACK_REQUIRED 2

  File sytem callback required.
- #define WSF_EFS_GET_FAILED 0xFFFF

  Get operation failure.
- #define WSF_EFS_PUT_FAILED 0xFFFF

  PUT operation failure.

Invalid Parameter Identifiers

- #define WSF_EFS_INVALID_HANDLE 0xFFFF

  Invalid Handle.
- #define WSF_EFS_INVALID_OFFSET 0xFFFFFFFF

  Invalid Offset.
- #define WSF_EFS_INVALID_SIZE 0xFFFFFFFF

  Invalid Size.
- #define WSF_EFS_INVALID_MEDIA 0xFF

  Invalid Media.

File Types

- #define WSF_EFS_FILE_TYPE_BULK 0

  Bulk File Type.
- #define WSF_EFS_FILE_TYPE_STREAM 1

  Stream File Type.

File Permissions

- #define WSF_EFS_REMOTE_PERMISSIONS_MASK 0xFF

  Remote Permissions.
- #define WSF_EFS_REMOTE_GET_PERMITTED 0x01

  Remote Get Permitted.
- #define WSF_EFS_REMOTE_PUT_PERMITTED 0x02

  Remote Put Permitted.
- #define WSF_EFS_REMOTE_ERASE_PERMITTED 0x04

  Remote Erase Permitted.
- #define WSF_EFS_REMOTE_VERIFY_PERMITTED 0x08

  Remote Verify Permitted.
- #define WSF_EFS_LOCAL_GET_PERMITTED 0x0100

  Local Get Permitted.
- #define WSF_EFS_LOCAL_PUT_PERMITTED 0x0200

  Local Put Permitted.
- #define WSF_EFS_LOCAL_ERASE_PERMITTED 0x0400

  Local Erase Permitted.
- #define WSF_EFS_REMOTE_VISIBLE 0x0800

  File Visible via Remote WDXS.

Standard Media Specific Command Identifiers

- #define WSF_EFS_WDXS_PUT_COMPLETE_CMD 0x00

  Put Complete.
- #define WSF_EFS_VALIDATE_CMD 0x01

  Validate Req for the file.
- #define WSF_EFS_RESTORE_ATTRS_CMD 0x02

  Restore file attributes.
- #define WSF_EFS_GET_FS_CBACK_MODE_CMD 0x03

  Filesystem callback check.
- #define WSF_EFS_CFG_FOTA_CMD 0x04

  Configure FOTA and reboot.

2.7.1 Detailed Description

2.7.2 Typedef Documentation

2.7.2.1 wsfMediaInitFunc_t

```
typedef uint8_t wsfMediaInitFunc_t(void)
```

Media Init function, called when media is registered.

Returns

Status of the operation.

Definition at line 183 of file wsf_efs.h.

2.7.2.2 wsfMediaEraseFunc_t

```
typedef uint8_t wsfMediaEraseFunc_t(uint32_t address, uint32_t size)
```

Media Erase function.

Parameters

pAddress	Address in media to start erasing.
size	Number of bytes to erase.

Returns

Status of the operation.

Definition at line 198 of file wsf_efs.h.

2.7.2.3 wsfMediaReadFunc_t

```
typedef uint8_t wsfMediaReadFunc_t(uint8_t *pBuf, uint32_t address, uint32_t size)
```

Media Read function.

Parameters

pBuf	Buffer to hold data.
pAddress	Address in media to read from.
size	Size of pBuf in bytes.

Returns

Status of the operation.

Definition at line 215 of file wsf_efs.h.

2.7.2.4 wsfMediaWriteFunc_t

```
typedef uint8_t wsfMediaWriteFunc_t(const uint8_t *pBuf, uint32_t address, uint32_t size)
```

Media Write function.

Parameters

pBuf	Buffer with data to be written.
pAddress	Address in media to write to.
size	Size of pBuf in bytes.

Returns

Status of the operation.

Definition at line 232 of file wsf_efs.h.

2.7.2.5 wsfMediaHandleCmdFunc_t

```
typedef uint8_t wsfMediaHandleCmdFunc_t(uint8_t cmd, uint32_t param)
```

Media Specific Command handler.

Parameters

cmd	Identifier of the media specific command.
param	Optional Parameter to the command.

Returns

Status of the operation.

Definition at line 245 of file wsf_efs.h.

2.7.3 Function Documentation

2.7.3.1 WsfEfsInit()

```
void WsfEfsInit (
            void  )
```

Initialise the embedded file system.

Returns

none.

2.7.3.2 WsfEfsAddFile()

```
wsfEfsHandle_t WsfEfsAddFile (
            uint32_t maxSize,
            uint8_t media,
            wsfEsfAttributes_t * pAttr,
            uint32_t offset )
```

Create a file in the embedded file system.

Parameters

maxSize	Max length in bytes of of the file.
media	Identifier of the media where the file is stored.
pAttr	Attributes of the file
offset	Offset address of the file in the memory space.

Returns

File Handle, or WSF_EFS_INVALID_HANDLE.

2.7.3.3 WsfEfsRemoveFile()

```
uint8_t WsfEfsRemoveFile (
            wsfEfsHandle_t handle )
```

Deletes a file in the embedded file system.

Parameters

handle	Handle identifying the file.

Returns

WSF_EFS_SUCCESS or WSF_EFS_FAILURE.

2.7.3.4 WsfEfsErase()

```
uint8_t WsfEfsErase (
            wsfEfsHandle_t handle )
```

Clears the contents of a file without deleting the file.

Parameters

handle	Handle identifying the file.

Returns

WSF_EFS_SUCCESS or WSF_EFS_FAILURE.

2.7.3.5 WsfEfsGetAttributes()

```
uint8_t WsfEfsGetAttributes (
            wsfEfsHandle_t handle,
            wsfEsfAttributes_t * pAttr )
```

Gets the attributes of a file.

Parameters

handle	Handle identifying the file.
pAttr	Pointer to memory to store the attributes.

Returns

WSF_EFS_SUCCESS or WSF_EFS_FAILURE.

2.7.3.6 WsfEfsSetAttributes()

```
uint8_t WsfEfsSetAttributes (
            wsfEfsHandle_t handle,
            wsfEsfAttributes_t * pInfo )
```

Updates the attributes of a file.

Parameters

handle	Handle identifying the file.
pInfo	Pointer to memory to with the updated attributes.

Returns

WSF_EFS_SUCCESS or WSF_EFS_FAILURE.

2.7.3.7 WsfEfsGet()

```
uint16_t WsfEfsGet (
              wsfEfsHandle_t handle,
              uint32_t offset,
              uint8_t * pBuffer,
              uint16_t len )
```

Copies data from a file.

Parameters

handle	Handle identifying the file.
offset	Offset into the file to begin copying from.
pBuffer	Location to copy the data to.
len	Number of bytes to copy into pBuffer.

Returns

The number of bytes read from the file

2.7.3.8 WsfEfsPut()

```
uint16_t WsfEfsPut (
              wsfEfsHandle_t handle,
              uint32_t offset,
              const uint8_t * pBuffer,
              uint16_t len )
```

Writes data to a file.

Parameters

handle	Handle identifying the file.
offset	Offset into the file to begin writing to.
pBuffer	Data to write to the file.
len	Number of bytes to write to the file.

Returns

The number of bytes written to the file

2.7.3.9 WsfEfsRegisterMedia()

```
uint8_t WsfEfsRegisterMedia (
            const wsfEfsMedia_t * pMediaCtrl,
            uint8_t mediaID )
```

Registers a File Storage Medium with the Embedded File System.

Parameters

pMediaCtrl	Pointer to the media control structure.
mediaID	User specified identifier of the media.

Returns

WSF_EFS_SUCCESS or WSF_EFS_FAILURE.

2.7.3.10 WsfEfsGetFileByHandle()

```
wsfEfsControl_t* WsfEfsGetFileByHandle (
            wsfEfsHandle_t handle )
```

Returns the file control block for the given handle.

Parameters

handle	Handle of the file

Returns

File control block, or NULL.

2.7.3.11 WsfEfsGetFileName()

```
char* WsfEfsGetFileName (
            wsfEfsHandle_t handle )
```

Get the name of a file.

Parameters

handle	File Handle.

Returns

Filename string of a file.

2.7.3.12 WsfEfsGetFileVersion()

```
char* WsfEfsGetFileVersion (
            wsfEfsHandle_t handle )
```

Get the version of a file.

Parameters

handle	File Handle.

Returns

Version string of a file.

2.7.3.13 WsfEfsGetFileSize()

```
uint32_t WsfEfsGetFileSize (
            wsfEfsHandle_t handle )
```

Get the size of a file.

Parameters

handle	File Handle.

Returns

Size of the file.

2.7.3.14 WsfEfsGetFileMaxSize()

```
uint32_t WsfEfsGetFileMaxSize (
            wsfEfsHandle_t handle )
```

Get the number of bytes of memory reserved for use by a file.

Parameters

handle	File Handle.

Returns

Max size of the file.

2.7.3.15 WsfEfsGetFileType()

```
uint8_t WsfEfsGetFileType (
            wsfEfsHandle_t handle )
```

Get the type of a file.

Parameters

handle	File Handle.

Returns

Type of file (bulk or stream).

2.7.3.16 WsfEfsGetFilePermissions()

```
uint16_t WsfEfsGetFilePermissions (
            wsfEfsHandle_t handle )
```

Get the permissions of a file.

Parameters

handle	File Handle.

Returns

Permissions of the file.

2.7.3.17 WsfEfsMediaSpecificCommand()

```
uint8_t WsfEfsMediaSpecificCommand (
            wsfEfsHandle_t handle,
            uint8_t cmd,
            uint32_t param )
```

Execute a media specific command on a file.

Parameters

handle	File Handle.
cmd	Media specific command identifier.
param	Command specific parameter.

Returns

Status of the operation.

2.8 WSF Heap API

Functions

- uint32_t WsfHeapCountAvailable (void)

  Get heap available.
- uint32_t WsfHeapCountUsed (void)

  Get heap used.
- void WsfHeapAlloc (uint32_t size)

  Reserve heap memory.
- void ∗ WsfHeapGetFreeStartAddress (void)

  Get next available heap memory.

2.8.1 Detailed Description

2.8.2 Function Documentation

2.8.2.1 WsfHeapCountAvailable()

```
uint32_t WsfHeapCountAvailable (
            void )
```

Get heap available.

Returns

Number of bytes of heap memory available.

2.8.2.2 WsfHeapCountUsed()

```
uint32_t WsfHeapCountUsed (
            void )
```

Get heap used.

Returns

Number of bytes of heap memory used.

2.8.2.3 WsfHeapAlloc()

```
void WsfHeapAlloc (
            uint32_t size )
```

Reserve heap memory.

Parameters

size	Number of bytes of heap memory used.

2.8.2.4 WsfHeapGetFreeStartAddress()

```
void* WsfHeapGetFreeStartAddress (
            void  )
```

Get next available heap memory.

Returns

Address of the start of heap memory.

2.9 WSF Math API

Macros

- #define WSF_MIN(a, b) ((a) < (b) ? (a) : (b))

  Returns the minimum of two values.
- #define WSF_MAX(a, b) ((a) > (b) ? (a) : (b))

  Returns the maximum of two values.

2.9.1 Detailed Description

2.10 WSF Message API

Functions

- bool_t CheckWsfMsgDataAlloc (uint16_t len, uint8_t tailroom)

  Verify whether a data buffer with required length is available to send a message buffer with WsfMsgSend().
- void ∗ WsfMsgDataAlloc (uint16_t len, uint8_t tailroom)

  Allocate a data message buffer to be sent with WsfMsgSend().
- bool_t CheckWsfMsgAlloc (uint16_t len)

  Verify whether a buffer with required length is available to send a message buffer with WsfMsgSend().
- void ∗ WsfMsgAlloc (uint16_t len)

  Allocate a message buffer to be sent with WsfMsgSend().
- void WsfMsgFree (void ∗pMsg)

  Free a message buffer allocated with WsfMsgAlloc().
- void WsfMsgSend (wsfHandlerId_t handlerId, void ∗pMsg)

  Send a message to an event handler.
- void WsfMsgEnq (wsfQueue_t ∗pQueue, wsfHandlerId_t handlerId, void ∗pMsg)

  Enqueue a message.
- void ∗ WsfMsgDeq (wsfQueue_t ∗pQueue, wsfHandlerId_t ∗pHandlerId)

  Dequeue a message.
- void ∗ WsfMsgPeek (wsfQueue_t ∗pQueue, wsfHandlerId_t ∗pHandlerId)

  Get the next message without removing it from the queue.
- void ∗ WsfMsgNPeek (wsfQueue_t ∗pQueue, uint8_t n, wsfHandlerId_t ∗pHandlerId)

  Get the Nth message without removing it from the queue.

2.10.1 Detailed Description

2.10.2 Function Documentation

2.10.2.1 CheckWsfMsgDataAlloc()

```
bool_t CheckWsfMsgDataAlloc (
             uint16_t len,
             uint8_t tailroom )
```

Verify whether a data buffer with required length is available to send a message buffer with WsfMsgSend().

Parameters

len	Message length in bytes.
tailroom	Tailroom length in bytes.

Returns

True if buffer is available, false if buffer allocation is not possible.

2.10.2.2 WsfMsgDataAlloc()

```
void* WsfMsgDataAlloc (
            uint16_t len,
            uint8_t tailroom )
```

Allocate a data message buffer to be sent with WsfMsgSend().

Parameters

len	Message length in bytes.
tailroom	Tailroom length in bytes.

Returns

Pointer to data message buffer or NULL if allocation failed.

2.10.2.3 CheckWsfMsgAlloc()

```
bool_t CheckWsfMsgAlloc (
            uint16_t len )
```

Verify whether a buffer with required length is available to send a message buffer with WsfMsgSend().

Parameters

len	Message length in bytes.

Returns

True if buffer is available, false if buffer allocation is not possible.

2.10.2.4 WsfMsgAlloc()

```
void* WsfMsgAlloc (
            uint16_t len )
```

Allocate a message buffer to be sent with WsfMsgSend().

Parameters

len	Message length in bytes.

Returns

Pointer to message buffer or NULL if allocation failed.

2.10.2.5 WsfMsgFree()

```
void WsfMsgFree (
             void * pMsg )
```

Free a message buffer allocated with WsfMsgAlloc().

Parameters

pMsg	Pointer to message buffer.

2.10.2.6 WsfMsgSend()

```
void WsfMsgSend (
             wsfHandlerId_t handlerId,
             void * pMsg )
```

Send a message to an event handler.

Parameters

handler↩ Id	Event handler ID.
pMsg	Pointer to message buffer.

2.10.2.7 WsfMsgEnq()

```
void WsfMsgEnq (
             wsfQueue_t * pQueue,
             wsfHandlerId_t handlerId,
             void * pMsg )
```

Enqueue a message.

Parameters

pQueue	Pointer to queue.
handler↩ Id	Set message handler ID to this value.
pMsg	Pointer to message buffer.

2.10.2.8 WsfMsgDeq()

```
void* WsfMsgDeq (
            wsfQueue_t * pQueue,
            wsfHandlerId_t * pHandlerId )
```

Dequeue a message.

Parameters

pQueue	Pointer to queue.
p↩ HandlerId	Handler ID of returned message; this is a return parameter.

Returns

Pointer to message that has been dequeued or NULL if queue is empty.

2.10.2.9 WsfMsgPeek()

```
void* WsfMsgPeek (
            wsfQueue_t * pQueue,
            wsfHandlerId_t * pHandlerId )
```

Get the next message without removing it from the queue.

Parameters

pQueue	Pointer to queue.
p↩ HandlerId	Handler ID of returned message; this is a return parameter.

Returns

Pointer to the next message on the queue or NULL if queue is empty.

2.10.2.10 WsfMsgNPeek()

```
void* WsfMsgNPeek (
            wsfQueue_t * pQueue,
            uint8_t n,
            wsfHandlerId_t * pHandlerId )
```

Get the Nth message without removing it from the queue.

Parameters

pQueue	Pointer to queue.
n	Nth item from the top (0 = top element).
p←↩ HandlerId	Handler ID of returned message; this is a return parameter.

Returns

Pointer to the next message on the queue or NULL if queue is empty.

2.11 WSF NVM API

Typedefs

- typedef void(∗ WsfNvmCompEvent_t) (bool_t status)

  Operation completion callback.

Functions

- static uint64_t WsfNvmConvertChar8to64Bit (char ∗charId)

  Read data.
- void WsfNvmInit (void)

  Initialize the WSF NVM.
- bool_t WsfNvmReadData (uint64_t id, uint8_t ∗pData, uint16_t len, WsfNvmCompEvent_t compCback)

  Read data.
- bool_t WsfNvmWriteData (uint64_t id, const uint8_t ∗pData, uint16_t len, WsfNvmCompEvent_t compCback)

  Write data.
- bool_t WsfNvmEraseData (uint64_t id, WsfNvmCompEvent_t compCback)

  Erase data.
- void WsfNvmEraseDataAll (WsfNvmCompEvent_t compCback)

  Erase all data located in NVM storage.

2.11.1 Detailed Description

2.11.2 Function Documentation

2.11.2.1 WsfNvmConvertChar8to64Bit()

```
static uint64_t WsfNvmConvertChar8to64Bit (
            char ∗ charId )  [inline], [static]
```

Read data.

Parameters

char← Id	charactor arrary for NVM ID.

Returns

if Read NVM successfully.

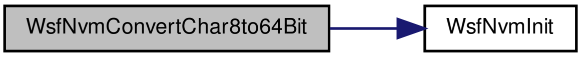
Definition at line 77 of file wsf_nvm.h.

References WsfNvmInit().

```
{
  uint64_t retValue = 0;

  for (uint8_t i = 0; i < 8; i++)
  {
    retValue |= ((uint64_t)charId[7 - i]) << (8*i);
  }
  return retValue;
}
```

Here is the call graph for this function:



2.11.2.2 WsfNvmReadData()

```
bool_t WsfNvmReadData (
            uint64_t id,
            uint8_t * pData,
            uint16_t len,
            WsfNvmCompEvent_t compCback )
```

Read data.

Parameters

id	Read ID.
pData	Buffer to read to.
len	Data length to read.
compCback	Read callback.

Returns

TRUE if NVM operation is successful, FALSE otherwise.

2.11.2.3 WsfNvmWriteData()

```
bool_t WsfNvmWriteData (
            uint64_t id,
            const uint8_t * pData,
            uint16_t len,
            WsfNvmCompEvent_t compCback )
```

Write data.

Parameters

id	Write ID.
pData	Buffer to write.
len	Data length to write.
compCback	Write callback.

Returns

TRUE if NVM operation is successful, FALSE otherwise.

2.11.2.4 WsfNvmEraseData()

```
bool_t WsfNvmEraseData (
             uint64_t id,
             WsfNvmCompEvent_t compCback )
```

Erase data.

Parameters

id	Erase ID.
compCback	Write callback.

Returns

TRUE if NVM operation is successful, FALSE otherwise.

2.11.2.5 WsfNvmEraseDataAll()

```
void WsfNvmEraseDataAll (
             WsfNvmCompEvent_t compCback )
```

Erase all data located in NVM storage.

Parameters

compCback	Erase callback.

Note

Security Risk Warning. NVM storage could be shared by multiple Apps.

2.12 WSF OS API

Data Structures

- struct wsfMsgHdr_t

    Common message structure passed to event handler.

Macros

- #define WSF_OS_DIAG FALSE

    OS Diagnostics.
- #define WSF_TASK_FROM_ID(handlerID) (((handlerID) >> 4) & 0x0F)

    Derive task from handler ID.
- #define WSF_HANDLER_FROM_ID(handlerID) ((handlerID) & 0x0F)

    Derive handler from handler ID.
- #define WSF_INVALID_TASK_ID 0xFF

    Invalid Task Identifier.
- #define WSF_OS_GET_ACTIVE_HANDLER_ID() WSF_INVALID_TASK_ID

    Get Diagnostic Task Identifier.

Typedefs

- typedef uint8_t wsfHandlerId_t

    Event handler ID data type.
- typedef uint16_t wsfEventMask_t

    Event handler event mask data type.
- typedef wsfHandlerId_t wsfTaskId_t

    Task ID data type.
- typedef uint8_t wsfTaskEvent_t

    Task event mask data type.
- typedef bool_t(∗ WsfOsIdleHandler_t) (void)

    Idle check function.
- typedef void(∗ wsfEventHandler_t) (wsfEventMask_t event, wsfMsgHdr_t ∗pMsg)

    Event handler callback function.

Functions

- void WsfSetEvent (wsfHandlerId_t handlerId, wsfEventMask_t event)

    Set an event for an event handler.
- void WsfTaskLock (void)

    Lock task scheduling.
- void WsfTaskUnlock (void)

    Unlock task scheduling.
- void WsfTaskSetReady (wsfHandlerId_t handlerId, wsfTaskEvent_t event)

    Set the task used by the given handler as ready to run.
- wsfQueue_t ∗ WsfTaskMsgQueue (wsfHandlerId_t handlerId)

    Return the task message queue used by the given handler.
- wsfHandlerId_t WsfOsSetNextHandler (wsfEventHandler_t handler)

Set the next WSF handler function in the WSF OS handler array. This function should only be called as part of the OS initialization procedure.
- void WsfOsInit (void)

  Initialize OS control structure.
- bool_t WsfOsReadyToSleep (void)

  Check if WSF is ready to sleep.
- void WsfOsDispatcher (void)

  Event dispatched. Designed to be called repeatedly from infinite loop.
- void WsfOsEnterMainLoop (void)

  OS starts main loop.
- void WsfOsRegisterIdleTask (WsfOsIdleHandler_t func)

  Register service check functions.

Variables

- wsfHandlerId_t WsfActiveHandler

  Diagnostic Task Identifier.

WSF Task Events

- #define WSF_MSG_QUEUE_EVENT 0x01

  Message queued for event handler.
- #define WSF_TIMER_EVENT 0x02

  Timer expired for event handler.
- #define WSF_HANDLER_EVENT 0x04

  Event set for event handler.

2.12.1 Detailed Description

2.12.2 Typedef Documentation

2.12.2.1 wsfEventHandler_t

```
typedef void(* wsfEventHandler_t) (wsfEventMask_t event, wsfMsgHdr_t *pMsg)
```

Event handler callback function.

Parameters

event	Mask of events set for the event handler.
pMsg	Pointer to message for the event handler.

Definition at line 151 of file wsf_os.h.

2.12.3 Function Documentation

2.12.3.1 WsfSetEvent()

```
void WsfSetEvent (
            wsfHandlerId_t handlerId,
            wsfEventMask_t event )
```

Set an event for an event handler.

Parameters

handler↩ Id	Handler ID.
event	Event or events to set.

2.12.3.2 WsfTaskSetReady()

```
void WsfTaskSetReady (
            wsfHandlerId_t handlerId,
            wsfTaskEvent_t event )
```

Set the task used by the given handler as ready to run.

Parameters

handler↩ Id	Event handler ID.
event	Task event mask.

2.12.3.3 WsfTaskMsgQueue()

```
wsfQueue_t* WsfTaskMsgQueue (
            wsfHandlerId_t handlerId )
```

Return the task message queue used by the given handler.

Parameters

handler↩ Id	Event handler ID.

Returns

Task message queue.

2.12.3.4 WsfOsSetNextHandler()

```
wsfHandlerId_t WsfOsSetNextHandler (
             wsfEventHandler_t handler )
```

Set the next WSF handler function in the WSF OS handler array. This function should only be called as part of the OS initialization procedure.

Parameters

handler	WSF handler function.

Returns

WSF handler ID for this handler.

2.12.3.5 WsfOsInit()

```
void WsfOsInit (
             void )
```

Initialize OS control structure.

Returns

None.

2.12.3.6 WsfOsReadyToSleep()

```
bool_t WsfOsReadyToSleep (
             void )
```

Check if WSF is ready to sleep.

Returns

Return TRUE if there are no pending WSF task events set, FALSE otherwise.

2.12.3.7 WsfOsDispatcher()

```
void WsfOsDispatcher (
             void  )
```

Event dispatched. Designed to be called repeatedly from infinite loop.

Returns

None.

2.12.3.8 WsfOsRegisterIdleTask()

```
void WsfOsRegisterIdleTask (
             WsfOsIdleHandler_t func )
```

Register service check functions.

Parameters

func	Service function.

2.13 WSF Queue API

Data Structures

- struct wsfQueue_t

  Queue structure.

Macros

- #define WSF_QUEUE_INIT(pQueue) {(pQueue)->pHead = NULL; (pQueue)->pTail = NULL;}

  Initialize a queue.

Functions

- void WsfQueueEnq (wsfQueue_t ∗pQueue, void ∗pElem)

  Enqueue an element to the tail of a queue.
- void ∗ WsfQueueDeq (wsfQueue_t ∗pQueue)

  Dequeue an element from the head of a queue.
- void WsfQueuePush (wsfQueue_t ∗pQueue, void ∗pElem)

  Push an element to the head of a queue.
- void WsfQueueInsert (wsfQueue_t ∗pQueue, void ∗pElem, void ∗pPrev)

  Insert an element into a queue. This function is typically used when iterating over a queue.
- void WsfQueueRemove (wsfQueue_t ∗pQueue, void ∗pElem, void ∗pPrev)

  Remove an element from a queue. This function is typically used when iterating over a queue.
- uint16_t WsfQueueCount (wsfQueue_t ∗pQueue)

  Count the number of elements in a queue.
- bool_t WsfQueueEmpty (wsfQueue_t ∗pQueue)

  Return TRUE if queue is empty.
- bool_t WsfIsQueueDepthOne (wsfQueue_t ∗pQueue)

  Check for a queue depth of 1 element.

2.13.1 Detailed Description

2.13.2 Function Documentation

2.13.2.1 WsfQueueEnq()

```
void WsfQueueEnq (
            wsfQueue_t ∗ pQueue,
            void ∗ pElem )
```

Enqueue an element to the tail of a queue.

Parameters

pQueue	Pointer to queue.
pElem	Pointer to element.

2.13.2.2 WsfQueueDeq()

```
void* WsfQueueDeq (
            wsfQueue_t * pQueue )
```

Dequeue an element from the head of a queue.

Parameters

pQueue	Pointer to queue.

Returns

Pointer to element that has been dequeued or NULL if queue is empty.

2.13.2.3 WsfQueuePush()

```
void WsfQueuePush (
            wsfQueue_t * pQueue,
            void * pElem )
```

Push an element to the head of a queue.

Parameters

pQueue	Pointer to queue.
pElem	Pointer to element.

2.13.2.4 WsfQueueInsert()

```
void WsfQueueInsert (
            wsfQueue_t * pQueue,
            void * pElem,
            void * pPrev )
```

Insert an element into a queue. This function is typically used when iterating over a queue.

Parameters

pQueue	Pointer to queue.
pElem	Pointer to element to be inserted.
pPrev	Pointer to previous element in the queue before element to be inserted. Note: set pPrev to NULL if pElem is first element in queue.

Returns

None.

2.13.2.5 WsfQueueRemove()

```
void WsfQueueRemove (
            wsfQueue_t * pQueue,
            void * pElem,
            void * pPrev )
```

Remove an element from a queue. This function is typically used when iterating over a queue.

Parameters

pQueue	Pointer to queue.
pElem	Pointer to element to be removed.
pPrev	Pointer to previous element in the queue before element to be removed.

2.13.2.6 WsfQueueCount()

```
uint16_t WsfQueueCount (
            wsfQueue_t * pQueue )
```

Count the number of elements in a queue.

Parameters

pQueue	Pointer to queue.

Returns

Number of elements in queue.

2.13.2.7 WsfQueueEmpty()

```
bool_t WsfQueueEmpty (
            wsfQueue_t * pQueue )
```

Return TRUE if queue is empty.

Parameters

pQueue	Pointer to queue.

Returns

TRUE if queue is empty, FALSE otherwise.

2.13.2.8 WsfIsQueueDepthOne()

```
bool_t WsfIsQueueDepthOne (
            wsfQueue_t * pQueue )
```

Check for a queue depth of 1 element.

Parameters

pQueue	Queue.

Returns

TRUE if Queue only has 1 element, FALSE otherwise.

2.14 WSF Timer API

Data Structures

- struct wsfTimer_t

  Timer structure.

Macros

- #define WSF_MS_PER_TICK 1

  Default milliseconds per tick rate.

Typedefs

- typedef uint32_t wsfTimerTicks_t

  Timer ticks data type.

Functions

- void WsfTimerInit (void)

  Initialize the timer service. This function should only be called once upon system initialization.
- void WsfTimerStartSec (wsfTimer_t ∗pTimer, wsfTimerTicks_t sec)

  Start a timer in units of seconds. Before this function is called parameter pTimer->handlerId must be set to the event handler for this timer and parameter pTimer->msg must be set to any application-defined timer event parameters.
- void WsfTimerStartMs (wsfTimer_t ∗pTimer, wsfTimerTicks_t ms)

  Start a timer in units of milliseconds.
- void WsfTimerStop (wsfTimer_t ∗pTimer)

  Stop a timer.
- void WsfTimerUpdate (wsfTimerTicks_t ticks)

  Update the timer service with the number of elapsed ticks. This function is typically called only from timer porting code.
- wsfTimerTicks_t WsfTimerNextExpiration (bool_t ∗pTimerRunning)

  Return the number of ticks until the next timer expiration. Note that this function can return zero even if a timer is running, indicating a timer has expired but has not yet been serviced.
- wsfTimer_t ∗ WsfTimerServiceExpired (wsfTaskId_t taskId)

  Service expired timers for the given task. This function is typically called only WSF OS porting code.

2.14.1 Detailed Description

2.14.2 Function Documentation

2.14.2.1 WsfTimerStartSec()

```
void WsfTimerStartSec (
             wsfTimer_t * pTimer,
             wsfTimerTicks_t sec )
```

Start a timer in units of seconds. Before this function is called parameter pTimer->handlerId must be set to the event handler for this timer and parameter pTimer->msg must be set to any application-defined timer event parameters.

Parameters

pTimer	Pointer to timer.
sec	Seconds until expiration.

2.14.2.2 WsfTimerStartMs()

```
void WsfTimerStartMs (
             wsfTimer_t * pTimer,
             wsfTimerTicks_t ms )
```

Start a timer in units of milliseconds.

Parameters

pTimer	Pointer to timer.
ms	Milliseconds until expiration.

2.14.2.3 WsfTimerStop()

```
void WsfTimerStop (
             wsfTimer_t * pTimer )
```

Stop a timer.

Parameters

pTimer	Pointer to timer.

2.14.2.4 WsfTimerUpdate()

```
void WsfTimerUpdate (
             wsfTimerTicks_t ticks )
```

Update the timer service with the number of elapsed ticks. This function is typically called only from timer porting code.

Parameters

ticks	Number of ticks since last update.

2.14.2.5 WsfTimerNextExpiration()

```
wsfTimerTicks_t WsfTimerNextExpiration (
            bool_t * pTimerRunning )
```

Return the number of ticks until the next timer expiration. Note that this function can return zero even if a timer is running, indicating a timer has expired but has not yet been serviced.

Parameters

pTimerRunning	Returns TRUE if a timer is running, FALSE if no timers running.

Returns

The number of ticks until the next timer expiration.

2.14.2.6 WsfTimerServiceExpired()

```
wsfTimer_t* WsfTimerServiceExpired (
            wsfTaskId_t taskId )
```

Service expired timers for the given task. This function is typically called only WSF OS porting code.

Parameters

taskId	OS Task ID of task servicing timers.

Returns

Pointer to next expired timer or NULL if there are no expired timers.

2.15 WSF Trace API

Macros

- #define WSF_TRACE_ENABLED FALSE

  Trace enable flag (default is disabled, override with compile-time directive).
- #define WSF_TOKEN_ENABLED FALSE

  Tokenized tracing enable flag (default is disabled, override with compile-time directive).
- #define LL_TRACE_ENABLED FALSE

  Trace enabled for controller.
- #define **AUD_TRACE_ENABLED** FALSE
- #define WSF_TRACE_INFO0(msg)

  0 argument WSF info trace.
- #define WSF_TRACE_INFO1(msg, var1)

  1 argument WSF info trace.
- #define WSF_TRACE_INFO2(msg, var1, var2)

  2 argument WSF info trace.
- #define WSF_TRACE_INFO3(msg, var1, var2, var3)

  3 argument WSF info trace.
- #define WSF_TRACE_INFO4(msg, var1, var2, var3, var4)

  4 argument WSF info trace.
- #define WSF_TRACE_INFO5(msg, var1, var2, var3, var4, var5)

  5 argument WSF info trace.
- #define WSF_TRACE_INFO6(msg, var1, var2, var3, var4, var5, var6)

  6 argument WSF info trace.
- #define WSF_TRACE_WARN0(msg) WSF_TRACE0("WSF", "WARN", msg)

  0 argument WSF warning trace.
- #define WSF_TRACE_WARN1(msg, var1) WSF_TRACE1("WSF", "WARN", msg, var1)

  1 argument WSF warning trace.
- #define WSF_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("WSF", "WARN", msg, var1, var2)

  2 argument WSF warning trace.
- #define WSF_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("WSF", "WARN", msg, var1, var2, var3)

  3 argument WSF warning trace.
- #define WSF_TRACE_WARN4(msg, var1, var2, var3, var4) WSF_TRACE4("WSF", "WARN", msg, var1, var2, var3, var4)

  5 argument WSF warning trace.
- #define WSF_TRACE_WARN5(msg, var1, var2, var3, var4, var5) WSF_TRACE5("WSF", "WARN", msg, var1, var2, var3, var4, var5)

  5 argument WSF warning trace.
- #define WSF_TRACE_WARN6(msg, var1, var2, var3, var4, var5, var6) WSF_TRACE6("WSF", "WARN", msg, var1, var2, var3, var4, var5, var6)

  6 argument WSF warning trace.
- #define WSF_TRACE_ERR0(msg) WSF_TRACE0("WSF", "ERR", msg)

  0 argument WSF error trace.
- #define WSF_TRACE_ERR1(msg, var1) WSF_TRACE1("WSF", "ERR", msg, var1)

  1 argument WSF error trace.
- #define WSF_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("WSF", "ERR", msg, var1, var2)

  2 argument WSF error trace.
- #define WSF_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("WSF", "ERR", msg, var1, var2, var3)

  3 argument WSF error trace.

- #define WSF_TRACE_ERR4(msg, var1, var2, var3, var4) WSF_TRACE4("WSF", "ERR", msg, var1, var2, var3, var4)

  4 argument WSF error trace.
- #define WSF_TRACE_ERR5(msg, var1, var2, var3, var4, var5) WSF_TRACE5("WSF", "ERR", msg, var1, var2, var3, var4, var5)

  5 argument WSF error trace.
- #define WSF_TRACE_ERR6(msg, var1, var2, var3, var4, var5, var6) WSF_TRACE6("WSF", "ERR", msg, var1, var2, var3, var4, var5, var6)

  5 argument WSF error trace.
- #define WSF_TRACE_ALLOC0(msg)

  0 argument WSF buffer allocation trace.
- #define WSF_TRACE_ALLOC1(msg, var1)

  1 argument WSF buffer allocation trace.
- #define WSF_TRACE_ALLOC2(msg, var1, var2)

  2 argument WSF buffer allocation trace.
- #define WSF_TRACE_ALLOC3(msg, var1, var2, var3)

  3 argument WSF buffer allocation trace.
- #define WSF_TRACE_FREE0(msg)

  0 argument WSF buffer free trace.
- #define WSF_TRACE_FREE1(msg, var1)

  1 argument WSF buffer free trace.
- #define WSF_TRACE_FREE2(msg, var1, var2)

  2 argument WSF buffer free trace.
- #define WSF_TRACE_FREE3(msg, var1, var2, var3)

  3 argument WSF buffer free trace.
- #define WSF_TRACE_MSG0(msg)

  0 argument WSF message trace.
- #define WSF_TRACE_MSG1(msg, var1)

  1 argument WSF message trace.
- #define WSF_TRACE_MSG2(msg, var1, var2)

  2 argument WSF message trace.
- #define WSF_TRACE_MSG3(msg, var1, var2, var3)

  3 argument WSF message trace.
- #define HCI_TRACE_INFO0(msg)

  0 argument HCI info trace.
- #define HCI_TRACE_INFO1(msg, var1)

  1 argument HCI info trace.
- #define HCI_TRACE_INFO2(msg, var1, var2)

  2 argument HCI info trace.
- #define HCI_TRACE_INFO3(msg, var1, var2, var3)

  3 argument HCI info trace.
- #define HCI_TRACE_WARN0(msg) WSF_TRACE0("HCI", "WARN", msg)

  0 argument HCI warning trace.
- #define HCI_TRACE_WARN1(msg, var1) WSF_TRACE1("HCI", "WARN", msg, var1)

  1 argument HCI warning trace.
- #define HCI_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("HCI", "WARN", msg, var1, var2)

  2 argument HCI warning trace.
- #define HCI_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("HCI", "WARN", msg, var1, var2, var3)

  3 argument HCI warning trace.
- #define HCI_TRACE_ERR0(msg) WSF_TRACE0("HCI", "ERR", msg)

  0 argument HCI error trace.

- #define HCI_TRACE_ERR1(msg, var1) WSF_TRACE1("HCI", "ERR", msg, var1)

  1 argument HCI error trace.
- #define HCI_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("HCI", "ERR", msg, var1, var2)

  2 argument HCI error trace.
- #define HCI_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("HCI", "ERR", msg, var1, var2, var3)

  3 argument HCI error trace.
- #define HCI_PDUMP_CMD(len, pBuf)

  HCI PDUMP on command.
- #define HCI_PDUMP_EVT(len, pBuf)

  HCI PDUMP on event.
- #define HCI_PDUMP_TX_ACL(len, pBuf)

  HCI PDUMP on transmitted ACL message.
- #define HCI_PDUMP_RX_ACL(len, pBuf)

  HCI PDUMP on Received ACL message.
- #define HCI_PDUMP_TX_ISO(len, pBuf)

  HCI PDUMP on transmitted ISO message.
- #define HCI_PDUMP_RX_ISO(len, pBuf)

  HCI PDUMP on Received ISO message.
- #define DM_TRACE_INFO0(msg)

  0 argument DM info trace.
- #define DM_TRACE_INFO1(msg, var1)

  1 argument DM info trace.
- #define DM_TRACE_INFO2(msg, var1, var2)

  2 argument DM info trace.
- #define DM_TRACE_INFO3(msg, var1, var2, var3)

  3 argument DM info trace.
- #define DM_TRACE_WARN0(msg) WSF_TRACE0("DM", "WARN", msg)

  0 argument DM warning trace.
- #define DM_TRACE_WARN1(msg, var1) WSF_TRACE1("DM", "WARN", msg, var1)

  1 argument DM warning trace.
- #define DM_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("DM", "WARN", msg, var1, var2)

  2 argument DM warning trace.
- #define DM_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("DM", "WARN", msg, var1, var2, var3)

  3 argument DM warning trace.
- #define DM_TRACE_ERR0(msg) WSF_TRACE0("DM", "ERR", msg)

  0 argument DM error trace.
- #define DM_TRACE_ERR1(msg, var1) WSF_TRACE1("DM", "ERR", msg, var1)

  1 argument DM error trace.
- #define DM_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("DM", "ERR", msg, var1, var2)

  2 argument DM error trace.
- #define DM_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("DM", "ERR", msg, var1, var2, var3)

  3 argument DM error trace.
- #define DM_TRACE_ALLOC0(msg) WSF_TRACE0("DM", "ALLOC", msg)

  0 argument DM buffer allocation trace.
- #define DM_TRACE_ALLOC1(msg, var1) WSF_TRACE1("DM", "ALLOC", msg, var1)

  1 argument DM buffer allocation trace.
- #define DM_TRACE_ALLOC2(msg, var1, var2) WSF_TRACE2("DM", "ALLOC", msg, var1, var2)

  2 argument DM buffer allocation trace.
- #define DM_TRACE_ALLOC3(msg, var1, var2, var3) WSF_TRACE3("DM", "ALLOC", msg, var1, var2, var3)

  3 argument DM buffer allocation trace.
- #define DM_TRACE_FREE0(msg) WSF_TRACE0("DM", "FREE", msg)

0 argument DM buffer free trace.

- #define DM_TRACE_FREE1(msg, var1) WSF_TRACE1("DM", "FREE", msg, var1)

  1 argument DM buffer free trace.
- #define DM_TRACE_FREE2(msg, var1, var2) WSF_TRACE2("DM", "FREE", msg, var1, var2)

  2 argument DM buffer free trace.
- #define DM_TRACE_FREE3(msg, var1, var2, var3) WSF_TRACE3("DM", "FREE", msg, var1, var2, var3)

  3 argument DM buffer free trace.
- #define L2C_TRACE_INFO0(msg)

  0 argument L2C info trace.
- #define L2C_TRACE_INFO1(msg, var1)

  1 argument L2C info trace.
- #define L2C_TRACE_INFO2(msg, var1, var2)

  2 argument L2C info trace.
- #define L2C_TRACE_INFO3(msg, var1, var2, var3)

  3 argument L2C info trace.
- #define L2C_TRACE_WARN0(msg) WSF_TRACE0("L2C", "WARN", msg)

  0 argument L2C warning trace.
- #define L2C_TRACE_WARN1(msg, var1) WSF_TRACE1("L2C", "WARN", msg, var1)

  1 argument L2C warning trace.
- #define L2C_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("L2C", "WARN", msg, var1, var2)

  2 argument L2C warning trace.
- #define L2C_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("L2C", "WARN", msg, var1, var2, var3)

  3 argument L2C warning trace.
- #define L2C_TRACE_ERR0(msg) WSF_TRACE0("L2C", "ERR", msg)

  0 argument L2C error trace.
- #define L2C_TRACE_ERR1(msg, var1) WSF_TRACE1("L2C", "ERR", msg, var1)

  1 argument L2C error trace.
- #define L2C_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("L2C", "ERR", msg, var1, var2)

  2 argument L2C error trace.
- #define L2C_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("L2C", "ERR", msg, var1, var2, var3)

  3 argument L2C error trace.
- #define ATT_TRACE_INFO0(msg)

  0 argument ATT info trace.
- #define ATT_TRACE_INFO1(msg, var1)

  1 argument ATT info trace.
- #define ATT_TRACE_INFO2(msg, var1, var2)

  2 argument ATT info trace.
- #define ATT_TRACE_INFO3(msg, var1, var2, var3)

  3 argument ATT info trace.
- #define ATT_TRACE_WARN0(msg) WSF_TRACE0("ATT", "WARN", msg)

  0 argument ATT warning trace.
- #define ATT_TRACE_WARN1(msg, var1) WSF_TRACE1("ATT", "WARN", msg, var1)

  1 argument ATT warning trace.
- #define ATT_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("ATT", "WARN", msg, var1, var2)

  2 argument ATT warning trace.
- #define ATT_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("ATT", "WARN", msg, var1, var2, var3)

  3 argument ATT warning trace.
- #define ATT_TRACE_ERR0(msg) WSF_TRACE0("ATT", "ERR", msg)

  0 argument ATT error trace.
- #define ATT_TRACE_ERR1(msg, var1) WSF_TRACE1("ATT", "ERR", msg, var1)

  1 argument ATT error trace.

- #define ATT_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("ATT", "ERR", msg, var1, var2)

  2 argument ATT error trace.
- #define ATT_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("ATT", "ERR", msg, var1, var2, var3)

  3 argument ATT error trace.
- #define EATT_TRACE_INFO0(msg) WSF_TRACE0("EATT", "INFO", msg)

  0 argument EATT info trace.
- #define EATT_TRACE_INFO1(msg, var1) WSF_TRACE1("EATT", "INFO", msg, var1)

  1 argument EATT info trace.
- #define EATT_TRACE_INFO2(msg, var1, var2) WSF_TRACE2("EATT", "INFO", msg, var1, var2)

  2 argument EATT info trace.
- #define EATT_TRACE_INFO3(msg, var1, var2, var3) WSF_TRACE3("EATT", "INFO", msg, var1, var2, var3)

  3 argument EATT info trace.
- #define EATT_TRACE_WARN0(msg) WSF_TRACE0("EATT", "WARN", msg)

  0 argument EATT warning trace.
- #define EATT_TRACE_WARN1(msg, var1) WSF_TRACE1("EATT", "WARN", msg, var1)

  1 argument EATT warning trace.
- #define EATT_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("EATT", "WARN", msg, var1, var2)

  2 argument EATT warning trace.
- #define EATT_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("EATT", "WARN", msg, var1, var2, var3)

  3 argument EATT warning trace.
- #define EATT_TRACE_ERR0(msg) WSF_TRACE0("EATT", "ERR", msg)

  0 argument EATT error trace.
- #define EATT_TRACE_ERR1(msg, var1) WSF_TRACE1("EATT", "ERR", msg, var1)

  1 argument EATT error trace.
- #define EATT_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("EATT", "ERR", msg, var1, var2)

  2 argument EATT error trace.
- #define EATT_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("EATT", "ERR", msg, var1, var2, var3)

  3 argument EATT error trace.
- #define SMP_TRACE_INFO0(msg)

  0 argument SMP info trace.
- #define SMP_TRACE_INFO1(msg, var1)

  1 argument SMP info trace.
- #define SMP_TRACE_INFO2(msg, var1, var2)

  2 argument SMP info trace.
- #define SMP_TRACE_INFO3(msg, var1, var2, var3)

  3 argument SMP info trace.
- #define SMP_TRACE_WARN0(msg) WSF_TRACE0("SMP", "WARN", msg)

  0 argument SMP warning trace.
- #define SMP_TRACE_WARN1(msg, var1) WSF_TRACE1("SMP", "WARN", msg, var1)

  1 argument SMP warning trace.
- #define SMP_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("SMP", "WARN", msg, var1, var2)

  2 argument SMP warning trace.
- #define SMP_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("SMP", "WARN", msg, var1, var2, var3)

  3 argument SMP warning trace.
- #define SMP_TRACE_ERR0(msg) WSF_TRACE0("SMP", "ERR", msg)

  0 argument SMP error trace.
- #define SMP_TRACE_ERR1(msg, var1) WSF_TRACE1("SMP", "ERR", msg, var1)

  1 argument SMP error trace.
- #define SMP_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("SMP", "ERR", msg, var1, var2)

2 argument SMP error trace.

- #define SMP_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("SMP", "ERR", msg, var1, var2, var3)

  3 argument SMP error trace.
- #define **SMP_TRACE_BUF**(msg)
- #define **APP_TRACE_DEBUG**(msg, ...)
- #define APP_TRACE_INFO(msg, ...)

  Variadic argument App info trace.
- #define APP_TRACE_INFO0(msg)

  0 argument App info trace.
- #define APP_TRACE_INFO1(msg, var1)

  1 argument App info trace.
- #define APP_TRACE_INFO2(msg, var1, var2)

  2 argument App info trace.
- #define APP_TRACE_INFO3(msg, var1, var2, var3)

  3 argument App info trace.
- #define APP_TRACE_INFO4(msg, var1, var2, var3, var4)

  4 argument App info trace.
- #define APP_TRACE_INFO5(msg, var1, var2, var3, var4, var5)

  5 argument App info trace.
- #define APP_TRACE_INFO6(msg, var1, var2, var3, var4, var5, var6)

  6 argument App info trace.
- #define APP_TRACE_INFO7(msg, var1, var2, var3, var4, var5, var6, var7)

  7 argument App info trace.
- #define APP_TRACE_INFO8(msg, var1, var2, var3, var4, var5, var6, var7, var8)

  8 argument App info trace.
- #define APP_TRACE_INFO9(msg, var1, var2, var3, var4, var5, var6, var7, var8, var9)

  9 argument App info trace.
- #define APP_TRACE_INFO12(msg, var1, var2, var3, var4, var5, var6, var7, var8, var9, var10, var11, var12)

  12 argument App info trace.
- #define APP_TRACE_WARN(msg, ...) WSF_TRACE("APP", "WARN", msg, ##__VA_ARGS__)

  Variadic argument App warning trace.
- #define APP_TRACE_WARN0(msg) WSF_TRACE0("APP", "WARN", msg)

  0 argument App warning trace.
- #define APP_TRACE_WARN1(msg, var1) WSF_TRACE1("APP", "WARN", msg, var1)

  1 argument App warning trace.
- #define APP_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("APP", "WARN", msg, var1, var2)

  2 argument App warning trace.
- #define APP_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("APP", "WARN", msg, var1, var2, var3)

  3 argument App warning trace.
- #define APP_TRACE_ERR(msg, ...) WSF_TRACE("APP", "ERR", msg, ##__VA_ARGS__)

  Variadic argument App error trace.
- #define APP_TRACE_ERR0(msg) WSF_TRACE0("APP", "ERR", msg)

  0 argument App error trace.
- #define APP_TRACE_ERR1(msg, var1) WSF_TRACE1("APP", "ERR", msg, var1)

  1 argument App error trace.
- #define APP_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("APP", "ERR", msg, var1, var2)

  2 argument App error trace.
- #define APP_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("APP", "ERR", msg, var1, var2, var3)

  3 argument App error trace.
- #define **LL_TRACE_INFO0**(msg)

- #define LL_TRACE_INFO1(msg, var1)

  1 argument LL info trace.
- #define LL_TRACE_INFO2(msg, var1, var2)

  2 argument LL info trace.
- #define LL_TRACE_INFO3(msg, var1, var2, var3)

  3 argument LL info trace.
- #define LL_TRACE_WARN0(msg)

  0 argument LL warning trace.
- #define LL_TRACE_WARN1(msg, var1)

  1 argument LL warning trace.
- #define LL_TRACE_WARN2(msg, var1, var2)

  2 argument LL warning trace.
- #define LL_TRACE_WARN3(msg, var1, var2, var3)

  3 argument LL warning trace.
- #define LL_TRACE_ERR0(msg)

  0 argument LL error trace.
- #define LL_TRACE_ERR1(msg, var1)

  1 argument LL error trace.
- #define LL_TRACE_ERR2(msg, var1, var2)

  2 argument LL error trace.
- #define LL_TRACE_ERR3(msg, var1, var2, var3)

  3 argument LL error trace.
- #define AUD_TRACE_INFO0(msg)

  0 argument audio info trace.
- #define AUD_TRACE_INFO1(msg, var1)

  1 argument audio info trace.
- #define AUD_TRACE_INFO2(msg, var1, var2)

  2 argument audio info trace.
- #define AUD_TRACE_INFO3(msg, var1, var2, var3)

  3 argument audio info trace.
- #define AUD_TRACE_WARN0(msg)

  0 argument audio warning trace.
- #define AUD_TRACE_WARN1(msg, var1)

  1 argument audio warning trace.
- #define AUD_TRACE_WARN2(msg, var1, var2)

  2 argument audio warning trace.
- #define AUD_TRACE_WARN3(msg, var1, var2, var3)

  3 argument audio warning trace.
- #define AUD_TRACE_ERR0(msg)

  0 argument audio error trace.
- #define AUD_TRACE_ERR1(msg, var1)

  1 argument audio error trace.
- #define AUD_TRACE_ERR2(msg, var1, var2)

  2 argument audio error trace.
- #define AUD_TRACE_ERR3(msg, var1, var2, var3)

  3 argument audio error trace.
- #define **MESH_TRACE_DEBUG**(msg, ...)
- #define MESH_TRACE_INFO(msg, ...)

  Variadic argument MESH info trace.
- #define MESH_TRACE_INFO0(msg)

  0 argument MESH info trace.

- #define MESH_TRACE_INFO1(msg, var1)

  1 argument MESH info trace.
- #define MESH_TRACE_INFO2(msg, var1, var2)

  2 argument MESH info trace.
- #define MESH_TRACE_INFO3(msg, var1, var2, var3)

  3 argument MESH info trace.
- #define MESH_TRACE_WARN(msg, ...) WSF_TRACE("MESH", "WARN", msg, ##__VA_ARGS__)

  Variadic argument MESH warning trace.
- #define MESH_TRACE_WARN0(msg) WSF_TRACE0("MESH", "WARN", msg)

  0 argument MESH warning trace.
- #define MESH_TRACE_WARN1(msg, var1) WSF_TRACE1("MESH", "WARN", msg, var1)

  1 argument MESH warning trace.
- #define MESH_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("MESH", "WARN", msg, var1, var2)

  2 argument MESH warning trace.
- #define MESH_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("MESH", "WARN", msg, var1, var2, var3)

  3 argument MESH warning trace.
- #define MESH_TRACE_ERR(msg, ...) WSF_TRACE("MESH", "ERR", msg, ##__VA_ARGS__)

  Variadic argument MESH warning trace.
- #define MESH_TRACE_ERR0(msg) WSF_TRACE0("MESH", "ERR", msg)

  0 argument MESH error trace.
- #define MESH_TRACE_ERR1(msg, var1) WSF_TRACE1("MESH", "ERR", msg, var1)

  1 argument MESH error trace.
- #define MESH_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("MESH", "ERR", msg, var1, var2)

  2 argument MESH error trace.
- #define MESH_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("MESH", "ERR", msg, var1, var2, var3)

  3 argument MESH error trace.
- #define **MMDL_TRACE_DEBUG**(msg, ...)
- #define MMDL_TRACE_INFO(msg, ...)

  Variadic argument MMDL info trace.
- #define MMDL_TRACE_INFO0(msg)

  0 argument MMDL info trace.
- #define MMDL_TRACE_INFO1(msg, var1)

  1 argument MMDL info trace.
- #define MMDL_TRACE_INFO2(msg, var1, var2)

  2 argument MMDL info trace.
- #define MMDL_TRACE_INFO3(msg, var1, var2, var3)

  3 argument MMDL info trace.
- #define MMDL_TRACE_WARN(msg, ...) WSF_TRACE("MMDL", "WARN", msg, ##__VA_ARGS__)

  Variadic argument MMDL info trace.
- #define MMDL_TRACE_WARN0(msg) WSF_TRACE0("MMDL", "WARN", msg)

  0 argument MMDL warning trace.
- #define MMDL_TRACE_WARN1(msg, var1) WSF_TRACE1("MMDL", "WARN", msg, var1)

  1 argument MMDL warning trace.
- #define MMDL_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("MMDL", "WARN", msg, var1, var2)

  2 argument MMDL warning trace.
- #define MMDL_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("MMDL", "WARN", msg, var1, var2, var3)

  3 argument MMDL warning trace.
- #define MMDL_TRACE_ERR(msg, ...) WSF_TRACE("MMDL", "ERR", msg, ##__VA_ARGS__)

  Variadic argument MMDL info trace.

- #define MMDL_TRACE_ERR0(msg) WSF_TRACE0("MMDL", "ERR", msg)

  0 argument MMDL error trace.
- #define MMDL_TRACE_ERR1(msg, var1) WSF_TRACE1("MMDL", "ERR", msg, var1)

  1 argument MMDL error trace.
- #define MMDL_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("MMDL", "ERR", msg, var1, var2)

  2 argument MMDL error trace.
- #define MMDL_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("MMDL", "ERR", msg, var1, var2, var3)

  3 argument MMDL error trace.
- #define LL_TRACE_ENABLE(ena)

  Enable LL trace.

Typedefs

- typedef bool_t(∗ WsfTraceHandler_t) (const uint8_t ∗pBuf, uint32_t len)

  Token event handler.
- typedef void(∗ WsfBt4TraceCback_t) (const char ∗pStr, va_list args)

  BT4 Platform trace callback.

Functions

- void WsfToken (uint32_t tok, uint32_t var)

  Output tokenized message.
- void WsfTraceEnable (bool_t enable)

  Enable trace messages.
- void WsfTrace (const char ∗pStr,...)

  Output trace message.
- void WsfTraceRegisterHandler (WsfTraceHandler_t traceCback)

  Register trace handler.
- void WsfTraceRegister (WsfBt4TraceCback_t cback)

  Register BT4 platform trace callback function.
- bool_t WsfTokenService (void)

  Service the trace ring buffer.

Trace macros

- #define **WSF_TRACE**(subsys, stat, msg, ...)
- #define **WSF_TRACE0**(subsys, stat, msg)
- #define **WSF_TRACE1**(subsys, stat, msg, var1)
- #define **WSF_TRACE2**(subsys, stat, msg, var1, var2)
- #define **WSF_TRACE3**(subsys, stat, msg, var1, var2, var3)
- #define **WSF_TRACE4**(subsys, stat, msg, var1, var2, var3, var4)
- #define **WSF_TRACE5**(subsys, stat, msg, var1, var2, var3, var4, var5)
- #define **WSF_TRACE6**(subsys, stat, msg, var1, var2, var3, var4, var5, var6)
- #define **WSF_TRACE7**(subsys, stat, msg, var1, var2, var3, var4, var5, var6, var7)
- #define **WSF_TRACE8**(subsys, stat, msg, var1, var2, var3, var4, var5, var6, var7, var8)
- #define **WSF_TRACE9**(subsys, stat, msg, var1, var2, var3, var4, var5, var6, var7, var8, var9)
- #define **WSF_TRACE12**(subsys, stat, msg, var1, var2, var3, var4, var5, var6, var7, var8, var9, var10, var11, var12)
- #define **WSF_TRACE_PRINT**(msg, ...)
- #define **WSF_TRACE_FLUSH**()

2.15.1 Detailed Description

2.15.2 Function Documentation

2.15.2.1 WsfToken()

```
void WsfToken (
            uint32_t tok,
            uint32_t var )
```

Output tokenized message.

Parameters

tok	Token
var	Variable

2.15.2.2 WsfTraceEnable()

```
void WsfTraceEnable (
            bool_t enable )
```

Enable trace messages.

Parameters

enable	TRUE to enable, FALSE to disable

2.15.2.3 WsfTrace()

```
void WsfTrace (
            const char * pStr,
             ... )
```

Output trace message.

Parameters

pStr	Format string Addition parameters variable arguments to the format string.

2.15.2.4 WsfTraceRegisterHandler()

```
void WsfTraceRegisterHandler (
            WsfTraceHandler_t traceCback )
```

Register trace handler.

Parameters

traceCback	Token event handler.

This routine registers a token callback. This callback is called when the next token event is ready to be written to the I/O.

2.15.2.5 WsfTraceRegister()

```
void WsfTraceRegister (
            WsfBt4TraceCback_t cback )
```

Register BT4 platform trace callback function.

Parameters

cback	Callback function

2.15.2.6 WsfTokenService()

```
bool_t WsfTokenService (
            void )
```

Service the trace ring buffer.

Returns

TRUE if trace messages pending, FALSE otherwise.

This routine is called in the main loop for a "push" type trace systems.

2.16 WSF Data Types

Integer Data Types

- #define **bool_t** uint8_t
- #define **FALSE** 0
- #define **TRUE** (!FALSE)
- #define **UINT64_C**(x) x##ULL
- #define **UINT32_C**(x) x##UL
- #define **UINT8_C**(x) (x)

2.16.1 Detailed Description

2.17 Wireless Software Foundation (WSF)

Modules

- WSF Assert API
- WSF Buffer API
- WSF Buffer IO API
- WSF Critical Section API
- WSF Embedded File System API
- WSF Heap API
- WSF Math API
- WSF Message API
- WSF NVM API
- WSF OS API
- WSF Queue API
- WSF Timer API
- WSF Trace API
- WSF Data Types
- WSF Utility API

2.17.1 Detailed Description

2.17.2 Introduction

The Wireless Software Foundation (WSF) is a simple OS wrapper, porting layer, and general-purpose software service used by the Packetcraft embedded software system. The goal of WSF is to stay small and lean, supporting only the basic services required by the system. It consists of the following:

- WSF Buffer IO API
- Portable Data Types
- Dynamic Buffer Service
- Queue Management
- WSF Heap API
- Messages Passing
- Timers
- Event Handlers
- Critical Sections
- Task Schedule Locking
- Assert
- Trace
- Embedded File System
- WSF NVM API
- Utilities

WSF does not define any tasks but defines some interfaces to tasks. It relies on the target OS to implement tasks and manage the timer and event handler services from target OS tasks. WSF can also act as a simple standalone OS in software systems without an existing OS.

2.17.3 Portable Data Types

WSF defines the following portable data types in file wsf_types.h. These data types are used throughout the software system.

See WSF Data Types for more detail.

2.17.4 Dynamic Buffer Service

The WSF buffer management service is a pool-based dynamic memory allocation service. The buffer service interface is defined in file wsf_buf.h.

See WSF Buffer API for more detail.

2.17.4.1 Buffer Tuning

WsfBufGetAllocStats() returns a 128-byte array indexed by length. After a device is operational for some time this indexed array will provide a histogram of buffer allocation that will help with sizing buffer pools. Note that for Bluetooth 4.2 and beyond the max PDU has been increased beyond 128 bytes and therefore may need to be resized accordingly.

The WSF buffer pool statistics allow for the “tuning” of buffer pools to optimize memory use on a device. When WSF_BUF_STATS == TRUE the wsfBufPool_t structure tracks the number of current allocations, the maximum buffer allocations from a specific pool at any given time, and the maximum request length from that pool. By analyzing these statistics against a device’s WSF buffer pool configuration you can re-size and optimize the number of buffers allocated for individual pools.

The maxAlloc gives you the maximum buffers allocated from an individual pool while operational. We recommend ensuring that the number of allocated pools is chosen to ensure that this maximum can be supported, with some additional buffers available for more resource intensive use cases. Note that in the case that a buffer pool overflows, buffers will be utilized from the next highest pool size. This occurs when the maxAlloc reaches the number of allocated buffers for that pool, and will affect the number of allocated buffers for larger pools. Because of this we recommend performing size optimizations starting with the smallest buffer pool.

The maxReqLen is the maximum length buffer requested from that allocation pool. This can be used (along with WsfBufGetAllocStats()) to optimize the size of buffer pools. For example if a 128 byte buffer pool is configured but the maximum size buffer ever allocated from this pool is 64 bytes than you may consider resizing to the maximum so the additional memory required for the larger buffers are not wasted.

2.17.5 Queue Management

The WSF queue service is a general purpose queue service that is used throughout the software system.

The queue service interface is defined in function wsf_queue.h.

See WSF Queue API for more detail.

2.17.6 Messages Passing

The WSF message service is used to pass messages to WSF event handlers.

The WSF message service is defined in file wsf_msg.h.

See WSF Message API for more detail.

2.17.7 Timers

The WSF timer service is used by WSF event handlers.

When a timer expires, the event handler associated with that timer is executed.

The WSF timer service is defined in file wsf_timer.h.

See WSF Timer API for more detail.

2.17.8 Event Handlers

WSF event handlers receive WSF events, messages, and timer expirations from other components in the software system. Event handlers are used by the main protocol subsystems of the stack.

The event handler interface is defined in file wsf_os.h.

See WSF OS API for more detail.

2.17.9 Critical Sections

WSF provides critical section macros that are used in code which might be executed in interrupt context to protect global data.

The critical section interface is defined in file wsf_cs.h.

See WSF Critical Section API for more detail.

2.17.10 Task Schedule Locking

WSF provides interfaces for locking and unlocking task scheduling. This allows for operation in pre-emptive multitasking environments.

The task schedule locking interface is defined in file wsf_os.h.

See WSF OS API for more detail.

2.17.11 Assert

WSF defines assert macros that are used for testing and debugging purposes.

The assert interface is defined in file wsf_assert.h.

See WSF Assert API for more detail.

2.17.12 Trace

WSF defines trace macros that are used throughout the software system for diagnostic purposes. A separate set of trace macros is used for each software subsystem (for example, WSF, HCI, DM, and ATT). This allows trace messages to be compiled in/out for each subsystem. Within each set of subsystem trace macros there are separate macros for different types of trace messages:

- INFO: Informational messages.
- WARN: Warning messages.
- ERR: Error messages.
- ALLOC: Memory or other resource is allocated.
- FREE: Memory or other resource is freed.
- MSG: WSF event handler message is sent.

The trace interface is define in file wsf_trace.h.

See WSF Trace API for more detail.

2.17.13 Embedded File System

The Embedded File System (EFS) provides applications with a simple interface for storing data in non-volatile and volatile memory. The Embedded File System (EFS) works in conjunctions with Packetcraft's proprietary Wireless Data Exchange Profile (WDXS) to provide wireless access to the Embedded File System.

The EFS is defined in file wsf_efs.h.

See WSF Embedded File System API for more detail.

2.17.13.1 Overview

The Embedded File System (EFS) provides a simple API for file storage in ExactLE applications. The EFS provides a common interface for storing files in Flash, EEPROM, RAM, and other forms of volatile and non-volatile storage. The EFS is compatible with the ARN's proprietary Wireless Data Exchange Profile (WDXS) for allowing wireless access to the file system.

The EFS is implemented in two layers: The File Access Layer and The Media Access Layer.

2.17.13.2 File Access Layer API

This section contains the Applications Interface for the File Access Layer. The file Access layer access to files on the file system.

The File Access Layer provides the Wireless Data Exchange Profile (WDXS) and BTLE Applications access to the File System. The File Access Layer provides functions for performing the following operations:

- Add a file.
- Remove a file.

- Erase the contents of a file.
- Get data from a file. The Get operation is similar to a file read.
- Put data into a file. The Put operation is similar to a file write.
- Get file attributes.
- Set file attributes.
- Register media types.

The File Access Layer makes calls into the File Media Layer to access the physical storage medium.

2.17.13.3 File Media Layer API

This section documents the Application Interface for the File Media Layer. The File Media Layer operates a glue layer between the File Access Layer and the drivers that drive the physical storage on the target hardware (Flash, EEPROM, etc.).

The File Media Layer provides an abstraction layer between the File Access Layer and physical storage medium such as RAM, Flash, and EEPROM. Callback functions to perform the following operations on the physical medium are registered with the EFS by the application:

- Initialize the medium
- Write to the medium
- Read from the medium
- Erase data in the medium

2.17.14 Utilities

WSF provides utility functions and macros for a variety of use cases. This includes Bluetooth device address parsing, memory access and mutation, CRC, string formatting and terminal interfacing.

See WSF Utility API for more detail.

Chapter 3

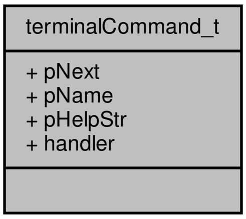
Data Structure Documentation

3.1 terminalCommand_t Struct Reference

Terminal command.

```
#include <terminal.h>
```

Collaboration diagram for terminalCommand_t:



Data Fields

- struct terminalCommand_tag ∗ pNext

  Pointer to next command in list.
- const char ∗ pName

  Name of command.
- const char ∗ pHelpStr

  Help String for command.
- terminalHandler_t handler

  Handler for command.

3.1.1 Detailed Description

Terminal command.

Definition at line 116 of file terminal.h.

The documentation for this struct was generated from the following file:

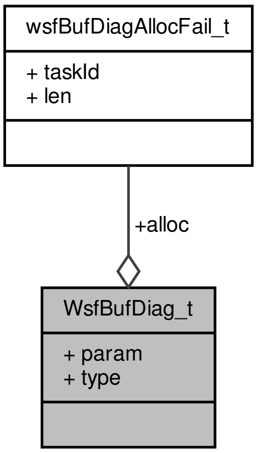
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/terminal.h

3.2 WsfBufDiag_t Struct Reference

WSF buffer diagnostics message.

```
#include <wsf_buf.h>
```

Collaboration diagram for WsfBufDiag_t:



Data Fields

- union {
  wsfBufDiagAllocFail_t alloc
  Buffer allocation failure.
  } param

  Union of diagnostic data types.
- uint8_t type

  Type of error.

3.2.1 Detailed Description

WSF buffer diagnostics message.

Definition at line 137 of file wsf_buf.h.

The documentation for this struct was generated from the following file:

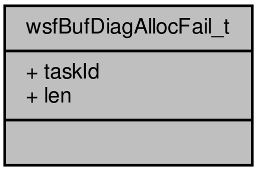
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_buf.h

3.3 wsfBufDiagAllocFail_t Struct Reference

WSF buffer diagnostics - buffer allocation failure.

```
#include <wsf_buf.h>
```

Collaboration diagram for wsfBufDiagAllocFail_t:



Data Fields

- uint8_t taskId

  Task handler ID where failure occured.
- uint16_t len

  Length of buffer being allocated.

3.3.1 Detailed Description

WSF buffer diagnostics - buffer allocation failure.

Definition at line 130 of file wsf_buf.h.

The documentation for this struct was generated from the following file:

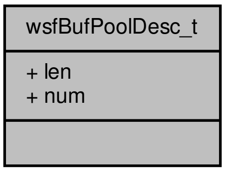
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_buf.h

3.4 wsfBufPoolDesc_t Struct Reference

Buffer pool descriptor structure.

```
#include <wsf_buf.h>
```

Collaboration diagram for wsfBufPoolDesc_t:



Data Fields

- uint16_t len

  Length of buffers in pool.
- uint8_t num

  Number of buffers in pool.

3.4.1 Detailed Description

Buffer pool descriptor structure.

Definition at line 110 of file wsf_buf.h.

The documentation for this struct was generated from the following file:

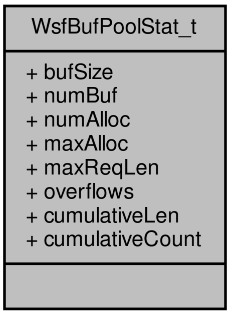
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_buf.h

3.5 WsfBufPoolStat_t Struct Reference

Pool statistics.

```
#include <wsf_buf.h>
```

Collaboration diagram for WsfBufPoolStat_t:



Data Fields

- uint16_t bufSize

  Pool buffer size.
- uint8_t numBuf

  Total number of buffers.
- uint8_t numAlloc

  Number of outstanding allocations.
- uint8_t maxAlloc

  High allocation watermark.
- uint16_t maxReqLen

  Maximum requested buffer length.
- uint8_t overflows

  Pool Overflow counter.
- uint32_t cumulativeLen

  Cumulative length of the allocated buffers.
- uint32_t cumulativeCount

  Cumulative count of the allocated buffers.

3.5.1 Detailed Description

Pool statistics.

Definition at line 117 of file wsf_buf.h.

The documentation for this struct was generated from the following file:

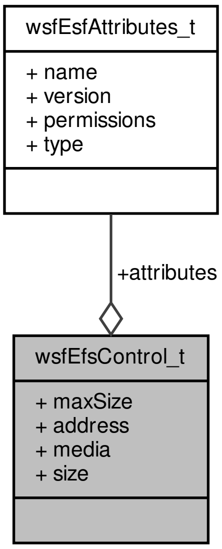
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_buf.h

3.6 wsfEfsControl_t Struct Reference

File control block data type.

```
#include <wsf_efs.h>
```

Collaboration diagram for wsfEfsControl_t:



Data Fields

- uint32_t maxSize

  File maximum size.
- uint32_t address

  File storage address.
- uint8_t media

  File media.
- uint32_t size

  File size.
- wsfEsfAttributes_t attributes

  File attributes.

3.6.1 Detailed Description

File control block data type.

Definition at line 159 of file wsf_efs.h.

The documentation for this struct was generated from the following file:

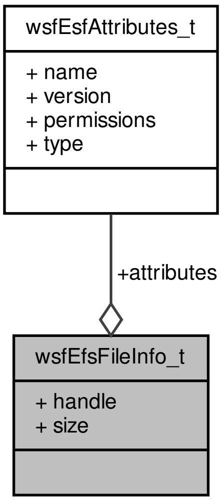
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_efs.h

3.7 wsfEfsFileInfo_t Struct Reference

File Listing Information.

```
#include <wsf_efs.h>
```

Collaboration diagram for wsfEfsFileInfo_t:



Data Fields

- wsfEfsHandle_t handle

  File handle.
- uint32_t size

  File size.
- wsfEsfAttributes_t attributes

  File attributes.

3.7.1 Detailed Description

File Listing Information.

Definition at line 169 of file wsf_efs.h.

The documentation for this struct was generated from the following file:

- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_efs.h

3.8 wsfEfsMedia_t Struct Reference

Media Control data type.

```
#include <wsf_efs.h>
```

Collaboration diagram for wsfEfsMedia_t:

Data Fields

- uint32_t startAddress

  Start address.
- uint32_t endAddress

  End address.
- uint32_t pageSize

  Page size.
- wsfMediaInitFunc_t ∗ init

  Media intialization callback.
- wsfMediaEraseFunc_t ∗ erase

  Media erase callback.
- wsfMediaReadFunc_t ∗ read

  Media read callback.
- wsfMediaWriteFunc_t ∗ write

  Media write callback.
- wsfMediaHandleCmdFunc_t ∗ handleCmd

  Media command handler callback.

3.8.1 Detailed Description

Media Control data type.

Definition at line 248 of file wsf_efs.h.

The documentation for this struct was generated from the following file:

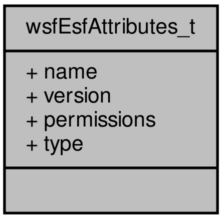
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_efs.h

3.9 wsfEsfAttributes_t Struct Reference

File attributes data type.

```
#include <wsf_efs.h>
```

Collaboration diagram for wsfEsfAttributes_t:



Data Fields

- char name [WSF_EFS_NAME_LEN]

  File name string.
- char version [WSF_EFS_VERSION_LEN]

  File version string.
- uint16_t permissions

  File permissions.
- uint8_t type

  File type.

3.9.1 Detailed Description

File attributes data type.

Definition at line 150 of file wsf_efs.h.

The documentation for this struct was generated from the following file:

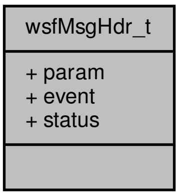
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_efs.h

3.10 wsfMsgHdr_t Struct Reference

Common message structure passed to event handler.

```
#include <wsf_os.h>
```

Collaboration diagram for wsfMsgHdr_t:



Data Fields

- uint16_t param

  General purpose parameter passed to event handler.
- uint8_t event

  General purpose event value passed to event handler.
- uint8_t status

  General purpose status value passed to event handler.

3.10.1 Detailed Description

Common message structure passed to event handler.

Definition at line 132 of file wsf_os.h.

The documentation for this struct was generated from the following file:

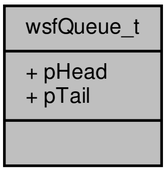
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_os.h

3.11 wsfQueue_t Struct Reference

Queue structure.

```
#include <wsf_queue.h>
```

Collaboration diagram for wsfQueue_t:



Data Fields

- void ∗ pHead

  head of queue
- void ∗ pTail

  tail of queue

3.11.1 Detailed Description

Queue structure.

Definition at line 46 of file wsf_queue.h.

The documentation for this struct was generated from the following file:

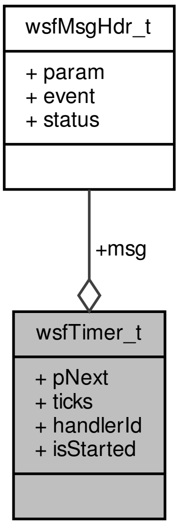
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_queue.h

3.12 wsfTimer_t Struct Reference

Timer structure.

```
#include <wsf_timer.h>
```

Collaboration diagram for wsfTimer_t:



Data Fields

- struct wsfTimer_tag ∗ pNext

  pointer to next timer in queue
- wsfMsgHdr_t msg

  application-defined timer event parameters
- wsfTimerTicks_t ticks

  number of ticks until expiration
- wsfHandlerId_t handlerId

  event handler for this timer
- bool_t isStarted

  TRUE if timer has been started.

3.12.1 Detailed Description

Timer structure.

Definition at line 78 of file wsf_timer.h.

The documentation for this struct was generated from the following file:

- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_timer.h

Chapter 4

File Documentation

4.1 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/hci_defs.h File Reference

HCI constants and definitions from the Bluetooth specification.

Macros

Packet definitions

- #define HCI_CMD_HDR_LEN 3
- #define HCI_ACL_HDR_LEN 4
- #define HCI_ISO_HDR_LEN 4
- #define HCI_EVT_HDR_LEN 2
- #define HCI_EVT_PARAM_MAX_LEN 255
- #define HCI_ACL_DEFAULT_LEN 27
- #define HCI_PB_FLAG_MASK 0x3000
- #define HCI_PB_START_H2C 0x0000
- #define HCI_PB_CONTINUE 0x1000
- #define HCI_PB_START_C2H 0x2000
- #define HCI_HANDLE_MASK 0x0FFF
- #define HCI_HANDLE_NONE 0xFFFF
- #define HCI_TS_FLAG_MASK (1 << 14)
- #define HCI_DATA_LOAD_LEN_MASK 0x3FFF
- #define HCI_ISO_DL_MIN_LEN 4
- #define HCI_ISO_DL_MAX_LEN 8
- #define HCI_ISO_TS_LEN 4
- #define HCI_ISO_DL_SDU_LEN_MASK 0x0FFF
- #define HCI_ISO_DL_PS_MASK 0xC000

Packet types

- #define HCI_CMD_TYPE 0x01
- #define HCI_ACL_TYPE 0x02
- #define HCI_EVT_TYPE 0x04
- #define HCI_ISO_TYPE 0x05

Error codes

- #define HCI_SUCCESS 0x00
- #define HCI_ERR_UNKNOWN_CMD 0x01

- #define HCI_ERR_UNKNOWN_HANDLE 0x02
- #define HCI_ERR_HARDWARE_FAILURE 0x03
- #define HCI_ERR_PAGE_TIMEOUT 0x04
- #define HCI_ERR_AUTH_FAILURE 0x05
- #define HCI_ERR_KEY_MISSING 0x06
- #define HCI_ERR_MEMORY_EXCEEDED 0x07
- #define HCI_ERR_CONN_TIMEOUT 0x08
- #define HCI_ERR_CONN_LIMIT 0x09
- #define HCI_ERR_SYNCH_CONN_LIMIT 0x0A
- #define HCI_ERR_ACL_CONN_EXISTS 0x0B
- #define HCI_ERR_CMD_DISALLOWED 0x0C
- #define HCI_ERR_REJ_RESOURCES 0x0D
- #define HCI_ERR_REJ_SECURITY 0x0E
- #define HCI_ERR_REJ_BD_ADDR 0x0F
- #define HCI_ERR_ACCEPT_TIMEOUT 0x10
- #define HCI_ERR_UNSUP_FEAT 0x11
- #define HCI_ERR_INVALID_PARAM 0x12
- #define HCI_ERR_REMOTE_TERMINATED 0x13
- #define HCI_ERR_REMOTE_RESOURCES 0x14
- #define HCI_ERR_REMOTE_POWER_OFF 0x15
- #define HCI_ERR_LOCAL_TERMINATED 0x16
- #define HCI_ERR_REPEATED_ATTEMPTS 0x17
- #define HCI_ERR_PAIRING_NOT_ALLOWED 0x18
- #define HCI_ERR_UNKNOWN_LMP_PDU 0x19
- #define HCI_ERR_UNSUP_REMOTE_FEAT 0x1A
- #define HCI_ERR_SCO_OFFSET 0x1B
- #define HCI_ERR_SCO_INTERVAL 0x1C
- #define HCI_ERR_SCO_MODE 0x1D
- #define HCI_ERR_LMP_PARAM 0x1E
- #define HCI_ERR_UNSPECIFIED 0x1F
- #define HCI_ERR_UNSUP_LMP_PARAM 0x20
- #define HCI_ERR_ROLE_CHANGE 0x21
- #define HCI_ERR_LL_RESP_TIMEOUT 0x22
- #define HCI_ERR_LMP_COLLISION 0x23
- #define HCI_ERR_LMP_PDU 0x24
- #define HCI_ERR_ENCRYPT_MODE 0x25
- #define HCI_ERR_LINK_KEY 0x26
- #define HCI_ERR_UNSUP_QOS 0x27
- #define HCI_ERR_INSTANT_PASSED 0x28
- #define HCI_ERR_UNSUP_UNIT_KEY 0x29
- #define HCI_ERR_TRANSACT_COLLISION 0x2A
- #define HCI_ERR_CHANNEL_CLASS 0x2E
- #define HCI_ERR_MEMORY 0x2F
- #define HCI_ERR_PARAMETER_RANGE 0x30
- #define HCI_ERR_ROLE_SWITCH_PEND 0x32
- #define HCI_ERR_RESERVED_SLOT 0x34
- #define HCI_ERR_ROLE_SWITCH 0x35
- #define HCI_ERR_INQ_TOO_LARGE 0x36
- #define HCI_ERR_UNSUP_SSP 0x37
- #define HCI_ERR_HOST_BUSY_PAIRING 0x38
- #define HCI_ERR_NO_CHANNEL 0x39
- #define HCI_ERR_CONTROLLER_BUSY 0x3A
- #define HCI_ERR_CONN_INTERVAL 0x3B
- #define HCI_ERR_ADV_TIMEOUT 0x3C
- #define HCI_ERR_MIC_FAILURE 0x3D
- #define HCI_ERR_CONN_FAIL 0x3E
- #define HCI_ERR_MAC_CONN_FAIL 0x3F
- #define HCI_ERR_COARSE_CLK_ADJ_REJ 0x40
- #define HCI_ERR_TYPE0_SUBMAP_NOT_DEF 0x41
- #define HCI_ERR_UNKNOWN_ADV_ID 0x42
- #define HCI_ERR_LIMIT_REACHED 0x43
- #define HCI_ERR_OP_CANCELLED_BY_HOST 0x44
- #define HCI_ERR_PKT_TOO_LONG 0x45

Command groups

- #define HCI_OGF_NOP 0x00
- #define HCI_OGF_LINK_CONTROL 0x01
- #define HCI_OGF_LINK_POLICY 0x02
- #define HCI_OGF_CONTROLLER 0x03
- #define HCI_OGF_INFORMATIONAL 0x04
- #define HCI_OGF_STATUS 0x05
- #define HCI_OGF_TESTING 0x06
- #define HCI_OGF_LE_CONTROLLER 0x08
- #define HCI_OGF_VENDOR_SPEC 0x3F

NOP command

- #define **HCI_OCF_NOP** 0x00

Link control commands

- #define **HCI_OCF_DISCONNECT** 0x06
- #define **HCI_OCF_READ_REMOTE_VER_INFO** 0x1D

Controller and baseband commands

- #define **HCI_OCF_SET_EVENT_MASK** 0x01
- #define **HCI_OCF_RESET** 0x03
- #define **HCI_OCF_READ_TX_PWR_LVL** 0x2D
- #define **HCI_OCF_SET_CONTROLLER_TO_HOST_FC** 0x31
- #define **HCI_OCF_HOST_BUFFER_SIZE** 0x33
- #define **HCI_OCF_HOST_NUM_CMPL_PKTS** 0x35
- #define **HCI_OCF_SET_EVENT_MASK_PAGE2** 0x63
- #define **HCI_OCF_READ_AUTH_PAYLOAD_TO** 0x7B
- #define **HCI_OCF_WRITE_AUTH_PAYLOAD_TO** 0x7C
- #define **HCI_OCF_CONFIG_DATA_PATH** 0x83

Informational commands

- #define **HCI_OCF_READ_LOCAL_VER_INFO** 0x01
- #define **HCI_OCF_READ_LOCAL_SUP_CMDS** 0x02
- #define **HCI_OCF_READ_LOCAL_SUP_FEAT** 0x03
- #define **HCI_OCF_READ_BUF_SIZE** 0x05
- #define **HCI_OCF_READ_BD_ADDR** 0x09
- #define **HCI_OCF_READ_LOCAL_SUP_CODECS** 0x0D
- #define **HCI_OCF_READ_LOCAL_SUP_CODEC_CAP** 0x0E
- #define **HCI_OCF_READ_LOCAL_SUP_CONTROLLER_DLY** 0x0F

Status commands

- #define **HCI_OCF_READ_RSSI** 0x05

LE controller commands

- #define **HCI_OCF_LE_SET_EVENT_MASK** 0x01
- #define **HCI_OCF_LE_READ_BUF_SIZE** 0x02
- #define **HCI_OCF_LE_READ_LOCAL_SUP_FEAT** 0x03
- #define **HCI_OCF_LE_SET_RAND_ADDR** 0x05
- #define **HCI_OCF_LE_SET_ADV_PARAM** 0x06
- #define **HCI_OCF_LE_READ_ADV_TX_POWER** 0x07
- #define **HCI_OCF_LE_SET_ADV_DATA** 0x08
- #define **HCI_OCF_LE_SET_SCAN_RESP_DATA** 0x09

- #define **HCI_OCF_LE_SET_ADV_ENABLE** 0x0A
- #define **HCI_OCF_LE_SET_SCAN_PARAM** 0x0B
- #define **HCI_OCF_LE_SET_SCAN_ENABLE** 0x0C
- #define **HCI_OCF_LE_CREATE_CONN** 0x0D
- #define **HCI_OCF_LE_CREATE_CONN_CANCEL** 0x0E
- #define **HCI_OCF_LE_READ_WHITE_LIST_SIZE** 0x0F
- #define **HCI_OCF_LE_CLEAR_WHITE_LIST** 0x10
- #define **HCI_OCF_LE_ADD_DEV_WHITE_LIST** 0x11
- #define **HCI_OCF_LE_REMOVE_DEV_WHITE_LIST** 0x12
- #define **HCI_OCF_LE_CONN_UPDATE** 0x13
- #define **HCI_OCF_LE_SET_HOST_CHAN_CLASS** 0x14
- #define **HCI_OCF_LE_READ_CHAN_MAP** 0x15
- #define **HCI_OCF_LE_READ_REMOTE_FEAT** 0x16
- #define **HCI_OCF_LE_ENCRYPT** 0x17
- #define **HCI_OCF_LE_RAND** 0x18
- #define **HCI_OCF_LE_START_ENCRYPTION** 0x19
- #define **HCI_OCF_LE_LTK_REQ_REPL** 0x1A
- #define **HCI_OCF_LE_LTK_REQ_NEG_REPL** 0x1B
- #define **HCI_OCF_LE_READ_SUP_STATES** 0x1C
- #define **HCI_OCF_LE_RECEIVER_TEST** 0x1D
- #define **HCI_OCF_LE_TRANSMITTER_TEST** 0x1E
- #define **HCI_OCF_LE_TEST_END** 0x1F
- #define **HCI_OCF_LE_REM_CONN_PARAM_REP** 0x20
- #define **HCI_OCF_LE_REM_CONN_PARAM_NEG_REP** 0x21
- #define **HCI_OCF_LE_SET_DATA_LEN** 0x22
- #define **HCI_OCF_LE_READ_DEF_DATA_LEN** 0x23
- #define **HCI_OCF_LE_WRITE_DEF_DATA_LEN** 0x24
- #define **HCI_OCF_LE_READ_LOCAL_P256_PUB_KEY** 0x25
- #define **HCI_OCF_LE_GENERATE_DHKEY** 0x26
- #define **HCI_OCF_LE_ADD_DEV_RES_LIST** 0x27
- #define **HCI_OCF_LE_REMOVE_DEV_RES_LIST** 0x28
- #define **HCI_OCF_LE_CLEAR_RES_LIST** 0x29
- #define **HCI_OCF_LE_READ_RES_LIST_SIZE** 0x2A
- #define **HCI_OCF_LE_READ_PEER_RES_ADDR** 0x2B
- #define **HCI_OCF_LE_READ_LOCAL_RES_ADDR** 0x2C
- #define **HCI_OCF_LE_SET_ADDR_RES_ENABLE** 0x2D
- #define **HCI_OCF_LE_SET_RES_PRIV_ADDR_TO** 0x2E
- #define **HCI_OCF_LE_READ_MAX_DATA_LEN** 0x2F
- #define **HCI_OCF_LE_READ_PHY** 0x30
- #define **HCI_OCF_LE_SET_DEF_PHY** 0x31
- #define **HCI_OCF_LE_SET_PHY** 0x32
- #define **HCI_OCF_LE_ENHANCED_RECEIVER_TEST** 0x33
- #define **HCI_OCF_LE_ENHANCED_TRANSMITTER_TEST** 0x34
- #define **HCI_OCF_LE_SET_ADV_SET_RAND_ADDR** 0x35
- #define **HCI_OCF_LE_SET_EXT_ADV_PARAM** 0x36
- #define **HCI_OCF_LE_SET_EXT_ADV_DATA** 0x37
- #define **HCI_OCF_LE_SET_EXT_SCAN_RESP_DATA** 0x38
- #define **HCI_OCF_LE_SET_EXT_ADV_ENABLE** 0x39
- #define **HCI_OCF_LE_READ_MAX_ADV_DATA_LEN** 0x3A
- #define **HCI_OCF_LE_READ_NUM_SUP_ADV_SETS** 0x3B
- #define **HCI_OCF_LE_REMOVE_ADV_SET** 0x3C
- #define **HCI_OCF_LE_CLEAR_ADV_SETS** 0x3D
- #define **HCI_OCF_LE_SET_PER_ADV_PARAM** 0x3E
- #define **HCI_OCF_LE_SET_PER_ADV_DATA** 0x3F
- #define **HCI_OCF_LE_SET_PER_ADV_ENABLE** 0x40
- #define **HCI_OCF_LE_SET_EXT_SCAN_PARAM** 0x41
- #define **HCI_OCF_LE_SET_EXT_SCAN_ENABLE** 0x42
- #define **HCI_OCF_LE_EXT_CREATE_CONN** 0x43
- #define **HCI_OCF_LE_PER_ADV_CREATE_SYNC** 0x44
- #define **HCI_OCF_LE_PER_ADV_CREATE_SYNC_CANCEL** 0x45
- #define **HCI_OCF_LE_PER_ADV_TERM_SYNC** 0x46
- #define **HCI_OCF_LE_ADD_DEV_PER_ADV_LIST** 0x47
- #define **HCI_OCF_LE_REMOVE_DEV_PER_ADV_LIST** 0x48
- #define **HCI_OCF_LE_CLEAR_PER_ADV_LIST** 0x49

- #define **HCI_OCF_LE_READ_PER_ADV_LIST_SIZE** 0x4A
- #define **HCI_OCF_LE_READ_TX_POWER** 0x4B
- #define **HCI_OCF_LE_READ_RF_PATH_COMP** 0x4C
- #define **HCI_OCF_LE_WRITE_RF_PATH_COMP** 0x4D
- #define **HCI_OCF_LE_SET_PRIVACY_MODE** 0x4E
- #define **HCI_OCF_LE_RECEIVER_TEST_V3** 0x4F
- #define **HCI_OCF_LE_TRANSMITTER_TEST_V3** 0x50
- #define **HCI_OCF_LE_SET_CONNLESS_CTE_TX_PARAMS** 0x51
- #define **HCI_OCF_LE_SET_CONNLESS_CTE_TX_ENABLE** 0x52
- #define **HCI_OCF_LE_SET_CONNLESS_IQ_SAMP_ENABLE** 0x53
- #define **HCI_OCF_LE_SET_CONN_CTE_RX_PARAMS** 0x54
- #define **HCI_OCF_LE_SET_CONN_CTE_TX_PARAMS** 0x55
- #define **HCI_OCF_LE_CONN_CTE_REQ_ENABLE** 0x56
- #define **HCI_OCF_LE_CONN_CTE_RSP_ENABLE** 0x57
- #define **HCI_OCF_LE_READ_ANTENNA_INFO** 0x58
- #define **HCI_OCF_LE_SET_PER_ADV_RCV_ENABLE** 0x59
- #define **HCI_OCF_LE_PER_ADV_SYNC_TRANSFER** 0x5A
- #define **HCI_OCF_LE_PER_ADV_SET_INFO_TRANSFER** 0x5B
- #define **HCI_OCF_LE_SET_PAST_PARAM** 0x5C
- #define **HCI_OCF_LE_SET_DEFAULT_PAST_PARAM** 0x5D
- #define **HCI_OCF_LE_GENERATE_DHKEY_V2** 0x5E
- #define **HCI_OCF_LE_MODIFY_SLEEP_CLK_ACC** 0x5F
- #define **HCI_OCF_LE_READ_BUF_SIZE_V2** 0x60
- #define **HCI_OCF_LE_READ_ISO_TX_SYNC** 0x61
- #define **HCI_OCF_LE_SET_CIG_PARAMS** 0x62
- #define **HCI_OCF_LE_SET_CIG_PARAMS_TEST** 0x63
- #define **HCI_OCF_LE_CREATE_CIS** 0x64
- #define **HCI_OCF_LE_REMOVE_CIG** 0x65
- #define **HCI_OCF_LE_ACCEPT_CIS_REQ** 0x66
- #define **HCI_OCF_LE_REJECT_CIS_REQ** 0x67
- #define **HCI_OCF_LE_CREATE_BIG** 0x68
- #define **HCI_OCF_LE_CREATE_BIG_TEST** 0x69
- #define **HCI_OCF_LE_TERMINATE_BIG** 0x6A
- #define **HCI_OCF_LE_BIG_CREATE_SYNC** 0x6B
- #define **HCI_OCF_LE_BIG_TERMINATE_SYNC** 0x6C
- #define **HCI_OCF_LE_REQUEST_PEER_SCA** 0x6D
- #define **HCI_OCF_LE_SETUP_ISO_DATA_PATH** 0x6E
- #define **HCI_OCF_LE_REMOVE_ISO_DATA_PATH** 0x6F
- #define **HCI_OCF_LE_ISO_TX_TEST** 0x70
- #define **HCI_OCF_LE_ISO_RX_TEST** 0x71
- #define **HCI_OCF_LE_ISO_READ_TEST_COUNTERS** 0x72
- #define **HCI_OCF_LE_ISO_TEST_END** 0x73
- #define **HCI_OCF_LE_SET_HOST_FEATURE** 0x74
- #define **HCI_OCF_LE_READ_ISO_LINK_QUAL** 0x75
- #define **HCI_OCF_LE_READ_ENHANCED_TX_POWER** 0x76
- #define **HCI_OCF_LE_READ_REMOTE_TX_POWER** 0x77
- #define **HCI_OCF_LE_SET_PATH_LOSS_REPORTING_PARAMS** 0x78
- #define **HCI_OCF_LE_SET_PATH_LOSS_REPORTING_ENABLE** 0x79
- #define **HCI_OCF_LE_SET_TX_POWER_REPORT_ENABLE** 0x7A

Opcode manipulation macros

- #define **HCI_OPCODE**(ogf, ocf) (((ogf) << 10) + (ocf))
- #define **HCI_OGF**(opcode) ((opcode) >> 10)
- #define **HCI_OCF**(opcode) ((opcode) & 0x03FF)

Command opcodes

- #define **HCI_OPCODE_NOP** HCI_OPCODE(HCI_OGF_NOP, HCI_OCF_NOP)
- #define **HCI_OPCODE_DISCONNECT** HCI_OPCODE(HCI_OGF_LINK_CONTROL, HCI_OCF_DISCONNECT)

- #define **HCI_OPCODE_READ_REMOTE_VER_INFO** HCI_OPCODE(HCI_OGF_LINK_CONTROL, HCI_OCF_READ_REMOTE_VER_INFO)
- #define **HCI_OPCODE_SET_EVENT_MASK** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_SET_EVENT_MASK)
- #define **HCI_OPCODE_RESET** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_RESET)
- #define **HCI_OPCODE_HOST_BUFFER_SIZE** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_HOST_BUFFER_SIZE)
- #define **HCI_OPCODE_READ_TX_PWR_LVL** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_READ_TX_PWR_LVL)
- #define **HCI_OPCODE_SET_EVENT_MASK_PAGE2** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_SET_EVENT_MASK_PAGE2)
- #define **HCI_OPCODE_READ_AUTH_PAYLOAD_TO** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_READ_AUTH_PAYLOAD_TO)
- #define **HCI_OPCODE_WRITE_AUTH_PAYLOAD_TO** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_WRITE_AUTH_PAYLOAD_TO)
- #define **HCI_OPCODE_CONFIG_DATA_PATH** HCI_OPCODE(HCI_OGF_CONTROLLER, HCI_OCF_CONFIG_DATA_PATH)
- #define **HCI_OPCODE_READ_LOCAL_VER_INFO** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_LOCAL_VER_INFO)
- #define **HCI_OPCODE_READ_LOCAL_SUP_CMDS** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_LOCAL_SUP_CMDS)
- #define **HCI_OPCODE_READ_LOCAL_SUP_FEAT** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_LOCAL_SUP_FEAT)
- #define **HCI_OPCODE_READ_BUF_SIZE** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_BUF_SIZE)
- #define **HCI_OPCODE_READ_BD_ADDR** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_BD_ADDR)
- #define **HCI_OPCODE_READ_LOCAL_SUP_CODECS** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_LOCAL_SUP_CODECS)
- #define **HCI_OPCODE_READ_LOCAL_SUP_CODEC_CAP** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_LOCAL_SUP_CODEC_CAP)
- #define **HCI_OPCODE_READ_LOCAL_SUP_CONTROLLER_DLY** HCI_OPCODE(HCI_OGF_INFORMATIONAL, HCI_OCF_READ_LOCAL_SUP_CONTROLLER_DLY)
- #define **HCI_OPCODE_READ_RSSI** HCI_OPCODE(HCI_OGF_STATUS, HCI_OCF_READ_RSSI)
- #define **HCI_OPCODE_LE_SET_EVENT_MASK** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EVENT_MASK)
- #define **HCI_OPCODE_LE_READ_BUF_SIZE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_BUF_SIZE)
- #define **HCI_OPCODE_LE_READ_LOCAL_SUP_FEAT** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_LOCAL_SUP_FEAT)
- #define **HCI_OPCODE_LE_SET_RAND_ADDR** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_RAND_ADDR)
- #define **HCI_OPCODE_LE_SET_ADV_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_ADV_PARAM)
- #define **HCI_OPCODE_LE_READ_ADV_TX_POWER** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_ADV_TX_POWER)
- #define **HCI_OPCODE_LE_SET_ADV_DATA** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_ADV_DATA)
- #define **HCI_OPCODE_LE_SET_SCAN_RESP_DATA** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_SCAN_RESP_DATA)
- #define **HCI_OPCODE_LE_SET_ADV_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_ADV_ENABLE)
- #define **HCI_OPCODE_LE_SET_SCAN_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_SCAN_PARAM)
- #define **HCI_OPCODE_LE_SET_SCAN_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_SCAN_ENABLE)
- #define **HCI_OPCODE_LE_CREATE_CONN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CREATE_CONN)
- #define **HCI_OPCODE_LE_CREATE_CONN_CANCEL** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CREATE_CONN_CANCEL)
- #define **HCI_OPCODE_LE_READ_WHITE_LIST_SIZE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_WHITE_LIST_SIZE)

- #define **HCI_OPCODE_LE_CLEAR_WHITE_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CLEAR_WHITE_LIST)
- #define **HCI_OPCODE_LE_ADD_DEV_WHITE_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ADD_DEV_WHITE_LIST)
- #define **HCI_OPCODE_LE_REMOVE_DEV_WHITE_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REMOVE_DEV_WHITE_LIST)
- #define **HCI_OPCODE_LE_CONN_UPDATE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CONN_UPDATE)
- #define **HCI_OPCODE_LE_SET_HOST_CHAN_CLASS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_HOST_CHAN_CLASS)
- #define **HCI_OPCODE_LE_READ_CHAN_MAP** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_CHAN_MAP)
- #define **HCI_OPCODE_LE_READ_REMOTE_FEAT** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_REMOTE_FEAT)
- #define **HCI_OPCODE_LE_ENCRYPT** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ENCRYPT)
- #define **HCI_OPCODE_LE_RAND** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_RAND)
- #define **HCI_OPCODE_LE_START_ENCRYPTION** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_START_ENCRYPTION)
- #define **HCI_OPCODE_LE_LTK_REQ_REPL** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_LTK_REQ_REPL)
- #define **HCI_OPCODE_LE_LTK_REQ_NEG_REPL** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_LTK_REQ_NEG_REPL)
- #define **HCI_OPCODE_LE_READ_SUP_STATES** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_SUP_STATES)
- #define **HCI_OPCODE_LE_RECEIVER_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_RECEIVER_TEST)
- #define **HCI_OPCODE_LE_TRANSMITTER_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_TRANSMITTER_TEST)
- #define **HCI_OPCODE_LE_TEST_END** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_TEST_END)
- #define **HCI_OPCODE_LE_REM_CONN_PARAM_REP** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REM_CONN_PARAM_REP)
- #define **HCI_OPCODE_LE_REM_CONN_PARAM_NEG_REP** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REM_CONN_PARAM_NEG_REP)
- #define **HCI_OPCODE_LE_SET_DATA_LEN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_DATA_LEN)
- #define **HCI_OPCODE_LE_READ_DEF_DATA_LEN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_DEF_DATA_LEN)
- #define **HCI_OPCODE_LE_WRITE_DEF_DATA_LEN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_WRITE_DEF_DATA_LEN)
- #define **HCI_OPCODE_LE_READ_LOCAL_P256_PUB_KEY** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_LOCAL_P256_PUB_KEY)
- #define **HCI_OPCODE_LE_GENERATE_DHKEY** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_GENERATE_DHKEY)
- #define **HCI_OPCODE_LE_ADD_DEV_RES_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ADD_DEV_RES_LIST)
- #define **HCI_OPCODE_LE_REMOVE_DEV_RES_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REMOVE_DEV_RES_LIST)
- #define **HCI_OPCODE_LE_CLEAR_RES_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CLEAR_RES_LIST)
- #define **HCI_OPCODE_LE_READ_RES_LIST_SIZE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_RES_LIST_SIZE)
- #define **HCI_OPCODE_LE_READ_PEER_RES_ADDR** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_PEER_RES_ADDR)
- #define **HCI_OPCODE_LE_READ_LOCAL_RES_ADDR** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_LOCAL_RES_ADDR)
- #define **HCI_OPCODE_LE_SET_ADDR_RES_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_ADDR_RES_ENABLE)
- #define **HCI_OPCODE_LE_SET_RES_PRIV_ADDR_TO** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_RES_PRIV_ADDR_TO)

- #define **HCI_OPCODE_LE_READ_MAX_DATA_LEN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_MAX_DATA_LEN)
- #define **HCI_OPCODE_LE_READ_PHY** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_PHY)
- #define **HCI_OPCODE_LE_SET_DEF_PHY** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_DEF_PHY)
- #define **HCI_OPCODE_LE_SET_PHY** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PHY)
- #define **HCI_OPCODE_LE_ENHANCED_RECEIVER_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ENHANCED_RECEIVER_TEST)
- #define **HCI_OPCODE_LE_ENHANCED_TRANSMITTER_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ENHANCED_TRANSMITTER_TEST)
- #define **HCI_OPCODE_LE_SET_ADV_SET_RAND_ADDR** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_ADV_SET_RAND_ADDR)
- #define **HCI_OPCODE_LE_SET_EXT_ADV_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EXT_ADV_PARAM)
- #define **HCI_OPCODE_LE_SET_EXT_ADV_DATA** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EXT_ADV_DATA)
- #define **HCI_OPCODE_LE_SET_EXT_SCAN_RESP_DATA** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EXT_SCAN_RESP_DATA)
- #define **HCI_OPCODE_LE_SET_EXT_ADV_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EXT_ADV_ENABLE)
- #define **HCI_OPCODE_LE_READ_MAX_ADV_DATA_LEN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_MAX_ADV_DATA_LEN)
- #define **HCI_OPCODE_LE_READ_NUM_SUP_ADV_SETS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_NUM_SUP_ADV_SETS)
- #define **HCI_OPCODE_LE_REMOVE_ADV_SET** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REMOVE_ADV_SET)
- #define **HCI_OPCODE_LE_CLEAR_ADV_SETS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CLEAR_ADV_SETS)
- #define **HCI_OPCODE_LE_SET_PER_ADV_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PER_ADV_PARAM)
- #define **HCI_OPCODE_LE_SET_PER_ADV_DATA** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PER_ADV_DATA)
- #define **HCI_OPCODE_LE_SET_PER_ADV_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PER_ADV_ENABLE)
- #define **HCI_OPCODE_LE_SET_EXT_SCAN_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EXT_SCAN_PARAM)
- #define **HCI_OPCODE_LE_SET_EXT_SCAN_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_EXT_SCAN_ENABLE)
- #define **HCI_OPCODE_LE_EXT_CREATE_CONN** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_EXT_CREATE_CONN)
- #define **HCI_OPCODE_LE_PER_ADV_CREATE_SYNC** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_PER_ADV_CREATE_SYNC)
- #define **HCI_OPCODE_LE_PER_ADV_CREATE_SYNC_CANCEL** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_PER_ADV_CREATE_SYNC_CANCEL)
- #define **HCI_OPCODE_LE_PER_ADV_TERMINATE_SYNC** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_PER_ADV_TERM_SYNC)
- #define **HCI_OPCODE_LE_ADD_DEV_PER_ADV_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ADD_DEV_PER_ADV_LIST)
- #define **HCI_OPCODE_LE_REMOVE_DEV_PER_ADV_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REMOVE_DEV_PER_ADV_LIST)
- #define **HCI_OPCODE_LE_CLEAR_PER_ADV_LIST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CLEAR_PER_ADV_LIST)
- #define **HCI_OPCODE_LE_READ_PER_ADV_LIST_SIZE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_PER_ADV_LIST_SIZE)
- #define **HCI_OPCODE_LE_READ_TX_POWER** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_TX_POWER)
- #define **HCI_OPCODE_LE_WRITE_RF_PATH_COMP** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_WRITE_RF_PATH_COMP)
- #define **HCI_OPCODE_LE_READ_RF_PATH_COMP** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_RF_PATH_COMP)

- #define **HCI_OPCODE_LE_SET_PRIVACY_MODE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PRIVACY_MODE)
- #define **HCI_OPCODE_LE_RECEIVER_TEST_V3** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_RECEIVER_TEST_V3)
- #define **HCI_OPCODE_LE_TRANSMITTER_TEST_V3** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_TRANSMITTER_TEST_V3)
- #define **HCI_OPCODE_LE_SET_CONNLESS_CTE_TX_PARAMS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CONNLESS_CTE_TX_PARAMS)
- #define **HCI_OPCODE_LE_SET_CONNLESS_CTE_TX_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CONNLESS_CTE_TX_ENABLE)
- #define **HCI_OPCODE_LE_SET_CONNLESS_IQ_SAMP_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CONNLESS_IQ_SAMP_ENABLE)
- #define **HCI_OPCODE_LE_SET_CONN_CTE_RX_PARAMS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CONN_CTE_RX_PARAMS)
- #define **HCI_OPCODE_LE_SET_CONN_CTE_TX_PARAMS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CONN_CTE_TX_PARAMS)
- #define **HCI_OPCODE_LE_CONN_CTE_REQ_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CONN_CTE_REQ_ENABLE)
- #define **HCI_OPCODE_LE_CONN_CTE_RSP_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CONN_CTE_RSP_ENABLE)
- #define **HCI_OPCODE_LE_READ_ANTENNA_INFO** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_ANTENNA_INFO)
- #define **HCI_OPCODE_LE_SET_PER_ADV_RCV_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PER_ADV_RCV_ENABLE)
- #define **HCI_OPCODE_LE_PER_ADV_SYNC_TRANSFER** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_PER_ADV_SYNC_TRANSFER)
- #define **HCI_OPCODE_LE_PER_ADV_SET_INFO_TRANSFER** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_PER_ADV_SET_INFO_TRANSFER)
- #define **HCI_OPCODE_LE_SET_PAST_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PAST_PARAM)
- #define **HCI_OPCODE_LE_SET_DEFAULT_PAST_PARAM** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_DEFAULT_PAST_PARAM)
- #define **HCI_OPCODE_LE_GENERATE_DHKEY_V2** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_GENERATE_DHKEY_V2)
- #define **HCI_OPCODE_LE_MODIFY_SLEEP_CLK_ACC** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_MODIFY_SLEEP_CLK_ACC)
- #define **HCI_OPCODE_LE_READ_BUF_SIZE_V2** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_BUF_SIZE_V2)
- #define **HCI_OPCODE_LE_READ_ISO_TX_SYNC** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_ISO_TX_SYNC)
- #define **HCI_OPCODE_LE_SET_CIG_PARAMS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CIG_PARAMS)
- #define **HCI_OPCODE_LE_SET_CIG_PARAMS_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_CIG_PARAMS_TEST)
- #define **HCI_OPCODE_LE_CREATE_CIS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CREATE_CIS)
- #define **HCI_OPCODE_LE_REMOVE_CIG** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REMOVE_CIG)
- #define **HCI_OPCODE_LE_ACCEPT_CIS_REQ** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ACCEPT_CIS_REQ)
- #define **HCI_OPCODE_LE_REJECT_CIS_REQ** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REJECT_CIS_REQ)
- #define **HCI_OPCODE_LE_CREATE_BIG** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CREATE_BIG)
- #define **HCI_OPCODE_LE_CREATE_BIG_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_CREATE_BIG_TEST)
- #define **HCI_OPCODE_LE_TERMINATE_BIG** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_TERMINATE_BIG)
- #define **HCI_OPCODE_LE_BIG_CREATE_SYNC** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_BIG_CREATE_SYNC)
- #define **HCI_OPCODE_LE_BIG_TERMINATE_SYNC** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_BIG_TERMINATE_SYNC)

- #define **HCI_OPCODE_LE_REQUEST_PEER_SCA** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REQUEST_PEER_SCA)
- #define **HCI_OPCODE_LE_SETUP_ISO_DATA_PATH** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SETUP_ISO_DATA_PATH)
- #define **HCI_OPCODE_LE_REMOVE_ISO_DATA_PATH** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_REMOVE_ISO_DATA_PATH)
- #define **HCI_OPCODE_LE_ISO_TX_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ISO_TX_TEST)
- #define **HCI_OPCODE_LE_ISO_RX_TEST** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ISO_RX_TEST)
- #define **HCI_OPCODE_LE_ISO_READ_TEST_COUNTERS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ISO_READ_TEST_COUNTERS)
- #define **HCI_OPCODE_LE_ISO_TEST_END** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_ISO_TEST_END)
- #define **HCI_OPCODE_LE_SET_HOST_FEATURE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_HOST_FEATURE)
- #define **HCI_OPCODE_LE_READ_ISO_LINK_QUAL** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_ISO_LINK_QUAL)
- #define **HCI_OPCODE_LE_READ_ENHANCED_TX_POWER** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_ENHANCED_TX_POWER)
- #define **HCI_OPCODE_LE_READ_REMOTE_TX_POWER** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_READ_REMOTE_TX_POWER)
- #define **HCI_OPCODE_LE_SET_PATH_LOSS_REPORTING_PARAMS** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PATH_LOSS_REPORTING_PARAMS)
- #define **HCI_OPCODE_LE_SET_PATH_LOSS_REPORTING_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_PATH_LOSS_REPORTING_ENABLE)
- #define **HCI_OPCODE_LE_SET_TX_POWER_REPORT_ENABLE** HCI_OPCODE(HCI_OGF_LE_CONTROLLER, HCI_OCF_LE_SET_TX_POWER_REPORT_ENABLE)

Packetcraft Vendor Specific

- #define **HCI_OPCODE_LE_VS_ENABLE_READ_FEAT_ON_CONN** ((uint16_t)(0xfff3))

Command parameter lengths

- #define **HCI_LEN_NOP** 0
- #define **HCI_LEN_DISCONNECT** 3
- #define **HCI_LEN_READ_REMOTE_VER_INFO** 2
- #define **HCI_LEN_SET_EVENT_MASK** 8
- #define **HCI_LEN_SET_EVENT_MASK_PAGE2** 8
- #define **HCI_LEN_RESET** 0
- #define **HCI_LEN_READ_TX_PWR_LVL** 3
- #define **HCI_LEN_SET_CONTROLLER_TO_HOST_FC** 1
- #define **HCI_LEN_HOST_BUFFER_SIZE** 7
- #define **HCI_LEN_HOST_NUM_CMPL_PKTS** 1
- #define **HCI_LEN_CONFIG_DATA_PATH**(cLen) (3 + (cLen))
- #define **HCI_LEN_READ_LOCAL_VER_INFO** 0
- #define **HCI_LEN_READ_LOCAL_SUP_CMDS** 0
- #define **HCI_LEN_READ_LOCAL_SUP_FEAT** 0
- #define **HCI_LEN_READ_BUF_SIZE** 0
- #define **HCI_LEN_READ_BD_ADDR** 0
- #define **HCI_LEN_READ_LOCAL_SUP_CODECS** 0
- #define **HCI_LEN_READ_LOCAL_SUP_CODEC_CAP** 7
- #define **HCI_LEN_READ_LOCAL_SUP_CONTROLLER_DLY**(ccLen) (8 + (ccLen))
- #define **HCI_LEN_READ_RSSI** 2
- #define **HCI_LEN_READ_AUTH_PAYLOAD_TO** 2
- #define **HCI_LEN_WRITE_AUTH_PAYLOAD_TO** 4
- #define **HCI_LEN_LE_SET_EVENT_MASK** 8
- #define **HCI_LEN_LE_READ_BUF_SIZE** 0
- #define **HCI_LEN_LE_READ_LOCAL_SUP_FEAT** 0
- #define **HCI_LEN_LE_SET_RAND_ADDR** 6

- #define **HCI_LEN_LE_SET_ADV_PARAM** 15
- #define **HCI_LEN_LE_READ_ADV_TX_POWER** 0
- #define **HCI_LEN_LE_SET_ADV_DATA** 32
- #define **HCI_LEN_LE_SET_SCAN_RESP_DATA** 32
- #define **HCI_LEN_LE_SET_ADV_ENABLE** 1
- #define **HCI_LEN_LE_SET_SCAN_PARAM** 7
- #define **HCI_LEN_LE_SET_SCAN_ENABLE** 2
- #define **HCI_LEN_LE_CREATE_CONN** 25
- #define **HCI_LEN_LE_CREATE_CONN_CANCEL** 0
- #define **HCI_LEN_LE_READ_WHITE_LIST_SIZE** 0
- #define **HCI_LEN_LE_CLEAR_WHITE_LIST** 0
- #define **HCI_LEN_LE_ADD_DEV_WHITE_LIST** 7
- #define **HCI_LEN_LE_REMOVE_DEV_WHITE_LIST** 7
- #define **HCI_LEN_LE_CONN_UPDATE** 14
- #define **HCI_LEN_LE_SET_HOST_CHAN_CLASS** 5
- #define **HCI_LEN_LE_READ_CHAN_MAP** 2
- #define **HCI_LEN_LE_READ_REMOTE_FEAT** 2
- #define **HCI_LEN_LE_ENCRYPT** 32
- #define **HCI_LEN_LE_RAND** 0
- #define **HCI_LEN_LE_START_ENCRYPTION** 28
- #define **HCI_LEN_LE_LTK_REQ_REPL** 18
- #define **HCI_LEN_LE_LTK_REQ_NEG_REPL** 2
- #define **HCI_LEN_LE_READ_SUP_STATES** 0
- #define **HCI_LEN_LE_RECEIVER_TEST** 1
- #define **HCI_LEN_LE_TRANSMITTER_TEST** 3
- #define **HCI_LEN_LE_TEST_END** 0
- #define **HCI_LEN_LE_REM_CONN_PARAM_REP** 14
- #define **HCI_LEN_LE_REM_CONN_PARAM_NEG_REP** 3
- #define **HCI_LEN_LE_SET_DATA_LEN** 6
- #define **HCI_LEN_LE_READ_DEF_DATA_LEN** 0
- #define **HCI_LEN_LE_WRITE_DEF_DATA_LEN** 4
- #define **HCI_LEN_LE_READ_LOCAL_P256_PUB_KEY** 0
- #define **HCI_LEN_LE_GENERATE_DHKEY** 64
- #define **HCI_LEN_LE_ADD_DEV_RES_LIST** 39
- #define **HCI_LEN_LE_REMOVE_DEV_RES_LIST** 7
- #define **HCI_LEN_LE_CLEAR_RES_LIST** 0
- #define **HCI_LEN_LE_READ_RES_LIST_SIZE** 0
- #define **HCI_LEN_LE_READ_PEER_RES_ADDR** 7
- #define **HCI_LEN_LE_READ_LOCAL_RES_ADDR** 7
- #define **HCI_LEN_LE_SET_ADDR_RES_ENABLE** 1
- #define **HCI_LEN_LE_SET_RES_PRIV_ADDR_TO** 2
- #define **HCI_LEN_LE_READ_MAX_DATA_LEN** 0
- #define **HCI_LEN_LE_READ_PHY** 2
- #define **HCI_LEN_LE_SET_DEF_PHY** 3
- #define **HCI_LEN_LE_SET_PHY** 7
- #define **HCI_LEN_LE_ENHANCED_RECEIVER_TEST** 3
- #define **HCI_LEN_LE_ENHANCED_TRANSMITTER_TEST** 4
- #define **HCI_LEN_LE_SET_ADV_SET_RAND_ADDR** 7
- #define **HCI_LEN_LE_SET_EXT_ADV_PARAM** 25
- #define **HCI_LEN_LE_SET_EXT_ADV_DATA**(len) (4 + (len))
- #define **HCI_LEN_LE_SET_EXT_SCAN_RESP_DATA**(len) (4 + (len))
- #define **HCI_LEN_LE_EXT_ADV_ENABLE**(numSets) (2 + (4 ∗ (numSets)))
- #define **HCI_LEN_LE_READ_MAX_ADV_DATA_LEN** 0
- #define **HCI_LEN_LE_READ_NUM_OF_SUP_ADV_SETS** 0
- #define **HCI_LEN_LE_REMOVE_ADV_SET** 1
- #define **HCI_LEN_LE_CLEAR_ADV_SETS** 0
- #define **HCI_LEN_LE_SET_PER_ADV_PARAM** 7
- #define **HCI_LEN_LE_SET_PER_ADV_DATA**(len) (3 + (len))
- #define **HCI_LEN_LE_SET_PER_ADV_ENABLE** 2
- #define **HCI_LEN_LE_SET_EXT_SCAN_PARAM**(numPhys) (3 + (5 ∗ (numPhys)))
- #define **HCI_LEN_LE_SET_EXT_SCAN_ENABLE** 6
- #define **HCI_LEN_LE_EXT_CREATE_CONN**(numPhys) (10 + (16 ∗ (numPhys)))
- #define **HCI_LEN_LE_PER_ADV_CREATE_SYNC** 14
- #define **HCI_LEN_LE_PER_ADV_CREATE_SYNC_CANCEL** 0

- #define **HCI_LEN_LE_PER_ADV_TERMINATE_SYNC** 2
- #define **HCI_LEN_LE_ADD_DEV_PER_ADV_LIST** 8
- #define **HCI_LEN_LE_REMOVE_DEV_PER_ADV_LIST** 8
- #define **HCI_LEN_LE_CLEAR_PER_ADV_LIST** 0
- #define **HCI_LEN_LE_READ_PER_ADV_LIST_SIZE** 0
- #define **HCI_LEN_LE_READ_TX_POWER** 0
- #define **HCI_LEN_LE_READ_RF_PATH_COMP** 0
- #define **HCI_LEN_LE_WRITE_RF_PATH_COMP** 4
- #define **HCI_LEN_LE_SET_PRIVACY_MODE** 8
- #define **HCI_LEN_LE_SET_CONN_CTE_RX_PARAMS**(spLen) (5 + (spLen))
- #define **HCI_LEN_LE_SET_CONN_CTE_TX_PARAMS**(spLen) (4 + (spLen))
- #define **HCI_LEN_LE_CONN_CTE_REQ_ENABLE** 7
- #define **HCI_LEN_LE_CONN_CTE_RSP_ENABLE** 3
- #define **HCI_LEN_LE_READ_ANTENNA_INFO** 0
- #define **HCI_LEN_LE_SET_PER_ADV_RCV_ENABLE** 3
- #define **HCI_LEN_LE_PER_ADV_SYNC_TRANSFER** 6
- #define **HCI_LEN_LE_PER_ADV_SET_INFO_TRANSFER** 5
- #define **HCI_LEN_LE_SET_PAST_PARAM** 8
- #define **HCI_LEN_LE_SET_DEFAULT_PAST_PARAM** 6
- #define **HCI_LEN_LE_GENERATE_DHKEY_V2** 65
- #define **HCI_LEN_LE_SET_CIG_PARAMS**(numCis) (15 + (9 ∗ (numCis)))
- #define **HCI_LEN_LE_CREATE_CIS**(numCis) (1 + (4 ∗ (numCis)))
- #define **HCI_LEN_LE_REMOVE_CIG** 1
- #define **HCI_LEN_LE_ACCEPT_CIS_REQ** 2
- #define **HCI_LEN_LE_REJECT_CIS_REQ** 3
- #define **HCI_LEN_LE_REQUEST_PEER_SCA** 2
- #define **HCI_LEN_LE_CREATE_BIS** (15 + HCI_BC_LEN)
- #define **HCI_LEN_LE_TERMINATE_BIG** 2
- #define **HCI_LEN_LE_BIG_CREATE_SYNC**(numBis) (8 + HCI_BC_LEN + (numBis))
- #define **HCI_LEN_LE_BIG_TERMINATE_SYNC** 1
- #define **HCI_LEN_LE_SETUP_ISO_DATA_PATH**(ccLen) (13 + (ccLen))
- #define **HCI_LEN_LE_REMOVE_ISO_DATA_PATH** 3
- #define **HCI_LEN_LE_ISO_TX_TEST** 3
- #define **HCI_LEN_LE_ISO_RX_TEST** 3
- #define **HCI_LEN_LE_ISO_READ_TEST_COUNTERS** 2
- #define **HCI_LEN_LE_ISO_TEST_END** 2
- #define **HCI_LEN_LE_SET_HOST_FEATURE** 2
- #define **HCI_LEN_LE_DISABLE_SLAVELATENCY** 3
- #define **HCI_LEN_LE_OVERRULE_REMOTE_MAX_RX_OCTETS_AND_TIME** 6
- #define **HCI_LEN_LE_SET_TRANSMIT_POWER** 1
- #define **HCI_LEN_LE_SET_EVENT_NOTIFICATION_BIT** 1
- #define **HCI_LEN_LE_RESET_EVENT_NOTIFICATION_BIT** 1

Events

- #define **HCI_DISCONNECT_CMPL_EVT** 0x05
- #define **HCI_ENC_CHANGE_EVT** 0x08
- #define **HCI_READ_REMOTE_VER_INFO_CMPL_EVT** 0x0C
- #define **HCI_CMD_CMPL_EVT** 0x0E
- #define **HCI_CMD_STATUS_EVT** 0x0F
- #define **HCI_HW_ERROR_EVT** 0x10
- #define **HCI_NUM_CMPL_PKTS_EVT** 0x13
- #define **HCI_DATA_BUF_OVERFLOW_EVT** 0x1A
- #define **HCI_ENC_KEY_REFRESH_CMPL_EVT** 0x30
- #define **HCI_LE_META_EVT** 0x3E
- #define **HCI_AUTH_PAYLOAD_TIMEOUT_EVT** 0x57
- #define **HCI_VENDOR_SPEC_EVT** 0xFF

LE Subevents

- #define **HCI_LE_CONN_CMPL_EVT** 0x01
- #define **HCI_LE_ADV_REPORT_EVT** 0x02

- #define **HCI_LE_CONN_UPDATE_CMPL_EVT** 0x03
- #define **HCI_LE_READ_REMOTE_FEAT_CMPL_EVT** 0x04
- #define **HCI_LE_LTK_REQ_EVT** 0x05
- #define **HCI_LE_REM_CONN_PARAM_REQ_EVT** 0x06
- #define **HCI_LE_DATA_LEN_CHANGE_EVT** 0x07
- #define **HCI_LE_READ_LOCAL_P256_PUB_KEY_CMPL_EVT** 0x08
- #define **HCI_LE_GENERATE_DHKEY_CMPL_EVT** 0x09
- #define **HCI_LE_ENHANCED_CONN_CMPL_EVT** 0x0A
- #define **HCI_LE_DIRECT_ADV_REPORT_EVT** 0x0B
- #define **HCI_LE_PHY_UPDATE_CMPL_EVT** 0x0C
- #define **HCI_LE_EXT_ADV_REPORT_EVT** 0x0D
- #define **HCI_LE_PER_ADV_SYNC_EST_EVT** 0x0E
- #define **HCI_LE_PER_ADV_REPORT_EVT** 0x0F
- #define **HCI_LE_PER_ADV_SYNC_LOST_EVT** 0x10
- #define **HCI_LE_SCAN_TIMEOUT_EVT** 0x11
- #define **HCI_LE_ADV_SET_TERM_EVT** 0x12
- #define **HCI_LE_SCAN_REQ_RCVD_EVT** 0x13
- #define **HCI_LE_CH_SEL_ALGO_EVT** 0x14
- #define **HCI_LE_CONNLESS_IQ_REPORT_EVT** 0x15
- #define **HCI_LE_CONN_IQ_REPORT_EVT** 0x16
- #define **HCI_LE_CTE_REQ_FAILED_EVT** 0x17
- #define **HCI_LE_PER_SYNC_TRSF_RCVD_EVT** 0x18
- #define **HCI_LE_CIS_EST_EVT** 0x19
- #define **HCI_LE_CIS_REQ_EVT** 0x1A
- #define **HCI_LE_CREATE_BIG_CMPL_EVT** 0x1B
- #define **HCI_LE_TERMINATE_BIG_CMPL_EVT** 0x1C
- #define **HCI_LE_BIG_SYNC_EST_EVT** 0x1D
- #define **HCI_LE_BIG_SYNC_LOST_EVT** 0x1E
- #define **HCI_LE_REQ_PEER_SCA_CMPLT_EVT** 0x1F
- #define **HCI_LE_PATH_LOSS_REPORT_EVT** 0x20
- #define **HCI_LE_POWER_REPORT_EVT** 0x21
- #define **HCI_LE_BIG_INFO_ADV_REPORT_EVT** 0x22

Event parameter lengths

- #define HCI_LEN_DISCONNECT_CMPL 4
- #define HCI_LEN_READ_REMOTE_VER_INFO_CMPL 8
- #define HCI_LEN_CMD_CMPL 3
- #define HCI_LEN_CMD_STATUS 4
- #define HCI_LEN_HW_ERR 1
- #define HCI_LEN_NUM_CMPL_PKTS(numHdls) (1 + (4 ∗ numHdls))
- #define HCI_LEN_ENC_CHANGE 4
- #define HCI_LEN_ENC_KEY_REFRESH_CMPL 3
- #define HCI_LEN_LE_CONN_CMPL 19
- #define HCI_LEN_LE_ADV_RPT_MIN 12
- #define HCI_LEN_LE_CONN_UPDATE_CMPL 10
- #define HCI_LEN_LE_READ_REMOTE_FEAT_CMPL 12
- #define HCI_LEN_LE_LTK_REQ 13
- #define HCI_LEN_LE_REM_CONN_PARAM_REQ 11
- #define HCI_LEN_LE_DATA_LEN_CHANGE 11
- #define HCI_LEN_LE_READ_PUB_KEY_CMPL 66
- #define HCI_LEN_LE_GEN_DHKEY_CMPL 34
- #define HCI_LEN_LE_ENHANCED_CONN_CMPL 31
- #define HCI_LEN_LE_DIRECT_ADV_REPORT 18
- #define HCI_LEN_AUTH_PAYLOAD_TIMEOUT 2
- #define HCI_LEN_LE_PHY_UPDATE_CMPL 6
- #define HCI_LEN_LE_CH_SEL_ALGO 4
- #define HCI_LEN_LE_PHY_UPDATE_CMPL 6
- #define HCI_LEN_LE_EXT_ADV_REPORT_MIN 26
- #define HCI_LEN_LE_PER_ADV_SYNC_EST 16
- #define HCI_LEN_LE_PER_ADV_REPORT 8
- #define HCI_LEN_LE_PER_ADV_SYNC_LOST 3
- #define HCI_LEN_LE_SCAN_TIMEOUT 1

- #define HCI_LEN_LE_ADV_SET_TERM 6
- #define HCI_LEN_LE_SCAN_REQ_RCVD 9
- #define HCI_LEN_LE_PER_SYNC_TRSF_RCVT 20
- #define HCI_LEN_LE_CIS_EST 29
- #define HCI_LEN_LE_CIS_REQ 7
- #define HCI_LEN_LE_PEER_SCA_CMPL 5
- #define HCI_LEN_LE_CREATE_BIG_CMPL(numBis) (19 + (2 ∗ numBis))
- #define HCI_LEN_LE_TERMINATE_BIG_CMPL 3
- #define HCI_LEN_LE_BIG_SYNC_EST(numBis) (15 + (2 ∗ numBis))
- #define HCI_LEN_LE_BIG_SYNC_LOST 3
- #define HCI_LEN_LE_POWER_REPORT 9
- #define HCI_LEN_LE_PATH_LOSS_ZONE 5
- #define HCI_LEN_LE_BIG_INFO_ADV_REPORT 20

Supported commands

- #define HCI_SUP_DISCONNECT 0x20
- #define HCI_SUP_READ_REMOTE_VER_INFO 0x80
- #define HCI_SUP_SET_EVENT_MASK 0x40
- #define HCI_SUP_RESET 0x80
- #define HCI_SUP_READ_TX_PWR_LVL 0x04
- #define HCI_SUP_READ_LOCAL_VER_INFO 0x08
- #define HCI_SUP_READ_LOCAL_SUP_FEAT 0x20
- #define HCI_SUP_READ_BD_ADDR 0x02
- #define HCI_SUP_READ_RSSI 0x20
- #define HCI_SUP_SET_EVENT_MASK_PAGE2 0x04
- #define HCI_SUP_LE_SET_EVENT_MASK 0x01
- #define HCI_SUP_LE_READ_BUF_SIZE 0x02
- #define HCI_SUP_LE_READ_LOCAL_SUP_FEAT 0x04
- #define HCI_SUP_LE_SET_RAND_ADDR 0x10
- #define HCI_SUP_LE_SET_ADV_PARAM 0x20
- #define HCI_SUP_LE_READ_ADV_TX_POWER 0x40
- #define HCI_SUP_LE_SET_ADV_DATA 0x80
- #define HCI_SUP_LE_SET_SCAN_RESP_DATA 0x01
- #define HCI_SUP_LE_SET_ADV_ENABLE 0x02
- #define HCI_SUP_LE_SET_SCAN_PARAM 0x04
- #define HCI_SUP_LE_SET_SCAN_ENABLE 0x08
- #define HCI_SUP_LE_CREATE_CONN 0x10
- #define HCI_SUP_LE_CREATE_CONN_CANCEL 0x20
- #define HCI_SUP_LE_READ_WHITE_LIST_SIZE 0x40
- #define HCI_SUP_LE_CLEAR_WHITE_LIST 0x80
- #define HCI_SUP_LE_ADD_DEV_WHITE_LIST 0x01
- #define HCI_SUP_LE_REMOVE_DEV_WHITE_LIST 0x02
- #define HCI_SUP_LE_CONN_UPDATE 0x04
- #define HCI_SUP_LE_SET_HOST_CHAN_CLASS 0x08
- #define HCI_SUP_LE_READ_CHAN_MAP 0x10
- #define HCI_SUP_LE_READ_REMOTE_FEAT 0x20
- #define HCI_SUP_LE_ENCRYPT 0x40
- #define HCI_SUP_LE_RAND 0x80
- #define HCI_SUP_LE_START_ENCRYPTION 0x01
- #define HCI_SUP_LE_LTK_REQ_REPL 0x02
- #define HCI_SUP_LE_LTK_REQ_NEG_REPL 0x04
- #define HCI_SUP_LE_READ_SUP_STATES 0x08
- #define HCI_SUP_LE_RECEIVER_TEST 0x10
- #define HCI_SUP_LE_TRANSMITTER_TEST 0x20
- #define HCI_SUP_LE_TEST_END 0x40
- #define HCI_SUP_READ_AUTH_PAYLOAD_TO 0x10
- #define HCI_SUP_WRITE_AUTH_PAYLOAD_TO 0x20
- #define HCI_SUP_LE_REM_CONN_PARAM_REQ_REPL 0x10
- #define HCI_SUP_LE_REM_CONN_PARAM_REQ_NEG_REPL 0x20
- #define HCI_SUP_LE_SET_DATA_LEN 0x40
- #define HCI_SUP_LE_READ_DEF_DATA_LEN 0x80
- #define HCI_SUP_LE_WRITE_DEF_DATA_LEN 0x01

- #define HCI_SUP_LE_READ_LOCAL_P256_PUB_KEY 0x02
- #define HCI_SUP_LE_GENERATE_DHKEY 0x04
- #define HCI_SUP_LE_ADD_DEV_RES_LIST_EVT 0x08
- #define HCI_SUP_LE_REMOVE_DEV_RES_LIST 0x10
- #define HCI_SUP_LE_CLEAR_RES_LIST 0x20
- #define HCI_SUP_LE_READ_RES_LIST_SIZE 0x40
- #define HCI_SUP_LE_READ_PEER_RES_ADDR 0x80
- #define HCI_SUP_LE_READ_LOCAL_RES_ADDR 0x01
- #define HCI_SUP_LE_SET_ADDR_RES_ENABLE 0x02
- #define HCI_SUP_LE_SET_RES_PRIV_ADDR_TO 0x04
- #define HCI_SUP_LE_READ_MAX_DATA_LEN 0x08
- #define HCI_SUP_LE_READ_PHY 0x10
- #define HCI_SUP_LE_SET_DEF_PHY 0x20
- #define HCI_SUP_LE_SET_PHY 0x40
- #define HCI_SUP_LE_ENHANCED_RECEIVER_TEST 0x80
- #define HCI_SUP_LE_ENHANCED_TRANSMITTER_TEST 0x01
- #define HCI_SUP_LE_SET_ADV_SET_RAND_ADDR 0x02
- #define HCI_SUP_LE_SET_EXT_ADV_PARAM 0x04
- #define HCI_SUP_LE_SET_EXT_ADV_DATA 0x08
- #define HCI_SUP_LE_SET_EXT_SCAN_RESP_DATA 0x10
- #define HCI_SUP_LE_SET_EXT_ADV_ENABLE 0x20
- #define HCI_SUP_LE_READ_MAX_ADV_DATA_LEN 0x40
- #define HCI_SUP_LE_READ_NUM_OF_SUP_ADV_SETS 0x80
- #define HCI_SUP_LE_REMOVE_ADV_SET 0x01
- #define HCI_SUP_LE_CLEAR_ADV_SETS 0x02
- #define HCI_SUP_LE_SET_PER_ADV_PARAM 0x04
- #define HCI_SUP_LE_SET_PER_ADV_DATA 0x08
- #define HCI_SUP_LE_SET_PER_ADV_ENABLE 0x10
- #define HCI_SUP_LE_SET_EXT_SCAN_PARAM 0x20
- #define HCI_SUP_LE_SET_EXT_SCAN_ENABLE 0x40
- #define HCI_SUP_LE_EXT_CREATE_CONN 0x80
- #define HCI_SUP_LE_PER_ADV_CREATE_SYNC 0x01
- #define HCI_SUP_LE_PER_ADV_CREATE_SYNC_CANCEL 0x02
- #define HCI_SUP_LE_PER_ADV_TERMINATE_SYNC 0x04
- #define HCI_SUP_LE_ADD_DEV_PER_ADV_LIST 0x08
- #define HCI_SUP_LE_REMOVE_DEV_PER_ADV_LIST 0x10
- #define HCI_SUP_LE_CLEAR_PER_ADV_LIST 0x20
- #define HCI_SUP_LE_READ_PER_ADV_LIST_SIZE 0x40
- #define HCI_SUP_LE_READ_TX_POWER 0x80
- #define HCI_SUP_LE_READ_RF_PATH_COMP 0x01
- #define HCI_SUP_LE_WRITE_RF_PATH_COMP 0x02
- #define HCI_SUP_LE_SET_PRIVACY_MODE 0x04
- #define HCI_SUP_LE_RECEIVER_TEST_V3 0x08
- #define HCI_SUP_LE_TRANSMITTER_TEST_V3 0x10
- #define HCI_SUP_LE_SET_CONNLESS_CTE_TX_PARAMS 0x20
- #define HCI_SUP_LE_SET_CONNLESS_CTE_TX_ENABLE 0x40
- #define HCI_SUP_LE_SET_CONNLESS_IQ_SAMP_ENABLE 0x80
- #define HCI_SUP_LE_SET_CONN_CTE_RX_PARAMS 0x01
- #define HCI_SUP_LE_SET_CONN_CTE_TX_PARAMS 0x02
- #define HCI_SUP_LE_CONN_CTE_REQ_ENABLE 0x04
- #define HCI_SUP_LE_CONN_CTE_RSP_ENABLE 0x08
- #define HCI_SUP_LE_READ_ANTENNA_INFO 0x10
- #define HCI_SUP_LE_SET_PER_ADV_RCV_ENABLE 0x20
- #define HCI_SUP_LE_PER_ADV_SYNC_TRANSFER 0x40
- #define HCI_SUP_LE_PER_ADV_SET_INFO_TRANSFER 0x80
- #define HCI_SUP_LE_SET_PAST_PARAM 0x01
- #define HCI_SUP_LE_SET_DEFAULT_PAST_PARAM 0x02
- #define HCI_SUP_LE_GENERATE_DHKEY_V2 0x04
- #define HCI_SUP_LE_MODIFY_SLEEP_CLK_ACCURACY 0x10
- #define HCI_SUP_LE_READ_BUF_SIZE_V2 0x20
- #define HCI_SUP_LE_READ_ISO_TX_SYNC 0x40
- #define HCI_SUP_LE_SET_CIG_PARAM 0x80
- #define HCI_SUP_LE_SET_CIG_PARAM_TEST 0x01
- #define HCI_SUP_LE_CREATE_CIS 0x02

- #define HCI_SUP_LE_REMOVE_CIG 0x04
- #define HCI_SUP_LE_ACCEPT_CIS_REQ 0x08
- #define HCI_SUP_LE_REJECT_CIS_REQ 0x10
- #define HCI_SUP_LE_CREATE_BIG 0x20
- #define HCI_SUP_LE_CREATE_BIG_TEST 0x40
- #define HCI_SUP_LE_TERMINATE_BIG 0x80
- #define HCI_SUP_LE_BIG_CREATE_SYNC 0x01
- #define HCI_SUP_LE_BIG_TERMINATE_SYNC 0x02
- #define HCI_SUP_LE_REQ_PEER_SCA 0x04
- #define HCI_SUP_LE_SETUP_ISO_DATA_PATH 0x08
- #define HCI_SUP_LE_REMOVE_ISO_DATA_PATH 0x10
- #define HCI_SUP_LE_ISO_TRANSMIT_TEST 0x20
- #define HCI_SUP_LE_ISO_RECEIVE_TEST 0x40
- #define HCI_SUP_LE_ISO_READ_TEST_COUNTERS 0x80
- #define HCI_SUP_LE_ISO_TEST_END 0x01
- #define HCI_SUP_LE_SET_HOST_FEATURE 0x02
- #define HCI_SUP_LE_READ_ISO_LINK_QUALITY 0x04
- #define HCI_SUP_LE_ENH_READ_TX_POWER_LEVEL 0x08
- #define HCI_SUP_LE_READ_REMOTE_TX_POWER_LEVEL 0x01
- #define HCI_SUP_LE_SET_PATH_LOSS_REPORT_PARAM 0x02
- #define HCI_SUP_LE_SET_PATH_LOSS_REPORT_ENABLE 0x04
- #define HCI_SUP_LE_SET_TX_POWER_REPORT_ENABLE 0x08
- #define HCI_SUP_LE_TRANSMITTER_TEST_V4 0x01
- #define HCI_SUP_READ_LOCAL_SUP_CODECS_V2 0x02
- #define HCI_SUP_READ_LOCAL_SUP_CODEC_CAP 0x04
- #define HCI_SUP_READ_LOCAL_SUP_CTR_DLY 0x08
- #define HCI_SUP_CONFIG_DATA_PATH 0x10
- #define HCI_SUP_CMD_LEN 64

Event mask

- #define HCI_EVT_MASK_DISCONNECT_CMPL 0x10
- #define HCI_EVT_MASK_ENC_CHANGE 0x80
- #define HCI_EVT_MASK_READ_REMOTE_VER_INFO_CMPL 0x08
- #define HCI_EVT_MASK_HW_ERROR 0x80
- #define HCI_EVT_MASK_DATA_BUF_OVERFLOW 0x02
- #define HCI_EVT_MASK_ENC_KEY_REFRESH_CMPL 0x80
- #define HCI_EVT_MASK_LE_META 0x20

Event mask page 2

- #define HCI_EVT_MASK_AUTH_PAYLOAD_TIMEOUT 0x80

LE event mask

- #define HCI_EVT_MASK_LE_CONN_CMPL_EVT 0x01
- #define HCI_EVT_MASK_LE_ADV_REPORT_EVT 0x02
- #define HCI_EVT_MASK_LE_CONN_UPDATE_CMPL_EVT 0x04
- #define HCI_EVT_MASK_LE_READ_REMOTE_FEAT_CMPL_EVT 0x08
- #define HCI_EVT_MASK_LE_LTK_REQ_EVT 0x10
- #define HCI_EVT_MASK_LE_REMOTE_CONN_PARAM_REQ_EVT 0x20
- #define HCI_EVT_MASK_LE_DATA_LEN_CHANGE_EVT 0x40
- #define HCI_EVT_MASK_LE_READ_LOCAL_P256_PUB_KEY_CMPL 0x80
- #define HCI_EVT_MASK_LE_GENERATE_DHKEY_CMPL 0x01
- #define HCI_EVT_MASK_LE_ENHANCED_CONN_CMPL_EVT 0x02
- #define HCI_EVT_MASK_LE_DIRECT_ADV_REPORT_EVT 0x04
- #define HCI_EVT_MASK_LE_PHY_UPDATE_CMPL_EVT 0x08
- #define HCI_EVT_MASK_LE_EXT_ADV_REPORT_EVT 0x10
- #define HCI_EVT_MASK_LE_PER_ADV_SYNC_EST_EVT 0x20
- #define HCI_EVT_MASK_LE_PER_ADV_REPORT_EVT 0x40
- #define HCI_EVT_MASK_LE_PER_ADV_SYNC_LOST_EVT 0x80
- #define HCI_EVT_MASK_LE_SCAN_TIMEOUT_EVT 0x01

- #define HCI_EVT_MASK_LE_ADV_SET_TERM_EVT 0x02
- #define HCI_EVT_MASK_LE_SCAN_REQ_RCVD_EVT 0x04
- #define HCI_EVT_MASK_LE_CH_SEL_ALGO_EVT 0x08
- #define HCI_EVT_MASK_LE_CONNLESS_IQ_REPORT_EVT 0x10
- #define HCI_EVT_MASK_LE_CONN_IQ_REPORT_EVT 0x20
- #define HCI_EVT_MASK_LE_CTE_REQ_FAILED_EVT 0x40
- #define HCI_EVT_MASK_LE_PER_SYNC_TRSF_RCVT_EVT 0x80
- #define HCI_EVT_MASK_LE_CIS_EST_EVT 0x01
- #define HCI_EVT_MASK_LE_CIS_REQ_EVT 0x02
- #define HCI_EVT_MASK_LE_CREATE_BIG_CMPL_EVT 0x04
- #define HCI_EVT_MASK_LE_TERMINATE_BIG_CMPL_EVT 0x08
- #define HCI_EVT_MASK_LE_BIG_SYNC_EST_EVT 0x10
- #define HCI_EVT_MASK_LE_BIG_SYNC_LOST_EVT 0x20
- #define HCI_EVT_MASK_LE_PEER_SCA_CMPL_EVT 0x40
- #define HCI_EVT_MASK_LE_PATH_LOSS_REPORT_EVT 0x80
- #define HCI_EVT_MASK_LE_TX_POWER_REPORT_EVT 0x01
- #define HCI_EVT_MASK_LE_BIG_INFO_ADV_RPT_EVT 0x02

LE supported features

- #define HCI_LE_SUP_FEAT_ENCRYPTION 0x0000000000000001
- #define HCI_LE_SUP_FEAT_CONN_PARAM_REQ_PROC 0x0000000000000002
- #define HCI_LE_SUP_FEAT_EXT_REJECT_IND 0x0000000000000004
- #define HCI_LE_SUP_FEAT_SLV_INIT_FEAT_EXCH 0x0000000000000008
- #define HCI_LE_SUP_FEAT_LE_PING 0x0000000000000010
- #define HCI_LE_SUP_FEAT_DATA_LEN_EXT 0x0000000000000020
- #define HCI_LE_SUP_FEAT_PRIVACY 0x0000000000000040
- #define HCI_LE_SUP_FEAT_EXT_SCAN_FILT_POLICY 0x0000000000000080
- #define HCI_LE_SUP_FEAT_LE_2M_PHY 0x0000000000000100
- #define HCI_LE_SUP_FEAT_STABLE_MOD_IDX_TRANSMITTER 0x0000000000000200
- #define HCI_LE_SUP_FEAT_STABLE_MOD_IDX_RECEIVER 0x0000000000000400
- #define HCI_LE_SUP_FEAT_LE_CODED_PHY 0x0000000000000800
- #define HCI_LE_SUP_FEAT_LE_EXT_ADV 0x0000000000001000
- #define HCI_LE_SUP_FEAT_LE_PER_ADV 0x0000000000002000
- #define HCI_LE_SUP_FEAT_CH_SEL_2 0x0000000000004000
- #define HCI_LE_SUP_FEAT_LE_POWER_CLASS_1 0x0000000000008000
- #define HCI_LE_SUP_FEAT_MIN_NUN_USED_CHAN 0x0000000000010000
- #define HCI_LE_SUP_FEAT_CONN_CTE_REQ 0x0000000000020000
- #define HCI_LE_SUP_FEAT_CONN_CTE_RSP 0x0000000000040000
- #define HCI_LE_SUP_FEAT_CONNLESS_CTE_TRANS 0x0000000000080000
- #define HCI_LE_SUP_FEAT_CONNLESS_CTE_RECV 0x0000000000100000
- #define HCI_LE_SUP_FEAT_ANTENNA_SWITCH_AOD 0x0000000000200000
- #define HCI_LE_SUP_FEAT_ANTENNA_SWITCH_AOA 0x0000000000400000
- #define HCI_LE_SUP_FEAT_RECV_CTE 0x0000000000800000
- #define HCI_LE_SUP_FEAT_PAST_SENDER 0x0000000001000000
- #define HCI_LE_SUP_FEAT_PAST_RECIPIENT 0x0000000002000000
- #define HCI_LE_SUP_FEAT_SCA_UPDATE 0x0000000004000000
- #define HCI_LE_SUP_FEAT_REMOTE_PUB_KEY_VALIDATION 0x0000000008000000
- #define HCI_LE_SUP_FEAT_CIS_MASTER 0x0000000010000000
- #define HCI_LE_SUP_FEAT_CIS_SLAVE 0x0000000020000000
- #define HCI_LE_SUP_FEAT_ISO_BROADCASTER 0x0000000040000000
- #define HCI_LE_SUP_FEAT_ISO_SYNC_RECEIVER 0x0000000080000000
- #define HCI_LE_SUP_FEAT_ISO_HOST_SUPPORT 0x0000000100000000
- #define HCI_LE_SUP_FEAT_POWER_CONTROL_REQUEST 0x0000000200000000
- #define HCI_LE_SUP_FEAT_POWER_CHANGE_IND 0x0000000400000000
- #define HCI_LE_SUP_FEAT_PATH_LOSS_MONITOR 0x0000000800000000

LE feature bit positon in FeatureSet stored in the Controller

- #define HCI_LE_FEAT_BIT_ISO_HOST_SUPPORT 32

Advertising command parameters

- #define HCI_ADV_MIN_INTERVAL 0x0020
- #define HCI_ADV_MAX_INTERVAL 0x4000
- #define HCI_ADV_DIRECTED_MAX_DURATION 0x0500
- #define HCI_ADV_TYPE_CONN_UNDIRECT 0x00
- #define HCI_ADV_TYPE_CONN_DIRECT 0x01
- #define HCI_ADV_TYPE_DISC_UNDIRECT 0x02
- #define HCI_ADV_TYPE_NONCONN_UNDIRECT 0x03
- #define HCI_ADV_TYPE_CONN_DIRECT_LO_DUTY 0x04
- #define HCI_ADV_CHAN_37 0x01
- #define HCI_ADV_CHAN_38 0x02
- #define HCI_ADV_CHAN_39 0x04
- #define HCI_ADV_FILT_NONE 0x00
- #define HCI_ADV_FILT_SCAN 0x01
- #define HCI_ADV_FILT_CONN 0x02
- #define HCI_ADV_FILT_ALL 0x03

Scan command parameters

- #define HCI_SCAN_TYPE_PASSIVE 0
- #define HCI_SCAN_TYPE_ACTIVE 1
- #define HCI_SCAN_INTERVAL_MIN 0x0004
- #define HCI_SCAN_INTERVAL_MAX 0x4000
- #define HCI_SCAN_INTERVAL_DEFAULT 0x0010
- #define HCI_SCAN_WINDOW_MIN 0x0004
- #define HCI_SCAN_WINDOW_MAX 0x4000
- #define HCI_SCAN_WINDOW_DEFAULT 0x0010

Connection command parameters

- #define HCI_CONN_INTERVAL_MIN 0x0006
- #define HCI_CONN_INTERVAL_MAX 0x0C80
- #define HCI_CONN_LATENCY_MAX 0x01F3
- #define HCI_SUP_TIMEOUT_MIN 0x000A
- #define HCI_SUP_TIMEOUT_MAX 0x0C80

Misc command parameters

- #define HCI_ROLE_MASTER 0
- #define HCI_ROLE_SLAVE 1
- #define HCI_READ_TX_PWR_CURRENT 0
- #define HCI_READ_TX_PWR_MAX 1
- #define HCI_TX_PWR_MIN -30
- #define HCI_TX_PWR_MAX 20
- #define HCI_TX_PWR_NO_PREFERENCE 127
- #define HCI_VERSION 6
- #define HCI_RSSI_MIN -127
- #define HCI_RSSI_MAX 20
- #define HCI_ADDR_TYPE_PUBLIC 0
- #define HCI_ADDR_TYPE_RANDOM 1
- #define HCI_ADDR_TYPE_PUBLIC_IDENTITY 2
- #define HCI_ADDR_TYPE_RANDOM_IDENTITY 3
- #define HCI_ADDR_TYPE_ANONYMOUS 0xFF
- #define HCI_FILT_NONE 0
- #define HCI_FILT_WHITE_LIST 1
- #define HCI_FILT_RES_INIT 2
- #define HCI_FILT_WHITE_LIST_RES_INIT 3
- #define HCI_FILT_PER_ADV_PARAM 0
- #define HCI_FILT_PER_ADV_LIST 1
- #define HCI_ROLE_MASTER 0
- #define HCI_ROLE_SLAVE 1
- #define HCI_PRIV_MODE_NETWORK 0x00
- #define HCI_PRIV_MODE_DEVICE 0x01

Connection event parameters

- #define HCI_CLOCK_500PPM 0x00
- #define HCI_CLOCK_250PPM 0x01
- #define HCI_CLOCK_150PPM 0x02
- #define HCI_CLOCK_100PPM 0x03
- #define HCI_CLOCK_75PPM 0x04
- #define HCI_CLOCK_50PPM 0x05
- #define HCI_CLOCK_30PPM 0x06
- #define HCI_CLOCK_20PPM 0x07

Advertising report event parameters

- #define HCI_ADV_CONN_UNDIRECT 0x00
- #define HCI_ADV_CONN_DIRECT 0x01
- #define HCI_ADV_DISC_UNDIRECT 0x02
- #define HCI_ADV_NONCONN_UNDIRECT 0x03
- #define HCI_ADV_SCAN_RESPONSE 0x04

Extended advertising data operations

- #define HCI_ADV_DATA_OP_FRAG_INTER 0x00
- #define HCI_ADV_DATA_OP_FRAG_FIRST 0x01
- #define HCI_ADV_DATA_OP_FRAG_LAST 0x02
- #define HCI_ADV_DATA_OP_COMP_FRAG 0x03
- #define HCI_ADV_DATA_OP_UNCHANGED_DATA 0x04

Advertising data fragment preference

- #define HCI_ADV_DATA_FRAG_PREF_FRAG 0x00
- #define HCI_ADV_DATA_FRAG_PREF_NO_FRAG 0x01

Number of advertising sets

- #define HCI_ADV_NUM_SETS_ALL_DISABLE 0x00

Maximum number of scanning or initiating PHYs

- #define HCI_MAX_NUM_PHYS 3

Advertising PHY values

- #define HCI_ADV_PHY_LE_1M 0x01
- #define HCI_ADV_PHY_LE_2M 0x02
- #define HCI_ADV_PHY_LE_CODED 0x03

Scanner PHY value bits

- #define HCI_SCAN_PHY_LE_1M_BIT (1<<0)
- #define HCI_SCAN_PHY_LE_2M_BIT (1<<1)
- #define HCI_SCAN_PHY_LE_CODED_BIT (1<<2)

Initiator PHY value bits

- #define HCI_INIT_PHY_LE_1M_BIT (1<<0)
- #define HCI_INIT_PHY_LE_2M_BIT (1<<1)
- #define HCI_INIT_PHY_LE_CODED_BIT (1<<2)

Transmitter PHY value bits

- #define HCI_TRANS_PHY_LE_1M_BIT (1<<0)
- #define HCI_TRANS_PHY_LE_2M_BIT (1<<1)
- #define HCI_TRABS_PHY_LE_CODED_BIT (1<<2)

Advertising event properties type bits

- #define HCI_ADV_PROP_CONN_ADV_BIT (1<<0)
- #define HCI_ADV_PROP_SCAN_ADV_BIT (1<<1)
- #define HCI_ADV_PROP_DIRECT_ADV_BIT (1<<2)
- #define HCI_ADV_PROP_CONN_DIRECT_ADV_BIT (1<<3)
- #define HCI_ADV_PROP_USE_LEG_PDU_BIT (1<<4)
- #define HCI_ADV_PROP_OMIT_ADV_ADDR_BIT (1<<5)
- #define HCI_ADV_PROP_INC_TX_PWR_BIT (1<<6)

Advertising event properties for legacy PDUs

- #define HCI_ADV_PROP_LEG_CONN_UNDIRECT 0x13
- #define HCI_ADV_PROP_LEG_CONN_DIRECT 0x1D
- #define HCI_ADV_PROP_LEG_SCAN_UNDIRECT 0x12
- #define HCI_ADV_PROP_LEG_NONCONN_UNDIRECT 0x10
- #define HCI_ADV_PROP_LEG_CONN_DIRECT_LO_DUTY 0x15

Extended advertising report event type bits

- #define HCI_ADV_RPT_CONN_ADV_BIT (1<<0)
- #define HCI_ADV_RPT_SCAN_ADV_BIT (1<<1)
- #define HCI_ADV_RPT_DIRECT_ADV_BIT (1<<2)
- #define HCI_ADV_RPT_SCAN_RSP_BIT (1<<3)
- #define HCI_ADV_RPT_LEG_ADV_BIT (1<<4)
- #define HCI_ADV_RPT_DATA_STATUS_BITS (3<<5)

Advertising report event types for legacy PDUs

- #define HCI_ADV_RPT_LEG_CONN_UNDIRECT 0x13
- #define HCI_ADV_RPT_LEG_CONN_DIRECT 0x15
- #define HCI_ADV_RPT_LEG_SCAN_UNDIRECT 0x12
- #define HCI_ADV_RPT_LEG_NONCONN_UNDIRECT 0x10
- #define HCI_ADV_RPT_LEG_CONN_UNDIRECT_SCAN_RSP 0x1B
- #define HCI_ADV_RPT_LEG_SCAN_UNDIRECT_SCAN_RSP 0x1A

Advertising report data status

- #define HCI_ADV_RPT_DATA_CMPL 0x00
- #define HCI_ADV_RPT_DATA_INCMPL_MORE 0x01
- #define HCI_ADV_RPT_DATA_INCMPL_TRUNC 0x02

Extended advertising report event primary PHY values

- #define HCI_ADV_RPT_PHY_PRIM_LE_1M 0x01
- #define HCI_ADV_RPT_PHY_PRIM_LE_CODED 0x03

Extended advertising report event seconday PHY values

- #define HCI_ADV_RPT_PHY_SEC_NONE 0x00
- #define HCI_ADV_RPT_PHY_SEC_LE_1M 0x01
- #define HCI_ADV_RPT_PHY_SEC_LE_2M 0x02

- #define HCI_ADV_RPT_PHY_SEC_LE_CODED 0x03

Channel selection algorithm used

- #define HCI_CH_SEL_ALGO_1 0x00
- #define HCI_CH_SEL_ALGO_2 0x01

KeyType parameters

- #define HCI_PRIVATE_KEY_GENERATED 0x00
- #define HCI_PRIVATE_KEY_DEBUG 0x01

Minimum number of used channels

- #define HCI_MIN_NUM_OF_USED_CHAN 8

Synchronization timeout for the periodic advertising

- #define HCI_SYNC_MIN_TIMEOUT 0x000A
- #define HCI_SYNC_MAX_TIMEOUT 0x4000

Maximum synchronization skip

- #define HCI_SYNC_MAX_SKIP 0x01F3

Maximum synchronization handle

- #define HCI_SYNC_MAX_HANDLE 0x0EFF

Periodic sync transfer receive mode

- #define HCI_SYNC_TRSF_MODE_OFF 0x00
- #define HCI_SYNC_TRSF_MODE_REP_DISABLED 0x01,
- #define HCI_SYNC_TRSF_MODE_REP_ENABLED 0x02,

Periodic advertising create sync options bits

- #define HCI_OPTIONS_FILT_POLICY_BIT (1<<0)
- #define HCI_OPTIONS_INIT_RPT_ENABLE_BIT (1<<1)

PHY types

- #define HCI_PHY_NONE 0x00
- #define HCI_PHY_LE_1M_BIT (1<<0)
- #define HCI_PHY_LE_2M_BIT (1<<1)
- #define HCI_PHY_LE_CODED_BIT (1<<2)

All PHYs preference

- #define HCI_ALL_PHY_ALL_PREFERENCES 0x00
- #define HCI_ALL_PHY_TX_PREFERENCE_BIT (1<<0)
- #define HCI_ALL_PHY_RX_PREFERENCE_BIT (1<<1)

PHY options

- #define HCI_PHY_OPTIONS_NONE 0x00
- #define HCI_PHY_OPTIONS_S2_PREFERRED 0x01
- #define HCI_PHY_OPTIONS_S8_PREFERRED 0x02

CTE Slot Durations

- #define HCI_CTE_SLOT_DURATION_NONE 0x00
- #define HCI_CTE_SLOT_DURATION_1_US 0x01
- #define HCI_CTE_SLOT_DURATION_2_US 0x02

Permitted CTE Type bits

- #define HCI_CTE_TYPE_PERMIT_AOA_RSP_BIT (1<<0)
- #define HCI_CTE_TYPE_PERMIT_AOD_RSP_1_US_BIT (1<<1)
- #define HCI_CTE_TYPE_PERMIT_AOD_RSP_2_US_BIT (1<<2)

Requested CTE Types

- #define HCI_CTE_TYPE_REQ_AOA 0x00
- #define HCI_CTE_TYPE_REQ_AOD_1_US 0x01
- #define HCI_CTE_TYPE_REQ_AOD_2_US 0x02

Bluetooth core specification versions

- #define HCI_VER_BT_CORE_SPEC_4_0 0x06
- #define HCI_VER_BT_CORE_SPEC_4_1 0x07
- #define HCI_VER_BT_CORE_SPEC_4_2 0x08
- #define HCI_VER_BT_CORE_SPEC_5_0 0x09
- #define HCI_VER_BT_CORE_SPEC_5_1 0x0A
- #define HCI_VER_BT_CORE_SPEC_5_2 0x0B

Parameter lengths

- #define HCI_EVT_MASK_LEN 8
- #define HCI_EVT_MASK_PAGE_2_LEN 8
- #define HCI_LE_EVT_MASK_LEN 8
- #define HCI_FEAT_LEN 8
- #define HCI_ADV_DATA_LEN 31
- #define HCI_SCAN_DATA_LEN 31
- #define HCI_EXT_ADV_DATA_LEN 251
- #define HCI_EXT_ADV_CONN_DATA_LEN 191
- #define HCI_PER_ADV_DATA_LEN 252
- #define HCI_EXT_ADV_RPT_DATA_LEN 229
- #define HCI_PER_ADV_RPT_DATA_LEN 247
- #define HCI_CHAN_MAP_LEN 5
- #define HCI_KEY_LEN 16
- #define HCI_ENCRYPT_DATA_LEN 16
- #define HCI_RAND_LEN 8
- #define HCI_LE_STATES_LEN 8
- #define HCI_P256_KEY_LEN 64
- #define HCI_DH_KEY_LEN 32
- #define HCI_BC_LEN 16
- #define HCI_EXT_ADV_RPT_DATA_LEN_OFFSET 23
- #define HCI_PER_ADV_RPT_DATA_LEN_OFFSET 6

Number of Antenna IDs in Switching Pattern

- #define HCI_MIN_NUM_ANTENNA_IDS 2
- #define HCI_MAX_NUM_ANTENNA_IDS 75

IQ Report Sample Counts

- #define HCI_IQ_RPT_SAMPLE_CNT_MIN 9
- #define HCI_IQ_RPT_SAMPLE_CNT_MAX 82
- #define HCI_CONN_IQ_RPT_SAMPLE_CNT_OFFSET 12

CIS Count

- #define HCI_MAX_CIS_COUNT 0x10

BIS Count

- #define HCI_MAX_BIS_COUNT 0x10

CIG IDs

- #define HCI_MIN_CIG_ID 0x00
- #define HCI_MAX_CIG_ID 0xEF

CIS IDs

- #define HCI_MIN_CIS_ID 0x00
- #define HCI_MAX_CIS_ID 0xEF

Packing Scheme

- #define HCI_PACKING_SEQUENTIAL 0x00
- #define HCI_PACKING_INTERLEAVED 0x01

Framing

- #define HCI_FRAMING_UNFRAMED 0x00
- #define HCI_FRAMING_FRAMED 0x01

Slave Clock Accuracy

- #define HCI_MIN_SCA 0x00
- #define HCI_MAX_SCA 0x07

SDU Size

- #define HCI_MIN_SDU_SIZE 0x0000
- #define HCI_MAX_SDU_SIZE 0x0FFF

SDU Interval

- #define HCI_MIN_SDU_INTERV 0x0000FF
- #define HCI_MAX_SDU_INTERV 0x0FFFFF
- #define HCI_DEFAULT_SDU_INTERV 0x004E20

CIS Transport Latency

- #define HCI_MIN_CIS_TRANS_LAT 0x0005
- #define HCI_MAX_CIS_TRANS_LAT 0x0FA0
- #define HCI_DEFAULT_CIS_TRANS_LAT 0x0028

CIS Flush Time

- #define HCI_MIN_CIS_FT 0x01
- #define HCI_MAX_CIS_FT 0xFF

CIS Burst Number

- #define HCI_MIN_CIS_BN 0x00
- #define HCI_MAX_CIS_BN 0x0F

CIS Retransmission Number

- #define HCI_MIN_CIS_RTN 0x00
- #define HCI_MAX_CIS_RTN 0x0F

ISO Data Path Direction

- #define HCI_ISO_DATA_DIR_INPUT 0
- #define HCI_ISO_DATA_DIR_OUTPUT 1

ISO Data Path Direction Bit

- #define HCI_ISO_DATA_PATH_INPUT_BIT (1<<HCI_ISO_DATA_DIR_INPUT)
- #define HCI_ISO_DATA_PATH_OUTPUT_BIT (1<<HCI_ISO_DATA_DIR_OUTPUT)

ISO Data Path ID

- #define HCI_ISO_DATA_PATH_HCI 0x00
- #define HCI_ISO_DATA_PATH_VS 0x01
- #define HCI_ISO_DATA_PATH_DISABLED 0xFF

ISO test packet payload type

- #define HCI_ISO_ISO_PLD_TYPE_ZERO_LEN 0x00
- #define HCI_ISO_ISO_PLD_TYPE_VAR_LEN 0x01
- #define HCI_ISO_ISO_PLD_TYPE_MAX_LEN 0x02

Maximum number of codecs

- #define HCI_MAX_CODEC 5

Maximum length of codec-specific capability data

- #define HCI_CODEC_CAP_DATA_LEN 4

Codec transport types

- #define HCI_CODEC_TRANS_CIS_BIT (1<<2)
- #define HCI_CODEC_TRANS_BIS_BIT (1<<3)

ISO Header Packet Boundary

- #define HCI_ISO_HDR_PB_START_FRAG 0x00
- #define HCI_ISO_HDR_PB_CONT_FRAG 0x01
- #define HCI_ISO_HDR_PB_COMP_FRAG 0x02

- #define HCI_ISO_HDR_PB_END_FRAG 0x03

ISOAL Segmentation Header Start/Continuation Bit

- #define HCI_ISOAL_SEG_HDR_SC_START 0x00
- #define HCI_ISOAL_SEG_HDR_SC_CONT 0x01

Company ID

- #define HCI_ID_PACKETCRAFT 0x07E8
- #define HCI_ID_GREENPEAK 0x0453

  Greenpeak company ID.

Manufacturer location in Local version

- #define HCI_LOCAL_VER_MANUFACTURER_POS 4

Coding Format Assigned Numbers

- #define HCI_ID_LC3 0x01
- #define HCI_ID_VS 0xFF
- #define HCI_CODEC_TRANSPORT_CIS 0x02
- #define HCI_CODEC_TRANSPORT_BIS 0x03

4.1.1 Detailed Description

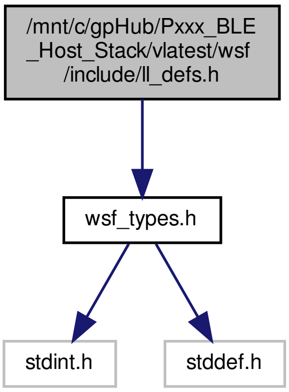
HCI constants and definitions from the Bluetooth specification.

4.2 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/ll_defs.h File Reference

Link layer constant definitions.

```
#include "wsf_types.h"
```

Include dependency graph for ll_defs.h:



Macros

- #define LL_VER_BT_CORE_SPEC_4_0 0x06
- #define LL_VER_BT_CORE_SPEC_4_1 0x07
- #define LL_VER_BT_CORE_SPEC_4_2 0x08
- #define LL_VER_BT_CORE_SPEC_5_0 0x09
- #define LL_VER_BT_CORE_SPEC_5_1 0x0A
- #define LL_VER_BT_CORE_SPEC_5_2 0x0B
- #define LL_VER_BT_CORE_SPEC_SYDNEY 0x0C
- #define LL_RSSI_MIN -127
- #define LL_RSSI_MAX 20
- #define LL_RSSI_NOT_AVAIL 127
- #define LL_CRC_LEN 3
- #define LL_AA_LEN 4
- #define LL_PREAMBLE_LEN_1M 1
- #define LL_PREAMBLE_LEN_2M 2
- #define LL_PREAMBLE_LEN_CODED_BITS 10
- #define LL_CI_LEN_BITS 2
- #define LL_TERM1_LEN_BITS 3
- #define LL_TERM2_LEN_BITS 3
- #define LL_RAND_ADDR_TYPE_MASK UINT64_C(0xC00000000000)
- #define LL_RAND_ADDR_TYPE_STATIC UINT64_C(0xC00000000000)
- #define LL_RAND_ADDR_TYPE_RPA UINT64_C(0x400000000000)
- #define LL_RAND_ADDR_TYPE_NRPA UINT64_C(0x000000000000)
- #define LL_SCAN_REQ_PDU_LEN 12
- #define LL_CONN_IND_PDU_LEN 34
- #define LL_CONN_RSP_PDU_LEN 14

- #define LL_CHAN_ADV_MIN_IDX 37
- #define LL_CHAN_ADV_MAX_IDX 39
- #define LL_NUM_CHAN_ADV 3
- #define LL_ADVBU_MAX_LEN 31
- #define LL_ADVB_MAX_LEN 39
- #define LL_ADVB_MIN_LEN (LL_ADVB_MAX_LEN - LL_ADVBU_MAX_LEN)
- #define LL_ADVB_MAX_TIME_1M ((LL_BLE_US_PER_BYTE_1M ∗ (LL_ADVB_MAX_LEN - LL_ADV_HDR_LEN)) + LL_MIN_PKT_TIME_US_1M)
- #define LL_ADVB_MAX_TIME_2M ((LL_BLE_US_PER_BYTE_2M ∗ (LL_ADVB_MAX_LEN - LL_ADV_HDR_LEN)) + LL_MIN_PKT_TIME_US_2M)
- #define LL_ADVB_MAX_TIME_S2 ((LL_BLE_US_PER_BYTE_CODED_S2 ∗ (LL_ADVB_MAX_LEN - LL_ADV_HDR_LEN)) + LL_MIN_PKT_TIME_US_CODED_S2)
- #define LL_ADVB_MAX_TIME_S8 ((LL_BLE_US_PER_BYTE_CODED_S8 ∗ (LL_ADVB_MAX_LEN - LL_ADV_HDR_LEN)) + LL_MIN_PKT_TIME_US_CODED_S8)
- #define LL_ADV_PKT_MAX_USEC LL_ADVB_MAX_TIME_1M
- #define LL_SCAN_REQ_MAX_USEC ((8 ∗ (LL_ADV_PREFIX_LEN + LL_SCAN_PREFIX_LEN)) + LL_MIN_PKT_TIME_US_1M)
- #define LL_SCAN_RSP_MAX_USEC LL_ADVB_MAX_TIME_1M
- #define LL_ADV_HDR_LEN 2
- #define LL_ADV_HDR_TYPE_OFFS 0
- #define LL_ADV_HDR_TYPE_MSK 0x0F
- #define LL_ADV_HDR_LEN_OFFS 1
- #define LL_ADV_HDR_LEN_MSK 0x3F
- #define LL_ADV_HDR_CP_MSK 0x40
- #define LL_ADV_EXT_HDR_LEN_MSK 0xFF
- #define LL_ADV_PREFIX_LEN 6
- #define LL_SCAN_PREFIX_LEN 6
- #define LL_ADV_ACCESS_ADDR UINT32_C(0x8E89BED6)
- #define LL_ADV_CRC_INIT UINT32_C(0x555555)
- #define LL_DIR_ADV_INTER_TICKS 6
- #define LL_DIR_ADV_DUR_TICKS 2048
- #define LL_MAX_ADV_HANDLE 0xEF
- #define LL_MAX_ADV_SID 0x0F
- #define LL_EXT_ADV_HDR_MIN_LEN 1
- #define LL_EXT_ADV_HDR_MAX_LEN 64
- #define LL_EXT_HDR_FLAG_LEN 1
- #define LL_EXT_ADVBU_MAX_LEN 251
- #define LL_EXT_ADVB_MAX_LEN 257
- #define LL_EXT_ADVB_NORMAL_LEN 50
- #define LL_EXT_HDR_ACAD_MAX_LEN LL_EXT_ADV_HDR_MAX_LEN - LL_EXT_ADV_HDR_MIN_LEN - LL_EXT_HDR_FLAG_LEN
- #define LL_EXT_ADVB_MAX_TIME_1M ((LL_BLE_US_PER_BYTE_1M ∗ (LL_EXT_ADVB_MAX_LEN - LL_ADV_HDR_LEN)) + LL_MIN_PKT_TIME_US_1M)
- #define LL_EXT_ADVB_MAX_TIME_2M ((LL_BLE_US_PER_BYTE_2M ∗ (LL_EXT_ADVB_MAX_LEN - LL_ADV_HDR_LEN)) + LL_MIN_PKT_TIME_US_2M)
- #define LL_EXT_ADVB_MAX_TIME_S2 ((LL_BLE_US_PER_BYTE_CODED_S2 ∗ (LL_EXT_ADVB_MAX_LEN - LL_ADV_HDR_LEN)) + LL_MIN_PKT_TIME_US_CODED_S2)
- #define LL_EXT_ADVB_MAX_TIME_S8 ((LL_BLE_US_PER_BYTE_CODED_S8 ∗ (LL_EXT_ADVB_MAX_LEN - LL_ADV_HDR_LEN)) + LL_MIN_PKT_TIME_US_CODED_S8)
- #define LL_EXT_ADVB_NORMAL_TIME_S8 ((LL_BLE_US_PER_BYTE_CODED_S8 ∗ (LL_EXT_ADVB_NORMAL_LEN - LL_ADV_HDR_LEN)) + LL_MIN_PKT_TIME_US_CODED_S8)
- #define LL_AUX_PTR_MAX_USEC 2457300
- #define LL_SYNC_MIN_TIMEOUT 0x000A
- #define LL_SYNC_MAX_TIMEOUT 0x4000
- #define LL_SYNC_MAX_SKIP 0x01F3

- #define LL_SYNC_MAX_HANDLE 0x0EFF
- #define LL_PER_ADV_INT_MIN 0x0006
- #define LL_SYNC_OFFS_ADJUST_USEC LL_AUX_PTR_MAX_USEC
- #define LL_SYNC_INFO_LEN 18
- #define LL_CONN_UPD_IND_PDU_LEN 12
- #define LL_CHAN_MAP_IND_PDU_LEN 8
- #define LL_TERMINATE_IND_PDU_LEN 2
- #define LL_ENC_REQ_LEN 23
- #define LL_ENC_RSP_LEN 13
- #define LL_START_ENC_LEN 1
- #define LL_UNKNOWN_RSP_LEN 2
- #define LL_FEATURE_PDU_LEN 9
- #define LL_PAUSE_ENC_LEN 1
- #define LL_VERSION_IND_PDU_LEN 6
- #define LL_REJECT_IND_PDU_LEN 2
- #define LL_CONN_PARAM_PDU_LEN 24
- #define LL_REJECT_EXT_IND_PDU_LEN 3
- #define LL_PING_PDU_LEN 1
- #define LL_DATA_LEN_PDU_LEN 9
- #define LL_PHY_PDU_LEN 3
- #define LL_PHY_UPD_IND_PDU_LEN 5
- #define LL_MIN_USED_CHAN_PDU_LEN 3
- #define LL_PERIODIC_SYNC_PDU_LEN 35
- #define LL_PEER_SCA_REQ_LEN 2
- #define LL_PEER_SCA_RSP_LEN 2
- #define LL_CIS_REQ_LEN 36
- #define LL_CIS_RSP_LEN 9
- #define LL_CIS_IND_LEN 16
- #define LL_CIS_TERM_LEN 4
- #define LL_CIS_SDU_CONFIG_REQ_LEN 13
- #define LL_CIS_SDU_CONFIG_RSP_LEN 4
- #define LL_PWR_CTRL_REQ_LEN 4
- #define LL_PWR_CTRL_RSP_LEN 5
- #define LL_PWR_CHANGE_IND_LEN 5
- #define LL_EMPTY_PDU_LEN 2
- #define LL_DATA_HDR_LEN 2
- #define LL_DATA_HDR_MAX_LEN 4
- #define LL_DATA_MIC_LEN 4
- #define LL_DATA_HDR_LLID_MSK 0x03
- #define LL_DATA_HDR_LEN_MSK 0xFF
- #define LL_MAX_NUM_CHAN_DATA 37
- #define LL_MIN_NUM_CHAN_DATA 2
- #define LL_ECC_KEY_LEN 32
- #define LL_DEF_RES_ADDR_TO_SEC 900
- #define LL_RAND_LEN 8
- #define LL_KEY_LEN 16
- #define LL_SKD_LEN LL_KEY_LEN
- #define LL_IV_LEN 8
- #define LL_BC_LEN LL_KEY_LEN
- #define LL_GIV_LEN 8
- #define LL_GSKD_LEN 16
- #define LL_DEF_AUTH_TO_MS 30000
- #define LL_DATA_LEN_TO_TIME_1M(len, enc) ((LL_BLE_US_PER_BYTE_1M ∗ ((len) + ((enc) ? LL_DATA_MIC_LEN : 0))) + LL_MIN_PKT_TIME_US_1M)

- #define LL_DATA_LEN_TO_TIME_2M(len, enc) ((LL_BLE_US_PER_BYTE_2M ∗ ((len) + ((enc) ? LL_DATA_MIC_LEN : 0))) + LL_MIN_PKT_TIME_US_2M)
- #define LL_DATA_LEN_TO_TIME_CODED_S8(len, enc) ((LL_BLE_US_PER_BYTE_CODED_S8 ∗ ((len) + ((enc) ? LL_DATA_MIC_LEN : 0))) + LL_MIN_PKT_TIME_US_CODED_S8)
- #define LL_DATA_LEN_TO_TIME_CODED_S2(len, enc) ((LL_BLE_US_PER_BYTE_CODED_S2 ∗ ((len) + ((enc) ? LL_DATA_MIC_LEN : 0))) + LL_MIN_PKT_TIME_US_CODED_S2)
- #define LL_MIN_INSTANT 6
- #define LL_MAX_ADV_DATA_LEN 1650
- #define LL_MAX_DATA_LEN_MIN 27
- #define LL_MAX_DATA_LEN_ABS_MAX 251
- #define LL_MAX_DATA_TIME_MIN 328
- #define LL_MAX_DATA_TIME_ABS_MAX 17040
- #define LL_MAX_DATA_TIME_ABS_MAX_1M 2128
- #define LL_MAX_DATA_TIME_ABS_MIN_CODED 2704
- #define LL_T_PRT_SEC 40
- #define LL_MAX_ADV_DLY_MS 10
- #define LL_MIN_CONN_INTERVAL 6
- #define LL_MAX_CONN_INTERVAL 3200
- #define LL_MIN_TX_WIN_SIZE 1
- #define LL_MAX_TX_WIN_SIZE 8
- #define LL_MAX_CONN_LATENCY 499
- #define LL_MIN_SUP_TIMEOUT 10
- #define LL_MAX_SUP_TIMEOUT 3200
- #define LL_MIN_POWER_THRESHOLD -128
- #define LL_MAX_POWER_THRESHOLD 127
- #define LL_MAX_PHYS 3
- #define LL_ALL_PHYS_MSK 0x7
- #define LL_ISO_DATA_HDR_LEN 2
- #define LL_ISO_DATA_PLD_MAX_LEN 251
- #define LL_ISO_PDU_MAX_LEN (HCI_ISO_HDR_LEN + BB_DATA_PLD_MAX_LEN + BB_DATA_PDU_TAILROOM)
- #define LL_ISO_SEG_HDR_LEN 2
- #define LL_ISO_SEG_TO_LEN 3
- #define LL_MAX_CIS_COUNT 0x10
- #define LL_MIN_CIG_ID 0x00
- #define LL_MAX_CIG_ID 0xEF
- #define LL_MIN_CIS_ID 0x00
- #define LL_MAX_CIS_ID 0xEF
- #define LL_MIN_ISO_INTERV 0x0004
- #define LL_MAX_ISO_INTERV 0x0C80
- #define LL_MIN_ISOAL_PDU_TYPE 0x00
- #define LL_MAX_ISOAL_PDU_TYPE 0x01
- #define LL_MIN_SDU_SIZE 0x000
- #define LL_MAX_SDU_SIZE 0xFFF
- #define LL_MIN_SDU_INTERV 0x000FF
- #define LL_MAX_SDU_INTERV 0xFFFFF
- #define LL_MIN_CIS_NSE 0x01
- #define LL_MAX_CIS_NSE 0x1F
- #define LL_MIN_CIS_PL 0x0000
- #define LL_MAX_CIS_PL 0x0FFB
- #define LL_MIN_CIS_TRANS_LAT 0x0005
- #define LL_MAX_CIS_TRANS_LAT 0x0FA0
- #define LL_MIN_CIS_PHY_BIT 0x00
- #define LL_MAX_CIS_PHY_BIT 0x02
- #define LL_MIN_CIS_FT 0x01

- #define LL_MAX_CIS_FT 0x1F
- #define LL_MIN_CIS_BN 0x00
- #define LL_MAX_CIS_BN 0x10
- #define LL_MIN_CIS_RTN 0x00
- #define LL_MAX_CIS_RTN 0x0F
- #define LL_ISO_TEST_VAR_MIN_LEN 4
- #define LL_ISO_TRANSPORT_LAT_MIN 0x0000EA
- #define LL_DTM_HDR_LEN 2
- #define LL_DTM_SYNC_WORD UINT32_C(0x71764129)
- #define LL_DTM_CRC_INIT UINT32_C(0x555555)
- #define LL_DTM_MAX_INT_US 12500
- #define LL_DTM_PDU_ABS_MAX_LEN 255
- #define LL_DTM_MAX_CHAN_IDX 39
- #define LL_CHAN_DATA_MIN_IDX 0
- #define LL_CHAN_DATA_MAX_IDX 36
- #define LL_CHAN_DATA_ALL UINT64_C(0x0000001FFFFFFFFF)
- #define LL_BLE_BIT_PER_US 1
- #define LL_BLE_US_PER_BYTE_1M 8
- #define LL_BLE_US_PER_BYTE_2M 4
- #define LL_BLE_US_PER_BYTE_CODED_S8 64
- #define LL_BLE_US_PER_BIT_CODED_S8 8
- #define LL_BLE_US_PER_BYTE_CODED_S2 16
- #define LL_BLE_US_PER_BIT_CODED_S2 2
- #define LL_BLE_TIFS_US 150
- #define LL_BLE_MAFS_US 300
- #define LL_BLE_US_PER_TICK 625
- #define LL_BLE_TMSS_US 150
- #define LL_MIN_PKT_TIME_US_1M 80
- #define LL_MIN_PKT_TIME_US_2M 44
- #define LL_MIN_PKT_TIME_US_CODED_S8 720
- #define LL_MIN_PKT_TIME_US_CODED_S2 462
- #define LL_MIN_ADV_TX_PWR_LVL -20
- #define LL_MAX_ADV_TX_PWR_LVL 10
- #define LL_MIN_TX_PWR_LVL -30
- #define LL_MAX_TX_PWR_LVL 20
- #define LL_MAX_TIFS_DEVIATION 2
- #define LL_WW_RX_DEVIATION_USEC 16
- #define LL_30_USEC_OFFS_MAX_USEC 245730
- #define LL_ACAD_OPCODE_LEN 1
- #define LL_ACAD_LEN_FIELD_LEN 1
- #define LL_ACAD_DATA_FIELD_MAX_LEN (LL_EXT_HDR_ACAD_MAX_LEN - LL_ACAD_OPCODE_L↩
EN - LL_ACAD_LEN_FIELD_LEN)
- #define LL_ACAD_CHAN_MAP_UPD_LEN 8
- #define LL_ACAD_BIG_INFO_UNENCRPT_LEN 33
- #define LL_ACAD_BIG_INFO_ENCRPT_LEN 57
- #define LL_BIG_OPCODE_LEN 1
- #define LL_BIG_CHAN_MAP_IND_PDU_LEN 7
- #define LL_BIG_TERMINATE_IND_PDU_LEN 3
- #define LL_BIG_MIN_INSTANT 6
- #define LL_BIG_CONTROL_ACCESS_ADDR UINT32_C(0x7A412493)
- #define LL_SCA_MIN_INDEX 0
- #define LL_SCA_MAX_INDEX 7
- #define LL_PWR_CONTROL_LIMIT_MIN_BIT (1 << 0)

  Power control Limit field.

- #define LL_PWR_CONTROL_LIMIT_MAX_BIT (1 << 1)
- #define LL_PWR_CTRL_APR_UNDEF 0xFF

  Power control APR field.
- #define LL_PWR_CTRL_TXPOWER_MAX 0x7F

  Power control txPower field.
- #define LL_PWR_CTRL_TXPOWER_MIN 0x7E
- #define LL_PWR_CTRL_TXPOWER_UNAVAILABLE 127
- #define LL_PWR_CTRL_TXPOWER_UNMANAGED 126
- #define LL_ISOAL_SEG_HDR_MASK_SC 0x01
- #define LL_ISOAL_SEG_HDR_MASK_CMPLT 0x02

Enumerations

- enum {
  LL_PDU_ADV_IND = 0,
  LL_PDU_ADV_DIRECT_IND = 1,
  LL_PDU_ADV_NONCONN_IND = 2,
  LL_PDU_SCAN_REQ = 3,
  LL_PDU_SCAN_RSP = 4,
  LL_PDU_CONNECT_IND = 5,
  LL_PDU_ADV_SCAN_IND = 6,
  LL_PDU_AUX_SCAN_REQ = 3,
  LL_PDU_AUX_CONNECT_REQ = 5,
  LL_PDU_ADV_EXT_IND = 7,
  LL_PDU_AUX_ADV_IND = 7,
  LL_PDU_AUX_SCAN_RSP = 7,
  LL_PDU_AUX_SYNC_IND = 7,
  LL_PDU_AUX_CHAIN_IND = 7,
  LL_PDU_AUX_CONNECT_RSP = 8 }

  Advertising channel PDU types.
- enum {
  LL_EXT_HDR_ADV_ADDR_BIT = (1 << 0),
  LL_EXT_HDR_TGT_ADDR_BIT = (1 << 1),
  LL_EXT_HDR_CTE_INFO_BIT = (1 << 2),
  LL_EXT_HDR_ADI_BIT = (1 << 3),
  LL_EXT_HDR_AUX_PTR_BIT = (1 << 4),
  LL_EXT_HDR_SYNC_INFO_BIT = (1 << 5),
  LL_EXT_HDR_TX_PWR_BIT = (1 << 6) }

  Extended header bit definition.
- enum {
  LL_SYNC_TRSF_MODE_OFF = 0,
  LL_SYNC_TRSF_MODE_REP_DISABLED = 1,
  LL_SYNC_TRSF_MODE_REP_ENABLED = 2,
  LL_SYNC_TRSF_MAX_MODE = LL_SYNC_TRSF_MODE_REP_ENABLED }

  Periodic sync transfer receive mode.
- enum {
  LL_PDU_CONN_UPDATE_IND = 0x00,
  LL_PDU_CHANNEL_MAP_IND = 0x01,
  LL_PDU_TERMINATE_IND = 0x02,
  LL_PDU_ENC_REQ = 0x03,
  LL_PDU_ENC_RSP = 0x04,
  LL_PDU_START_ENC_REQ = 0x05,
  LL_PDU_START_ENC_RSP = 0x06,
  LL_PDU_UNKNOWN_RSP = 0x07,
  LL_PDU_FEATURE_REQ = 0x08,

LL_PDU_FEATURE_RSP = 0x09,
LL_PDU_PAUSE_ENC_REQ = 0x0A,
LL_PDU_PAUSE_ENC_RSP = 0x0B,
LL_PDU_VERSION_IND = 0x0C,
LL_PDU_REJECT_IND = 0x0D,
LL_PDU_SLV_FEATURE_REQ = 0x0E,
LL_PDU_CONN_PARAM_REQ = 0x0F,
LL_PDU_CONN_PARAM_RSP = 0x10,
LL_PDU_REJECT_EXT_IND = 0x11,
LL_PDU_PING_REQ = 0x12,
LL_PDU_PING_RSP = 0x13,
LL_PDU_LENGTH_REQ = 0x14,
LL_PDU_LENGTH_RSP = 0x15,
LL_PDU_PHY_REQ = 0x16,
LL_PDU_PHY_RSP = 0x17,
LL_PDU_PHY_UPDATE_IND = 0x18,
LL_PDU_MIN_USED_CHAN_IND = 0x19,
LL_PDU_PERIODIC_SYNC_IND = 0x1C,
LL_PDU_PEER_SCA_REQ = 0x1D,
LL_PDU_PEER_SCA_RSP = 0x1E,
LL_PDU_CIS_REQ = 0x1F,
LL_PDU_CIS_RSP = 0x20,
LL_PDU_CIS_IND = 0x21,
LL_PDU_CIS_TERM_IND = 0x22,
LL_PDU_PWR_CTRL_REQ = 0x23,
LL_PDU_PWR_CTRL_RSP = 0x24,
LL_PDU_PWR_CHANGE_IND = 0x25,
LL_PDU_UNSPECIFIED = 0xFF }

Data channel LL Control PDU types.

- enum {
LL_LLID_VS_PDU = 0x00,
LL_LLID_EMPTY_PDU = 0x01,
LL_LLID_CONT_PDU = 0x01,
LL_LLID_START_PDU = 0x02,
LL_LLID_CTRL_PDU = 0x03 }

  Data PDU LLID types.

- enum LlIsoLlid_t {
LL_LLID_ISO_UNF_END_PDU = 0x00,
LL_LLID_ISO_UNF_CONT_PDU = 0x01,
LL_LLID_ISO_FRA_PDU = 0x02,
LL_LLID_ISO_EMPTY_PDU = 0x01,
LL_LLID_BIG_CTRL_PDU = 0x03 }

  ISO PDU LLID types.

- enum LlFraming_t {
LL_ISO_PDU_TYPE_UNFRAMED = 0x00,
LL_ISO_PDU_TYPE_FRAMED = 0x01 }

  ISO PDU type.

- enum {
LL_ISO_TEST_PL_LEN_ZERO = 0x00,
LL_ISO_TEST_PL_LEN_VAR = 0x01,
LL_ISO_TEST_PL_LEN_MAX = 0x02 }

  ISO test payload types.

- enum {
LL_ISO_SDU_STATE_NEW = 0x00,
LL_ISO_SDU_STATE_CONT = 0x01 }

  ISOAL SDU assembly states.

- enum {
  LL_ACAD_OPCODE_CHAN_MAP_UPD = 0x28,
  LL_ACAD_OPCODE_BIG_INFO = 0x2C }

  ACAD opcodes.
- enum {
  LL_BIG_OPCODE_CHAN_MAP_IND = 0x00,
  LL_BIG_OPCODE_BIG_TERM_IND = 0x01,
  LL_BIG_OPCODE_MAX }

  BIG Control opcodes.
- enum {
  LL_MODIFY_SCA_MORE_ACCURATE = 0x00,
  LL_MODIFY_SCA_LESS_ACCURATE = 0x01,
  LL_MODIFY_SCA_NO_ACTION }

  Action parameter.
- enum {
  LL_PATH_LOSS_ZONE_LOW = 0x00,
  LL_PATH_LOSS_ZONE_MID,
  LL_PATH_LOSS_ZONE_HIGH }

  Path loss monitoring zones.
- enum {
  LL_POWER_REPORT_REASON_LOCAL,
  LL_POWER_REPORT_REASON_REMOTE,
  LL_POWER_REPORT_REASON_READ_REMOTE }
- enum {
  LL_CODEC_TRANS_CIS_BIT = (1 << 2),
  LL_CODEC_TRANS_BIS_BIT = (1 << 3) }

  Codec transport types.

4.2.1 Detailed Description

Link layer constant definitions.

4.2.2 Macro Definition Documentation

4.2.2.1 LL_VER_BT_CORE_SPEC_4_0

```
#define LL_VER_BT_CORE_SPEC_4_0 0x06
```

Bluetooth core specification 4.0

Definition at line 40 of file ll_defs.h.

4.2.2.2 LL_VER_BT_CORE_SPEC_4_1

```
#define LL_VER_BT_CORE_SPEC_4_1 0x07
```

Bluetooth core specification 4.1

Definition at line 41 of file ll_defs.h.

4.2.2.3 LL_VER_BT_CORE_SPEC_4_2

```
#define LL_VER_BT_CORE_SPEC_4_2 0x08
```

Bluetooth core specification 4.2

Definition at line 42 of file ll_defs.h.

4.2.2.4 LL_VER_BT_CORE_SPEC_5_0

```
#define LL_VER_BT_CORE_SPEC_5_0 0x09
```

Bluetooth core specification 5.0

Definition at line 43 of file ll_defs.h.

4.2.2.5 LL_VER_BT_CORE_SPEC_5_1

```
#define LL_VER_BT_CORE_SPEC_5_1 0x0A
```

Bluetooth core specification 5.1

Definition at line 44 of file ll_defs.h.

4.2.2.6 LL_VER_BT_CORE_SPEC_5_2

```
#define LL_VER_BT_CORE_SPEC_5_2 0x0B
```

Bluetooth core specification 5.2

Definition at line 45 of file ll_defs.h.

4.2.2.7 LL_VER_BT_CORE_SPEC_SYDNEY

```
#define LL_VER_BT_CORE_SPEC_SYDNEY 0x0C
```

Bluetooth core specification Sydney

Definition at line 46 of file ll_defs.h.

4.2.2.8 LL_RSSI_MIN

```
#define LL_RSSI_MIN -127
```

Minimum RSSI value.

Definition at line 50 of file ll_defs.h.

4.2.2.9 LL_RSSI_MAX

```
#define LL_RSSI_MAX 20
```

Maximum RSSI value.

Definition at line 51 of file ll_defs.h.

4.2.2.10 LL_RSSI_NOT_AVAIL

```
#define LL_RSSI_NOT_AVAIL 127
```

RSSI is not available.

Definition at line 52 of file ll_defs.h.

4.2.2.11 LL_CRC_LEN

```
#define LL_CRC_LEN 3
```

CRC length.

Definition at line 54 of file ll_defs.h.

4.2.2.12 LL_AA_LEN

```
#define LL_AA_LEN 4
```

Access address length.

Definition at line 55 of file ll_defs.h.

4.2.2.13 LL_PREAMBLE_LEN_1M

```
#define LL_PREAMBLE_LEN_1M 1
```

Preamble length (LE 1M PHY).

Definition at line 56 of file ll_defs.h.

4.2.2.14 LL_PREAMBLE_LEN_2M

```
#define LL_PREAMBLE_LEN_2M 2
```

Preamble length (LE 2M PHY).

Definition at line 57 of file ll_defs.h.

4.2.2.15 LL_PREAMBLE_LEN_CODED_BITS

```
#define LL_PREAMBLE_LEN_CODED_BITS 10
```

Preamble length (LE Coded PHY).

Definition at line 58 of file ll_defs.h.

4.2.2.16 LL_CI_LEN_BITS

```
#define LL_CI_LEN_BITS 2
```

Coding indicator length (LE Coded PHY).

Definition at line 59 of file ll_defs.h.

4.2.2.17 LL_TERM1_LEN_BITS

```
#define LL_TERM1_LEN_BITS 3
```

TERM1 length (LE Coded PHY).

Definition at line 60 of file ll_defs.h.

4.2.2.18 LL_TERM2_LEN_BITS

```
#define LL_TERM2_LEN_BITS 3
```

TERM2 length (LE Coded PHY).

Definition at line 61 of file ll_defs.h.

4.2.2.19 LL_RAND_ADDR_TYPE_MASK

```
#define LL_RAND_ADDR_TYPE_MASK UINT64_C(0xC00000000000)
```

BD Random Address type mask.

Definition at line 63 of file ll_defs.h.

4.2.2.20 LL_RAND_ADDR_TYPE_STATIC

```
#define LL_RAND_ADDR_TYPE_STATIC UINT64_C(0xC00000000000)
```

Static Random Address type.

Definition at line 64 of file ll_defs.h.

4.2.2.21 LL_RAND_ADDR_TYPE_RPA

```
#define LL_RAND_ADDR_TYPE_RPA UINT64_C(0x400000000000)
```

Resolvable Private Address type.

Definition at line 65 of file ll_defs.h.

4.2.2.22 LL_RAND_ADDR_TYPE_NRPA

```
#define LL_RAND_ADDR_TYPE_NRPA UINT64_C(0x000000000000)
```

Non-Resolvable Private Address type.

Definition at line 66 of file ll_defs.h.

4.2.2.23 LL_SCAN_REQ_PDU_LEN

```
#define LL_SCAN_REQ_PDU_LEN 12
```

Size of a scan request PDU.

Definition at line 92 of file ll_defs.h.

4.2.2.24 LL_CONN_IND_PDU_LEN

```
#define LL_CONN_IND_PDU_LEN 34
```

Size of a connect indication PDU.

Definition at line 93 of file ll_defs.h.

4.2.2.25 LL_CONN_RSP_PDU_LEN

```
#define LL_CONN_RSP_PDU_LEN 14
```

Size of an auxiliary connect response PDU.

Definition at line 94 of file ll_defs.h.

4.2.2.26 LL_CHAN_ADV_MIN_IDX

```
#define LL_CHAN_ADV_MIN_IDX 37
```

Minimum advertising channel index.

Definition at line 96 of file ll_defs.h.

4.2.2.27 LL_CHAN_ADV_MAX_IDX

```
#define LL_CHAN_ADV_MAX_IDX 39
```

Maximum advertising channel index.

Definition at line 97 of file ll_defs.h.

4.2.2.28 LL_NUM_CHAN_ADV

```
#define LL_NUM_CHAN_ADV 3
```

Total number of advertising channels.

Definition at line 98 of file ll_defs.h.

4.2.2.29 LL_ADVBU_MAX_LEN

```
#define LL_ADVBU_MAX_LEN 31
```

Maximum advertising channel host data length.

Definition at line 100 of file ll_defs.h.

4.2.2.30 LL_ADVB_MAX_LEN

```
#define LL_ADVB_MAX_LEN 39
```

Maximum advertising channel PDU length.

Definition at line 101 of file ll_defs.h.

4.2.2.31 LL_ADVB_MIN_LEN

```
#define LL_ADVB_MIN_LEN (LL_ADVB_MAX_LEN - LL_ADVBU_MAX_LEN)
```

Minimum advertising channel packet length.

Definition at line 102 of file ll_defs.h.

4.2.2.32 LL_ADVB_MAX_TIME_1M

```
#define LL_ADVB_MAX_TIME_1M ((LL_BLE_US_PER_BYTE_1M * (LL_ADVB_MAX_LEN - LL_ADV_HDR_LEN)) + L↩
L_MIN_PKT_TIME_US_1M)
```

Maximum time for a 1M advertising channel PDU.

Definition at line 103 of file ll_defs.h.

4.2.2.33 LL_ADVB_MAX_TIME_2M

```
#define LL_ADVB_MAX_TIME_2M ((LL_BLE_US_PER_BYTE_2M * (LL_ADVB_MAX_LEN - LL_ADV_HDR_LEN)) + L↩
L_MIN_PKT_TIME_US_2M)
```

Maximum time for a 2M advertising channel PDU.

Definition at line 105 of file ll_defs.h.

4.2.2.34 LL_ADVB_MAX_TIME_S2

```
#define LL_ADVB_MAX_TIME_S2 ((LL_BLE_US_PER_BYTE_CODED_S2 * (LL_ADVB_MAX_LEN - LL_ADV_HDR_L↩
EN)) + LL_MIN_PKT_TIME_US_CODED_S2)
```

Maximum time for a Coded S2 advertising channel PDU.

Definition at line 107 of file ll_defs.h.

4.2.2.35 LL_ADVB_MAX_TIME_S8

```
#define LL_ADVB_MAX_TIME_S8 ((LL_BLE_US_PER_BYTE_CODED_S8 * (LL_ADVB_MAX_LEN - LL_ADV_HDR_L↩
EN)) + LL_MIN_PKT_TIME_US_CODED_S8)
```

Maximum time for a Coded S8 advertising channel PDU.

Definition at line 109 of file ll_defs.h.

4.2.2.36 LL_ADV_PKT_MAX_USEC

```
#define LL_ADV_PKT_MAX_USEC LL_ADVB_MAX_TIME_1M
```

Maximum time in microseconds for an advertising packet.

Definition at line 112 of file ll_defs.h.

4.2.2.37 LL_SCAN_REQ_MAX_USEC

```
#define LL_SCAN_REQ_MAX_USEC ((8 * (LL_ADV_PREFIX_LEN + LL_SCAN_PREFIX_LEN)) + LL_MIN_PKT_TIM←
E_US_1M)
```

Maximum time in microseconds for a scan request packet.

Definition at line 113 of file ll_defs.h.

4.2.2.38 LL_SCAN_RSP_MAX_USEC

```
#define LL_SCAN_RSP_MAX_USEC LL_ADVB_MAX_TIME_1M
```

Maximum time in microseconds for a scan response packet.

Definition at line 115 of file ll_defs.h.

4.2.2.39 LL_ADV_HDR_LEN

```
#define LL_ADV_HDR_LEN 2
```

Advertising channel header length.

Definition at line 117 of file ll_defs.h.

4.2.2.40 LL_ADV_HDR_TYPE_OFFS

```
#define LL_ADV_HDR_TYPE_OFFS 0
```

Advertising header type offset.

Definition at line 118 of file ll_defs.h.

4.2.2.41 LL_ADV_HDR_TYPE_MSK

```
#define LL_ADV_HDR_TYPE_MSK 0x0F
```

Advertising header type mask.

Definition at line 119 of file ll_defs.h.

4.2.2.42 LL_ADV_HDR_LEN_OFFS

```
#define LL_ADV_HDR_LEN_OFFS 1
```

Advertising header length offset.

Definition at line 120 of file ll_defs.h.

4.2.2.43 LL_ADV_HDR_LEN_MSK

```
#define LL_ADV_HDR_LEN_MSK 0x3F
```

Advertising header length mask for 4.2.

Definition at line 121 of file ll_defs.h.

4.2.2.44 LL_ADV_HDR_CP_MSK

```
#define LL_ADV_HDR_CP_MSK 0x40
```

Advertising header CP Mask for 5.1.

Definition at line 122 of file ll_defs.h.

4.2.2.45 LL_ADV_EXT_HDR_LEN_MSK

```
#define LL_ADV_EXT_HDR_LEN_MSK 0xFF
```

Advertising extension header length mask for 5.0.

Definition at line 123 of file ll_defs.h.

4.2.2.46 LL_ADV_PREFIX_LEN

```
#define LL_ADV_PREFIX_LEN 6
```

Advertising PDU payload prefix length (AdvA).

Definition at line 124 of file ll_defs.h.

4.2.2.47 LL_SCAN_PREFIX_LEN

```
#define LL_SCAN_PREFIX_LEN 6
```

Scan request/response PDU payload prefix length (AdvA).

Definition at line 125 of file ll_defs.h.

4.2.2.48 LL_ADV_ACCESS_ADDR

```
#define LL_ADV_ACCESS_ADDR UINT32_C(0x8E89BED6)
```

Advertising channel access address.

Definition at line 127 of file ll_defs.h.

4.2.2.49 LL_ADV_CRC_INIT

```
#define LL_ADV_CRC_INIT UINT32_C(0x555555)
```

Advertising CRC initial value.

Definition at line 128 of file ll_defs.h.

4.2.2.50 LL_DIR_ADV_INTER_TICKS

```
#define LL_DIR_ADV_INTER_TICKS 6
```

Advertising interval between directed advertising events (3.75 ms).

Definition at line 130 of file ll_defs.h.

4.2.2.51 LL_DIR_ADV_DUR_TICKS

```
#define LL_DIR_ADV_DUR_TICKS 2048
```

Maximum high duty cycle directed advertising duration (1.28 seconds).

Definition at line 131 of file ll_defs.h.

4.2.2.52 LL_MAX_ADV_HANDLE

```
#define LL_MAX_ADV_HANDLE 0xEF
```

Maximum advertising handle.

Definition at line 145 of file ll_defs.h.

4.2.2.53 LL_MAX_ADV_SID

```
#define LL_MAX_ADV_SID 0x0F
```

Maximum advertising SID

Definition at line 146 of file ll_defs.h.

4.2.2.54 LL_EXT_ADV_HDR_MIN_LEN

```
#define LL_EXT_ADV_HDR_MIN_LEN 1
```

Minimum extended advertising header length (ExtHdrLen and AdvMode fields).

Definition at line 148 of file ll_defs.h.

4.2.2.55 LL_EXT_ADV_HDR_MAX_LEN

```
#define LL_EXT_ADV_HDR_MAX_LEN 64
```

Maximum extended advertising header length (ExtHdrLen, AdvMode fields and Extended header).

Definition at line 149 of file ll_defs.h.

4.2.2.56 LL_EXT_HDR_FLAG_LEN

```
#define LL_EXT_HDR_FLAG_LEN 1
```

Length of extended header flag field

Definition at line 150 of file ll_defs.h.

4.2.2.57 LL_EXT_ADVBU_MAX_LEN

```
#define LL_EXT_ADVBU_MAX_LEN 251
```

Maximum extended advertising channel PDU host data length.

Definition at line 151 of file ll_defs.h.

4.2.2.58 LL_EXT_ADVB_MAX_LEN

```
#define LL_EXT_ADVB_MAX_LEN 257
```

Maximum extended advertising channel PDU length.

Definition at line 152 of file ll_defs.h.

4.2.2.59 LL_EXT_ADVB_NORMAL_LEN

```
#define LL_EXT_ADVB_NORMAL_LEN 50
```

Normal extended advertising channel PDU length.

Definition at line 153 of file ll_defs.h.

4.2.2.60 LL_EXT_HDR_ACAD_MAX_LEN

```
#define LL_EXT_HDR_ACAD_MAX_LEN LL_EXT_ADV_HDR_MAX_LEN - LL_EXT_ADV_HDR_MIN_LEN - LL_EXT_HDR↩
_FLAG_LEN
```

Maximum possible ACAD length (Max extended header minus ExtHdrLen, AdvMode, and extended header flag field.

Definition at line 155 of file ll_defs.h.

4.2.2.61 LL_EXT_ADVB_MAX_TIME_1M

```
#define LL_EXT_ADVB_MAX_TIME_1M ((LL_BLE_US_PER_BYTE_1M * (LL_EXT_ADVB_MAX_LEN - LL_ADV_HDR_L←
EN)) + LL_MIN_PKT_TIME_US_1M)
```

Maximum time for a 1M advertising channel PDU.

Definition at line 157 of file ll_defs.h.

4.2.2.62 LL_EXT_ADVB_MAX_TIME_2M

```
#define LL_EXT_ADVB_MAX_TIME_2M ((LL_BLE_US_PER_BYTE_2M * (LL_EXT_ADVB_MAX_LEN - LL_ADV_HDR_L←
EN)) + LL_MIN_PKT_TIME_US_2M)
```

Maximum time for a 2M advertising channel PDU.

Definition at line 159 of file ll_defs.h.

4.2.2.63 LL_EXT_ADVB_MAX_TIME_S2

```
#define LL_EXT_ADVB_MAX_TIME_S2 ((LL_BLE_US_PER_BYTE_CODED_S2 * (LL_EXT_ADVB_MAX_LEN - LL_ADV←
_HDR_LEN)) + LL_MIN_PKT_TIME_US_CODED_S2)
```

Maximum time for a Coded S2 advertising channel PDU.

Definition at line 161 of file ll_defs.h.

4.2.2.64 LL_EXT_ADVB_MAX_TIME_S8

```
#define LL_EXT_ADVB_MAX_TIME_S8 ((LL_BLE_US_PER_BYTE_CODED_S8 * (LL_EXT_ADVB_MAX_LEN - LL_ADV←
_HDR_LEN)) + LL_MIN_PKT_TIME_US_CODED_S8)
```

Maximum time for a Coded S8 advertising channel PDU.

Definition at line 163 of file ll_defs.h.

4.2.2.65 LL_EXT_ADVB_NORMAL_TIME_S8

```
#define LL_EXT_ADVB_NORMAL_TIME_S8 ((LL_BLE_US_PER_BYTE_CODED_S8 * (LL_EXT_ADVB_NORMAL_LEN -
LL_ADV_HDR_LEN)) + LL_MIN_PKT_TIME_US_CODED_S8)
```

Time for a Coded S8 advertising channel PDU with normal length.

Definition at line 165 of file ll_defs.h.

4.2.2.66 LL_AUX_PTR_MAX_USEC

```
#define LL_AUX_PTR_MAX_USEC 2457300
```

Maximum AuxPtr offset value in microseconds.

Definition at line 168 of file ll_defs.h.

4.2.2.67 LL_SYNC_MIN_TIMEOUT

```
#define LL_SYNC_MIN_TIMEOUT 0x000A
```

Minimum synchronization timeout.

Definition at line 170 of file ll_defs.h.

4.2.2.68 LL_SYNC_MAX_TIMEOUT

```
#define LL_SYNC_MAX_TIMEOUT 0x4000
```

Maximum synchronization timeout.

Definition at line 171 of file ll_defs.h.

4.2.2.69 LL_SYNC_MAX_SKIP

```
#define LL_SYNC_MAX_SKIP 0x01F3
```

Maximum synchronization skip.

Definition at line 172 of file ll_defs.h.

4.2.2.70 LL_SYNC_MAX_HANDLE

```
#define LL_SYNC_MAX_HANDLE 0x0EFF
```

Maximum synchronization handle.

Definition at line 173 of file ll_defs.h.

4.2.2.71 LL_PER_ADV_INT_MIN

```
#define LL_PER_ADV_INT_MIN 0x0006
```

Minimum periodic advertising interval.

Definition at line 175 of file ll_defs.h.

4.2.2.72 LL_SYNC_OFFS_ADJUST_USEC

```
#define LL_SYNC_OFFS_ADJUST_USEC LL_AUX_PTR_MAX_USEC
```

Sync offset adjust of 2.4573 seconds.

Definition at line 177 of file ll_defs.h.

4.2.2.73 LL_SYNC_INFO_LEN

```
#define LL_SYNC_INFO_LEN 18
```

Size of SyncInfo field.

Definition at line 178 of file ll_defs.h.

4.2.2.74 LL_CONN_UPD_IND_PDU_LEN

```
#define LL_CONN_UPD_IND_PDU_LEN 12
```

Connection update indication PDU length.

Definition at line 243 of file ll_defs.h.

4.2.2.75 LL_CHAN_MAP_IND_PDU_LEN

```
#define LL_CHAN_MAP_IND_PDU_LEN 8
```

Channel map indication PDU length.

Definition at line 244 of file ll_defs.h.

4.2.2.76 LL_TERMINATE_IND_PDU_LEN

```
#define LL_TERMINATE_IND_PDU_LEN 2
```

Terminate indication PDU length.

Definition at line 245 of file ll_defs.h.

4.2.2.77 LL_ENC_REQ_LEN

```
#define LL_ENC_REQ_LEN 23
```

Encryption request PDU length.

Definition at line 246 of file ll_defs.h.

4.2.2.78 LL_ENC_RSP_LEN

```
#define LL_ENC_RSP_LEN 13
```

Encryption response PDU length.

Definition at line 247 of file ll_defs.h.

4.2.2.79 LL_START_ENC_LEN

```
#define LL_START_ENC_LEN 1
```

Start encryption request/response PDU length.

Definition at line 248 of file ll_defs.h.

4.2.2.80 LL_UNKNOWN_RSP_LEN

```
#define LL_UNKNOWN_RSP_LEN 2
```

Unknown response PDU length.

Definition at line 249 of file ll_defs.h.

4.2.2.81 LL_FEATURE_PDU_LEN

```
#define LL_FEATURE_PDU_LEN 9
```

Feature request/response PDU length.

Definition at line 250 of file ll_defs.h.

4.2.2.82 LL_PAUSE_ENC_LEN

```
#define LL_PAUSE_ENC_LEN 1
```

Pause encryption request/response PDU length.

Definition at line 251 of file ll_defs.h.

4.2.2.83 LL_VERSION_IND_PDU_LEN

```
#define LL_VERSION_IND_PDU_LEN 6
```

Version indication PDU length.

Definition at line 252 of file ll_defs.h.

4.2.2.84 LL_REJECT_IND_PDU_LEN

```
#define LL_REJECT_IND_PDU_LEN 2
```

Reject indication PDU length.

Definition at line 253 of file ll_defs.h.

4.2.2.85 LL_CONN_PARAM_PDU_LEN

```
#define LL_CONN_PARAM_PDU_LEN 24
```

Connection parameter request or response PDU length.

Definition at line 255 of file ll_defs.h.

4.2.2.86 LL_REJECT_EXT_IND_PDU_LEN

```
#define LL_REJECT_EXT_IND_PDU_LEN 3
```

Reject extended indication PDU length.

Definition at line 256 of file ll_defs.h.

4.2.2.87 LL_PING_PDU_LEN

```
#define LL_PING_PDU_LEN 1
```

Ping request/response PDU length.

Definition at line 257 of file ll_defs.h.

4.2.2.88 LL_DATA_LEN_PDU_LEN

```
#define LL_DATA_LEN_PDU_LEN 9
```

Data length request or response PDU length.

Definition at line 258 of file ll_defs.h.

4.2.2.89 LL_PHY_PDU_LEN

```
#define LL_PHY_PDU_LEN 3
```

PHY request/response PDU length.

Definition at line 260 of file ll_defs.h.

4.2.2.90 LL_PHY_UPD_IND_PDU_LEN

```
#define LL_PHY_UPD_IND_PDU_LEN 5
```

PHY update indication PDU length.

Definition at line 261 of file ll_defs.h.

4.2.2.91 LL_MIN_USED_CHAN_PDU_LEN

```
#define LL_MIN_USED_CHAN_PDU_LEN 3
```

Minimum used channels indication PDU length.

Definition at line 262 of file ll_defs.h.

4.2.2.92 LL_PERIODIC_SYNC_PDU_LEN

```
#define LL_PERIODIC_SYNC_PDU_LEN 35
```

Periodic sync indication PDU length.

Definition at line 264 of file ll_defs.h.

4.2.2.93 LL_PEER_SCA_REQ_LEN

```
#define LL_PEER_SCA_REQ_LEN 2
```

Peer SCA request PDU length.

Definition at line 266 of file ll_defs.h.

4.2.2.94 LL_PEER_SCA_RSP_LEN

```
#define LL_PEER_SCA_RSP_LEN 2
```

Peer SCA response PDU length.

Definition at line 267 of file ll_defs.h.

4.2.2.95 LL_CIS_REQ_LEN

```
#define LL_CIS_REQ_LEN 36
```

CIS request PDU length.

Definition at line 268 of file ll_defs.h.

4.2.2.96 LL_CIS_RSP_LEN

```
#define LL_CIS_RSP_LEN 9
```

CIS response PDU length.

Definition at line 269 of file ll_defs.h.

4.2.2.97 LL_CIS_IND_LEN

```
#define LL_CIS_IND_LEN 16
```

CIS indication PDU length.

Definition at line 270 of file ll_defs.h.

4.2.2.98 LL_CIS_TERM_LEN

```
#define LL_CIS_TERM_LEN 4
```

CIS termination PDU length.

Definition at line 271 of file ll_defs.h.

4.2.2.99 LL_CIS_SDU_CONFIG_REQ_LEN

```
#define LL_CIS_SDU_CONFIG_REQ_LEN 13
```

CIS SDU config request PDU length.

Definition at line 272 of file ll_defs.h.

4.2.2.100 LL_CIS_SDU_CONFIG_RSP_LEN

```
#define LL_CIS_SDU_CONFIG_RSP_LEN 4
```

CIS SDU config response PDU length.

Definition at line 273 of file ll_defs.h.

4.2.2.101 LL_PWR_CTRL_REQ_LEN

```
#define LL_PWR_CTRL_REQ_LEN 4
```

Power Control request PDU length.

Definition at line 274 of file ll_defs.h.

4.2.2.102 LL_PWR_CTRL_RSP_LEN

```
#define LL_PWR_CTRL_RSP_LEN 5
```

Power Control response PDU length.

Definition at line 275 of file ll_defs.h.

4.2.2.103 LL_PWR_CHANGE_IND_LEN

```
#define LL_PWR_CHANGE_IND_LEN 5
```

Power Indication PDU length.

Definition at line 276 of file ll_defs.h.

4.2.2.104 LL_EMPTY_PDU_LEN

```
#define LL_EMPTY_PDU_LEN 2
```

Length of an empty data PDU.

Definition at line 278 of file ll_defs.h.

4.2.2.105 LL_DATA_HDR_LEN

```
#define LL_DATA_HDR_LEN 2
```

Data channel header length.

Definition at line 280 of file ll_defs.h.

4.2.2.106 LL_DATA_HDR_MAX_LEN

```
#define LL_DATA_HDR_MAX_LEN 4
```

Data channel header max length (with CTE field).

Definition at line 281 of file ll_defs.h.

4.2.2.107 LL_DATA_MIC_LEN

```
#define LL_DATA_MIC_LEN 4
```

Data channel PDU MIC length.

Definition at line 282 of file ll_defs.h.

4.2.2.108 LL_DATA_HDR_LLID_MSK

```
#define LL_DATA_HDR_LLID_MSK 0x03
```

Data PDU LLID mask.

Definition at line 284 of file ll_defs.h.

4.2.2.109 LL_DATA_HDR_LEN_MSK

```
#define LL_DATA_HDR_LEN_MSK 0xFF
```

Data header length mask. BLE 4.2 data len extension allows 8 bits.

Definition at line 285 of file ll_defs.h.

4.2.2.110 LL_MAX_NUM_CHAN_DATA

```
#define LL_MAX_NUM_CHAN_DATA 37
```

Maximum number of used data channels.

Definition at line 287 of file ll_defs.h.

4.2.2.111 LL_MIN_NUM_CHAN_DATA

```
#define LL_MIN_NUM_CHAN_DATA 2
```

Minimum number of used data channels.

Definition at line 288 of file ll_defs.h.

4.2.2.112 LL_ECC_KEY_LEN

```
#define LL_ECC_KEY_LEN 32
```

ECC key length.

Definition at line 303 of file ll_defs.h.

4.2.2.113 LL_DEF_RES_ADDR_TO_SEC

```
#define LL_DEF_RES_ADDR_TO_SEC 900
```

Default resolvable address timeout in seconds.

Definition at line 305 of file ll_defs.h.

4.2.2.114 LL_RAND_LEN

```
#define LL_RAND_LEN 8
```

Length of random number

Definition at line 307 of file ll_defs.h.

4.2.2.115 LL_KEY_LEN

```
#define LL_KEY_LEN 16
```

Encryption key length.

Definition at line 308 of file ll_defs.h.

4.2.2.116 LL_SKD_LEN

```
#define LL_SKD_LEN LL_KEY_LEN
```

Session key diversifier length.

Definition at line 309 of file ll_defs.h.

4.2.2.117 LL_IV_LEN

```
#define LL_IV_LEN 8
```

Initialization vector length.

Definition at line 310 of file ll_defs.h.

4.2.2.118 LL_BC_LEN

```
#define LL_BC_LEN LL_KEY_LEN
```

Broadcast code length.

Definition at line 311 of file ll_defs.h.

4.2.2.119 LL_GIV_LEN

```
#define LL_GIV_LEN 8
```

GIV length.

Definition at line 312 of file ll_defs.h.

4.2.2.120 LL_GSKD_LEN

```
#define LL_GSKD_LEN 16
```

GSKD code length.

Definition at line 313 of file ll_defs.h.

4.2.2.121 LL_DEF_AUTH_TO_MS

```
#define LL_DEF_AUTH_TO_MS 30000
```

Default authentication timeout in milliseconds.

Definition at line 315 of file ll_defs.h.

4.2.2.122 LL_DATA_LEN_TO_TIME_1M

```
#define LL_DATA_LEN_TO_TIME_1M(
              len,
              enc ) ((LL_BLE_US_PER_BYTE_1M * ((len) + ((enc) ? LL_DATA_MIC_LEN : 0))) + LL_↩
MIN_PKT_TIME_US_1M)
```

Convert data length to time.

Definition at line 319 of file ll_defs.h.

4.2.2.123 LL_DATA_LEN_TO_TIME_2M

```
#define LL_DATA_LEN_TO_TIME_2M(
              len,
              enc ) ((LL_BLE_US_PER_BYTE_2M * ((len) + ((enc) ? LL_DATA_MIC_LEN : 0))) + LL_↩
MIN_PKT_TIME_US_2M)
```

Convert data length to time.

Definition at line 322 of file ll_defs.h.

4.2.2.124 LL_DATA_LEN_TO_TIME_CODED_S8

```
#define LL_DATA_LEN_TO_TIME_CODED_S8(
              len,
              enc ) ((LL_BLE_US_PER_BYTE_CODED_S8 * ((len) + ((enc) ? LL_DATA_MIC_LEN : 0))) +
LL_MIN_PKT_TIME_US_CODED_S8)
```

Convert data length to time.

Definition at line 325 of file ll_defs.h.

4.2.2.125 LL_DATA_LEN_TO_TIME_CODED_S2

```
#define LL_DATA_LEN_TO_TIME_CODED_S2(
              len,
              enc ) ((LL_BLE_US_PER_BYTE_CODED_S2 * ((len) + ((enc) ? LL_DATA_MIC_LEN : 0))) +
LL_MIN_PKT_TIME_US_CODED_S2)
```

Convert data length to time.

Definition at line 328 of file ll_defs.h.

4.2.2.126 LL_MIN_INSTANT

```
#define LL_MIN_INSTANT 6
```

Minimum number of CE to apply a CONN_UPD or CHAN_MAP.

Definition at line 332 of file ll_defs.h.

4.2.2.127 LL_MAX_ADV_DATA_LEN

```
#define LL_MAX_ADV_DATA_LEN 1650
```

Maximum advertising data length.

Definition at line 334 of file ll_defs.h.

4.2.2.128 LL_MAX_DATA_LEN_MIN

```
#define LL_MAX_DATA_LEN_MIN 27
```

Minimum value for maximum Data PDU length

Definition at line 336 of file ll_defs.h.

4.2.2.129 LL_MAX_DATA_LEN_ABS_MAX

```
#define LL_MAX_DATA_LEN_ABS_MAX 251
```

Absolute maximum limit for maximum Data PDU length

Definition at line 337 of file ll_defs.h.

4.2.2.130 LL_MAX_DATA_TIME_MIN

```
#define LL_MAX_DATA_TIME_MIN 328
```

Minimum value for maximum Data PDU time

Definition at line 339 of file ll_defs.h.

4.2.2.131 LL_MAX_DATA_TIME_ABS_MAX

```
#define LL_MAX_DATA_TIME_ABS_MAX 17040
```

Absolute maximum limit for maximum Data PDU time

Definition at line 340 of file ll_defs.h.

4.2.2.132 LL_MAX_DATA_TIME_ABS_MAX_1M

```
#define LL_MAX_DATA_TIME_ABS_MAX_1M 2128
```

Absolute maximum limit for maximum Data PDU time (LE 1M PHY)

Definition at line 341 of file ll_defs.h.

4.2.2.133 LL_MAX_DATA_TIME_ABS_MIN_CODED

```
#define LL_MAX_DATA_TIME_ABS_MIN_CODED 2704
```

Absolute minimum limit for maximum Data PDU time (CODED PHY)

Definition at line 342 of file ll_defs.h.

4.2.2.134 LL_T_PRT_SEC

```
#define LL_T_PRT_SEC 40
```

LLCP procedure response timeout in seconds.

Definition at line 344 of file ll_defs.h.

4.2.2.135 LL_MAX_ADV_DLY_MS

```
#define LL_MAX_ADV_DLY_MS 10
```

Maximum advertising delay in milliseconds.

Definition at line 346 of file ll_defs.h.

4.2.2.136 LL_MIN_CONN_INTERVAL

```
#define LL_MIN_CONN_INTERVAL 6
```

Minimum value for connection interval.

Definition at line 348 of file ll_defs.h.

4.2.2.137 LL_MAX_CONN_INTERVAL

```
#define LL_MAX_CONN_INTERVAL 3200
```

Maximum value for connection interval.

Definition at line 349 of file ll_defs.h.

4.2.2.138 LL_MIN_TX_WIN_SIZE

```
#define LL_MIN_TX_WIN_SIZE 1
```

Minimum value for transmit window size.

Definition at line 351 of file ll_defs.h.

4.2.2.139 LL_MAX_TX_WIN_SIZE

```
#define LL_MAX_TX_WIN_SIZE 8
```

Maximum value for transmit window size.

Definition at line 352 of file ll_defs.h.

4.2.2.140 LL_MAX_CONN_LATENCY

```
#define LL_MAX_CONN_LATENCY 499
```

Maximum value for connection slave latency.

Definition at line 354 of file ll_defs.h.

4.2.2.141 LL_MIN_SUP_TIMEOUT

```
#define LL_MIN_SUP_TIMEOUT 10
```

Minimum value for connection supervision timeout.

Definition at line 356 of file ll_defs.h.

4.2.2.142 LL_MAX_SUP_TIMEOUT

```
#define LL_MAX_SUP_TIMEOUT 3200
```

Maximum value for connection supervision timeout.

Definition at line 357 of file ll_defs.h.

4.2.2.143 LL_MIN_POWER_THRESHOLD

```
#define LL_MIN_POWER_THRESHOLD -128
```

Minimum value for power threshold.

Definition at line 359 of file ll_defs.h.

4.2.2.144 LL_MAX_POWER_THRESHOLD

```
#define LL_MAX_POWER_THRESHOLD 127
```

Maximum value for power threshold.

Definition at line 360 of file ll_defs.h.

4.2.2.145 LL_MAX_PHYS

```
#define LL_MAX_PHYS 3
```

Number of LE PHYs.

Definition at line 362 of file ll_defs.h.

4.2.2.146 LL_ALL_PHYS_MSK

```
#define LL_ALL_PHYS_MSK 0x7
```

All supported LE PHYs mask.

Definition at line 363 of file ll_defs.h.

4.2.2.147 LL_ISO_DATA_HDR_LEN

```
#define LL_ISO_DATA_HDR_LEN 2
```

ISO Data PDU header length.

Definition at line 367 of file ll_defs.h.

4.2.2.148 LL_ISO_DATA_PLD_MAX_LEN

```
#define LL_ISO_DATA_PLD_MAX_LEN 251
```

Maximum ISO Data PDU payload length

Definition at line 368 of file ll_defs.h.

4.2.2.149 LL_ISO_PDU_MAX_LEN

```
#define LL_ISO_PDU_MAX_LEN (HCI_ISO_HDR_LEN + BB_DATA_PLD_MAX_LEN + BB_DATA_PDU_TAILROOM)
```

Maximum ISO Data PDU length.

Definition at line 369 of file ll_defs.h.

4.2.2.150 LL_ISO_SEG_HDR_LEN

```
#define LL_ISO_SEG_HDR_LEN 2
```

Segmentation header length.

Definition at line 372 of file ll_defs.h.

4.2.2.151 LL_ISO_SEG_TO_LEN

```
#define LL_ISO_SEG_TO_LEN 3
```

Segmentation Time Offset length.

Definition at line 373 of file ll_defs.h.

4.2.2.152 LL_MAX_CIS_COUNT

```
#define LL_MAX_CIS_COUNT 0x10
```

Maximum count for CIS.

Definition at line 375 of file ll_defs.h.

4.2.2.153 LL_MIN_CIG_ID

```
#define LL_MIN_CIG_ID 0x00
```

Minimum value for CIG ID.

Definition at line 377 of file ll_defs.h.

4.2.2.154 LL_MAX_CIG_ID

```
#define LL_MAX_CIG_ID 0xEF
```

Maximum value for CIG ID.

Definition at line 378 of file ll_defs.h.

4.2.2.155 LL_MIN_CIS_ID

```
#define LL_MIN_CIS_ID 0x00
```

Minimum value for CIS ID.

Definition at line 380 of file ll_defs.h.

4.2.2.156 LL_MAX_CIS_ID

```
#define LL_MAX_CIS_ID 0xEF
```

Maximum value for CIS ID.

Definition at line 381 of file ll_defs.h.

4.2.2.157 LL_MIN_ISO_INTERV

```
#define LL_MIN_ISO_INTERV 0x0004
```

Minimum value for ISO interval.

Definition at line 383 of file ll_defs.h.

4.2.2.158 LL_MAX_ISO_INTERV

```
#define LL_MAX_ISO_INTERV 0x0C80
```

Maximum value for ISO interval.

Definition at line 384 of file ll_defs.h.

4.2.2.159 LL_MIN_ISOAL_PDU_TYPE

```
#define LL_MIN_ISOAL_PDU_TYPE 0x00
```

Minimum value for ISOAL PDU type.

Definition at line 386 of file ll_defs.h.

4.2.2.160 LL_MAX_ISOAL_PDU_TYPE

```
#define LL_MAX_ISOAL_PDU_TYPE 0x01
```

Maximum value for ISOAL PDU type.

Definition at line 387 of file ll_defs.h.

4.2.2.161 LL_MIN_SDU_SIZE

```
#define LL_MIN_SDU_SIZE 0x000
```

Minimum value for SDU size.

Definition at line 389 of file ll_defs.h.

4.2.2.162 LL_MAX_SDU_SIZE

```
#define LL_MAX_SDU_SIZE 0xFFF
```

Maximum value for SDU size.

Definition at line 390 of file ll_defs.h.

4.2.2.163 LL_MIN_SDU_INTERV

```
#define LL_MIN_SDU_INTERV 0x000FF
```

Minimum value for SDU interval.

Definition at line 392 of file ll_defs.h.

4.2.2.164 LL_MAX_SDU_INTERV

```
#define LL_MAX_SDU_INTERV 0xFFFFF
```

Maximum value for SDU interval.

Definition at line 393 of file ll_defs.h.

4.2.2.165 LL_MIN_CIS_NSE

```
#define LL_MIN_CIS_NSE 0x01
```

Minimum value for CIS number of subevent.

Definition at line 395 of file ll_defs.h.

4.2.2.166 LL_MAX_CIS_NSE

```
#define LL_MAX_CIS_NSE 0x1F
```

Maximum value for CIS number of subevent.

Definition at line 396 of file ll_defs.h.

4.2.2.167 LL_MIN_CIS_PL

```
#define LL_MIN_CIS_PL 0x0000
```

Minimum value for CIS payload.

Definition at line 398 of file ll_defs.h.

4.2.2.168 LL_MAX_CIS_PL

```
#define LL_MAX_CIS_PL 0x0FFB
```

Maximum value for CIS payload.

Definition at line 399 of file ll_defs.h.

4.2.2.169 LL_MIN_CIS_TRANS_LAT

```
#define LL_MIN_CIS_TRANS_LAT 0x0005
```

Minimum value for CIS transport latency.

Definition at line 401 of file ll_defs.h.

4.2.2.170 LL_MAX_CIS_TRANS_LAT

```
#define LL_MAX_CIS_TRANS_LAT 0x0FA0
```

Maximum value for CIS transport latency.

Definition at line 402 of file ll_defs.h.

4.2.2.171 LL_MIN_CIS_PHY_BIT

```
#define LL_MIN_CIS_PHY_BIT 0x00
```

Minimum value for CIS PHY bit.

Definition at line 404 of file ll_defs.h.

4.2.2.172 LL_MAX_CIS_PHY_BIT

```
#define LL_MAX_CIS_PHY_BIT 0x02
```

Maximum value for CIS PHY bit.

Definition at line 405 of file ll_defs.h.

4.2.2.173 LL_MIN_CIS_FT

```
#define LL_MIN_CIS_FT 0x01
```

Minimum value for CIS flush time.

Definition at line 407 of file ll_defs.h.

4.2.2.174 LL_MAX_CIS_FT

```
#define LL_MAX_CIS_FT 0x1F
```

Maximum value for CIS flush time.

Definition at line 408 of file ll_defs.h.

4.2.2.175 LL_MIN_CIS_BN

```
#define LL_MIN_CIS_BN 0x00
```

Minimum value for CIS burst number.

Definition at line 410 of file ll_defs.h.

4.2.2.176 LL_MAX_CIS_BN

```
#define LL_MAX_CIS_BN 0x10
```

Maximum value for CIS burst number.

Definition at line 411 of file ll_defs.h.

4.2.2.177 LL_MIN_CIS_RTN

```
#define LL_MIN_CIS_RTN 0x00
```

Minimum value for CIS retransmission number.

Definition at line 413 of file ll_defs.h.

4.2.2.178 LL_MAX_CIS_RTN

```
#define LL_MAX_CIS_RTN 0x0F
```

Maximum value for CIS retransmission number.

Definition at line 414 of file ll_defs.h.

4.2.2.179 LL_ISO_TEST_VAR_MIN_LEN

```
#define LL_ISO_TEST_VAR_MIN_LEN 4
```

Minimum length test payload.

Definition at line 451 of file ll_defs.h.

4.2.2.180 LL_ISO_TRANSPORT_LAT_MIN

```
#define LL_ISO_TRANSPORT_LAT_MIN 0x0000EA
```

Minimum transport latency.

Definition at line 453 of file ll_defs.h.

4.2.2.181 LL_DTM_HDR_LEN

```
#define LL_DTM_HDR_LEN 2
```

Direct Test Mode PDU header length.

Definition at line 457 of file ll_defs.h.

4.2.2.182 LL_DTM_SYNC_WORD

```
#define LL_DTM_SYNC_WORD UINT32_C(0x71764129)
```

Direct Test Mode sync word.

Definition at line 458 of file ll_defs.h.

4.2.2.183 LL_DTM_CRC_INIT

```
#define LL_DTM_CRC_INIT UINT32_C(0x555555)
```

Direct Test Mode CRC initial value.

Definition at line 459 of file ll_defs.h.

4.2.2.184 LL_DTM_MAX_INT_US

```
#define LL_DTM_MAX_INT_US 12500
```

Maximum time interval between packets in microseconds.

Definition at line 460 of file ll_defs.h.

4.2.2.185 LL_DTM_PDU_ABS_MAX_LEN

```
#define LL_DTM_PDU_ABS_MAX_LEN 255
```

Absolute maximum DTM PDU length.

Definition at line 461 of file ll_defs.h.

4.2.2.186 LL_DTM_MAX_CHAN_IDX

```
#define LL_DTM_MAX_CHAN_IDX 39
```

Maximum channel index.

Definition at line 462 of file ll_defs.h.

4.2.2.187 LL_CHAN_DATA_MIN_IDX

```
#define LL_CHAN_DATA_MIN_IDX 0
```

Minimum data channel index.

Definition at line 466 of file ll_defs.h.

4.2.2.188 LL_CHAN_DATA_MAX_IDX

```
#define LL_CHAN_DATA_MAX_IDX 36
```

Maximum data channel index.

Definition at line 467 of file ll_defs.h.

4.2.2.189 LL_CHAN_DATA_ALL

```
#define LL_CHAN_DATA_ALL UINT64_C(0x0000001FFFFFFFFF)
```

Maximum data channel index.

Definition at line 468 of file ll_defs.h.

4.2.2.190 LL_BLE_BIT_PER_US

```
#define LL_BLE_BIT_PER_US 1
```

BLE PHY rate.

Definition at line 470 of file ll_defs.h.

4.2.2.191 LL_BLE_US_PER_BYTE_1M

```
#define LL_BLE_US_PER_BYTE_1M 8
```

BLE PHY speed (LE 1M PHY).

Definition at line 471 of file ll_defs.h.

4.2.2.192 LL_BLE_US_PER_BYTE_2M

```
#define LL_BLE_US_PER_BYTE_2M 4
```

BLE PHY speed (LE 2M PHY).

Definition at line 472 of file ll_defs.h.

4.2.2.193 LL_BLE_US_PER_BYTE_CODED_S8

```
#define LL_BLE_US_PER_BYTE_CODED_S8 64
```

BLE PHY speed (LE Coded PHY, S=8).

Definition at line 473 of file ll_defs.h.

4.2.2.194 LL_BLE_US_PER_BIT_CODED_S8

```
#define LL_BLE_US_PER_BIT_CODED_S8 8
```

BLE PHY speed (LE Coded PHY, S=8).

Definition at line 474 of file ll_defs.h.

4.2.2.195 LL_BLE_US_PER_BYTE_CODED_S2

```
#define LL_BLE_US_PER_BYTE_CODED_S2 16
```

BLE PHY speed (LE Coded PHY, S=2).

Definition at line 475 of file ll_defs.h.

4.2.2.196 LL_BLE_US_PER_BIT_CODED_S2

```
#define LL_BLE_US_PER_BIT_CODED_S2 2
```

BLE PHY speed (LE Coded PHY, S=2).

Definition at line 476 of file ll_defs.h.

4.2.2.197 LL_BLE_TIFS_US

```
#define LL_BLE_TIFS_US 150
```

BLE inter-frame space.

Definition at line 477 of file ll_defs.h.

4.2.2.198 LL_BLE_MAFS_US

```
#define LL_BLE_MAFS_US 300
```

BLE minimum AUX frame space.

Definition at line 478 of file ll_defs.h.

4.2.2.199 LL_BLE_US_PER_TICK

```
#define LL_BLE_US_PER_TICK 625
```

Microseconds per BLE tick.

Definition at line 479 of file ll_defs.h.

4.2.2.200 LL_BLE_TMSS_US

```
#define LL_BLE_TMSS_US 150
```

BLE minimum subevent space.

Definition at line 480 of file ll_defs.h.

4.2.2.201 LL_MIN_PKT_TIME_US_1M

```
#define LL_MIN_PKT_TIME_US_1M 80
```

Minimum packet time (i.e. empty PDU) on LE 1M PHY.

Definition at line 482 of file ll_defs.h.

4.2.2.202 LL_MIN_PKT_TIME_US_2M

```
#define LL_MIN_PKT_TIME_US_2M 44
```

Minimum packet time (i.e. empty PDU) on LE 2M PHY.

Definition at line 483 of file ll_defs.h.

4.2.2.203 LL_MIN_PKT_TIME_US_CODED_S8

```
#define LL_MIN_PKT_TIME_US_CODED_S8 720
```

Minimum packet time (i.e. empty PDU) on LE Coded PHY, S=8.

Definition at line 484 of file ll_defs.h.

4.2.2.204 LL_MIN_PKT_TIME_US_CODED_S2

```
#define LL_MIN_PKT_TIME_US_CODED_S2 462
```

Minimum packet time (i.e. empty PDU) on LE Coded PHY, S=2.

Definition at line 485 of file ll_defs.h.

4.2.2.205 LL_MIN_ADV_TX_PWR_LVL

```
#define LL_MIN_ADV_TX_PWR_LVL -20
```

Minimum Tx power level for advertising.

Definition at line 487 of file ll_defs.h.

4.2.2.206 LL_MAX_ADV_TX_PWR_LVL

```
#define LL_MAX_ADV_TX_PWR_LVL 10
```

Maximum Tx power level for advertising.

Definition at line 488 of file ll_defs.h.

4.2.2.207 LL_MIN_TX_PWR_LVL

```
#define LL_MIN_TX_PWR_LVL -30
```

Minimum Tx power level for connections.

Definition at line 490 of file ll_defs.h.

4.2.2.208 LL_MAX_TX_PWR_LVL

```
#define LL_MAX_TX_PWR_LVL 20
```

Maximum Tx power level for connections.

Definition at line 491 of file ll_defs.h.

4.2.2.209 LL_MAX_TIFS_DEVIATION

```
#define LL_MAX_TIFS_DEVIATION 2
```

Maximum TIFS deviation in microseconds.

Definition at line 493 of file ll_defs.h.

4.2.2.210 LL_WW_RX_DEVIATION_USEC

```
#define LL_WW_RX_DEVIATION_USEC 16
```

Rx deviation in microseconds for window widening.

Definition at line 495 of file ll_defs.h.

4.2.2.211 LL_30_USEC_OFFS_MAX_USEC

```
#define LL_30_USEC_OFFS_MAX_USEC 245730
```

Maximum value for 30 microseconds offset unit in microseconds.

Definition at line 497 of file ll_defs.h.

4.2.2.212 LL_ACAD_OPCODE_LEN

```
#define LL_ACAD_OPCODE_LEN 1
```

Length of a single ACAD opcode.

Definition at line 508 of file ll_defs.h.

4.2.2.213 LL_ACAD_LEN_FIELD_LEN

```
#define LL_ACAD_LEN_FIELD_LEN 1
```

Length of ACAD length field

Definition at line 509 of file ll_defs.h.

4.2.2.214 LL_ACAD_DATA_FIELD_MAX_LEN

```
#define LL_ACAD_DATA_FIELD_MAX_LEN (LL_EXT_HDR_ACAD_MAX_LEN - LL_ACAD_OPCODE_LEN - LL_ACAD_LEN_FIELD_LEN)
```

Length of max ACAD field length without opcode and length field

Definition at line 510 of file ll_defs.h.

4.2.2.215 LL_ACAD_CHAN_MAP_UPD_LEN

```
#define LL_ACAD_CHAN_MAP_UPD_LEN 8
```

Length of ACAD update periodic channel map data field.

Definition at line 512 of file ll_defs.h.

4.2.2.216 LL_ACAD_BIG_INFO_UNENCRPT_LEN

```
#define LL_ACAD_BIG_INFO_UNENCRPT_LEN 33
```

Length of ACAD for an unencrypted BIG Info.

Definition at line 513 of file ll_defs.h.

4.2.2.217 LL_ACAD_BIG_INFO_ENCRPT_LEN

```
#define LL_ACAD_BIG_INFO_ENCRPT_LEN 57
```

Length of ACAD for an encrypted BIG Info.

Definition at line 514 of file ll_defs.h.

4.2.2.218 LL_BIG_OPCODE_LEN

```
#define LL_BIG_OPCODE_LEN 1
```

Length of a BIG Control PDU opcode.

Definition at line 526 of file ll_defs.h.

4.2.2.219 LL_BIG_CHAN_MAP_IND_PDU_LEN

```
#define LL_BIG_CHAN_MAP_IND_PDU_LEN 7
```

BIG Channel Map indication PDU length.

Definition at line 528 of file ll_defs.h.

4.2.2.220 LL_BIG_TERMINATE_IND_PDU_LEN

```
#define LL_BIG_TERMINATE_IND_PDU_LEN 3
```

BIG Terminate indication PDU length.

Definition at line 529 of file ll_defs.h.

4.2.2.221 LL_BIG_MIN_INSTANT

```
#define LL_BIG_MIN_INSTANT 6
```

Minimum number of BIG Events to apply a BIG Control PDU.

Definition at line 531 of file ll_defs.h.

4.2.2.222 LL_BIG_CONTROL_ACCESS_ADDR

```
#define LL_BIG_CONTROL_ACCESS_ADDR UINT32_C(0x7A412493)
```

BIG Control access address.

Definition at line 533 of file ll_defs.h.

4.2.2.223 LL_SCA_MIN_INDEX

```
#define LL_SCA_MIN_INDEX 0
```

Index for lowest accuracy clock.

Definition at line 546 of file ll_defs.h.

4.2.2.224 LL_SCA_MAX_INDEX

```
#define LL_SCA_MAX_INDEX 7
```

Index for highest accuracy clock.

Definition at line 547 of file ll_defs.h.

4.2.2.225 LL_PWR_CONTROL_LIMIT_MIN_BIT

```
#define LL_PWR_CONTROL_LIMIT_MIN_BIT (1 << 0)
```

Power control Limit field.

txPower is at it's minimum limit.

Definition at line 552 of file ll_defs.h.

4.2.2.226 LL_PWR_CONTROL_LIMIT_MAX_BIT

```
#define LL_PWR_CONTROL_LIMIT_MAX_BIT (1 << 1)
```

txPower is at it's maximum limit.

Definition at line 553 of file ll_defs.h.

4.2.2.227 LL_PWR_CTRL_APR_UNDEF

```
#define LL_PWR_CTRL_APR_UNDEF 0xFF
```

Power control APR field.

Undefined power reduction limit.

Definition at line 556 of file ll_defs.h.

4.2.2.228 LL_PWR_CTRL_TXPOWER_MAX

```
#define LL_PWR_CTRL_TXPOWER_MAX 0x7F
```

Power control txPower field.

txPower symbol for maximum txPower.

Definition at line 559 of file ll_defs.h.

4.2.2.229 LL_PWR_CTRL_TXPOWER_MIN

```
#define LL_PWR_CTRL_TXPOWER_MIN 0x7E
```

txPower symbol for minimum txPower.

Definition at line 560 of file ll_defs.h.

4.2.2.230 LL_PWR_CTRL_TXPOWER_UNAVAILABLE

```
#define LL_PWR_CTRL_TXPOWER_UNAVAILABLE 127
```

Power is unavailable for this PHY.

Definition at line 561 of file ll_defs.h.

4.2.2.231 LL_PWR_CTRL_TXPOWER_UNMANAGED

```
#define LL_PWR_CTRL_TXPOWER_UNMANAGED 126
```

Power is unmanaged for this PHY.

Definition at line 562 of file ll_defs.h.

4.2.2.232 LL_ISOAL_SEG_HDR_MASK_SC

```
#define LL_ISOAL_SEG_HDR_MASK_SC 0x01
```

Mask for segmentation header start/continuation bit.

Definition at line 582 of file ll_defs.h.

4.2.2.233 LL_ISOAL_SEG_HDR_MASK_CMPLT

```
#define LL_ISOAL_SEG_HDR_MASK_CMPLT 0x02
```

Mask for segmentation header complete bit.

Definition at line 583 of file ll_defs.h.

4.2.3 Enumeration Type Documentation

4.2.3.1 anonymous enum

```
anonymous enum
```

Advertising channel PDU types.

Enumerator

LL_PDU_ADV_IND	Connectable undirected advertising PDU.
LL_PDU_ADV_DIRECT_IND	Connectable directed advertising PDU.
LL_PDU_ADV_NONCONN_IND	Non-connectable undirected advertising PDU.
LL_PDU_SCAN_REQ	Scan request PDU.
LL_PDU_SCAN_RSP	Scan response PDU.
LL_PDU_CONNECT_IND	Connect indication PDU.
LL_PDU_ADV_SCAN_IND	Scannable undirected advertising PDU.
LL_PDU_AUX_SCAN_REQ	Auxiliary scan request PDU.
LL_PDU_AUX_CONNECT_REQ	Auxiliary connect request PDU.
LL_PDU_ADV_EXT_IND	Extended advertising PDU.
LL_PDU_AUX_ADV_IND	Auxiliary advertising PDU.
LL_PDU_AUX_SCAN_RSP	Auxiliary scan response PDU.
LL_PDU_AUX_SYNC_IND	Auxiliary synchronize PDU.
LL_PDU_AUX_CHAIN_IND	Auxiliary chain PDU.
LL_PDU_AUX_CONNECT_RSP	Auxiliary connect response PDU.

Definition at line 71 of file ll_defs.h.

```
{
  /* --- Core Spec 4.0 --- */
  LL_PDU_ADV_IND                = 0,    /*!< Connectable undirected advertising PDU. */
  LL_PDU_ADV_DIRECT_IND         = 1,    /*!< Connectable directed advertising PDU. */
  LL_PDU_ADV_NONCONN_IND        = 2,    /*!< Non-connectable undirected advertising
    PDU. */
  LL_PDU_SCAN_REQ               = 3,    /*!< Scan request PDU. */
  LL_PDU_SCAN_RSP               = 4,    /*!< Scan response PDU. */
  LL_PDU_CONNECT_IND            = 5,    /*!< Connect indication PDU. */
  LL_PDU_ADV_SCAN_IND           = 6,    /*!< Scannable undirected advertising PDU. */
  /* --- Core Spec 5.0 --- */
  LL_PDU_AUX_SCAN_REQ           = 3,    /*!< Auxiliary scan request PDU. */
  LL_PDU_AUX_CONNECT_REQ        = 5,    /*!< Auxiliary connect request PDU. */
  LL_PDU_ADV_EXT_IND            = 7,    /*!< Extended advertising PDU. */
  LL_PDU_AUX_ADV_IND            = 7,    /*!< Auxiliary advertising PDU. */
  LL_PDU_AUX_SCAN_RSP           = 7,    /*!< Auxiliary scan response PDU. */
  LL_PDU_AUX_SYNC_IND           = 7,    /*!< Auxiliary synchronize PDU. */
  LL_PDU_AUX_CHAIN_IND          = 7,    /*!< Auxiliary chain PDU. */
  LL_PDU_AUX_CONNECT_RSP        = 8,    /*!< Auxiliary connect response PDU. */
};
```

4.2.3.2 anonymous enum

```
anonymous enum
```

Extended header bit definition.

Enumerator

LL_EXT_HDR_ADV_ADDR_BIT	Extended header AdvA bit.
LL_EXT_HDR_TGT_ADDR_BIT	Extended header TargetA bit.
LL_EXT_HDR_CTE_INFO_BIT	Extended header CTEInfo bit.
LL_EXT_HDR_ADI_BIT	Extended header AdvDataInfo bit.
LL_EXT_HDR_AUX_PTR_BIT	Extended header AuxPtr bit.
LL_EXT_HDR_SYNC_INFO_BIT	Extended header SyncInfo bit.
LL_EXT_HDR_TX_PWR_BIT	Extended header TxPower bit.

Definition at line 134 of file ll_defs.h.

```
{
  LL_EXT_HDR_ADV_ADDR_BIT    = (1 << 0), /*!< Extended header AdvA bit. */
  LL_EXT_HDR_TGT_ADDR_BIT    = (1 << 1), /*!< Extended header TargetA bit. */
  LL_EXT_HDR_CTE_INFO_BIT    = (1 << 2), /*!< Extended header CTEInfo bit. */
  LL_EXT_HDR_ADI_BIT         = (1 << 3), /*!< Extended header AdvDataInfo bit. */
  LL_EXT_HDR_AUX_PTR_BIT     = (1 << 4), /*!< Extended header AuxPtr bit. */
  LL_EXT_HDR_SYNC_INFO_BIT   = (1 << 5), /*!< Extended header SyncInfo bit. */
  LL_EXT_HDR_TX_PWR_BIT      = (1 << 6), /*!< Extended header TxPower bit. */
};
```

4.2.3.3 anonymous enum

`anonymous enum`

Periodic sync transfer receive mode.

Enumerator

LL_SYNC_TRSF_MODE_OFF	Periodic sync transfer receive is disabled.
LL_SYNC_TRSF_MODE_REP_DISABLED	Periodic sync transfer receive is enabled, report event is disabled.
LL_SYNC_TRSF_MODE_REP_ENABLED	Periodic sync transfer receive is enabled, report event is enabled.

Definition at line 181 of file ll_defs.h.

```
{
  LL_SYNC_TRSF_MODE_OFF           = 0,     /*!< Periodic sync transfer receive is
   disabled. */
  LL_SYNC_TRSF_MODE_REP_DISABLED  = 1,     /*!< Periodic sync transfer receive
   is enabled, report event is disabled. */
  LL_SYNC_TRSF_MODE_REP_ENABLED   = 2,     /*!< Periodic sync transfer receive
   is enabled, report event is enabled. */
  LL_SYNC_TRSF_MAX_MODE = LL_SYNC_TRSF_MODE_REP_ENABLED
};
```

4.2.3.4 anonymous enum

`anonymous enum`

Data channel LL Control PDU types.

Enumerator

LL_PDU_CONN_UPDATE_IND	Connection update indication PDU.
LL_PDU_CHANNEL_MAP_IND	Channel map indication PDU.
LL_PDU_TERMINATE_IND	Terminate indication PDU.
LL_PDU_ENC_REQ	Encryption request PDU.
LL_PDU_ENC_RSP	Encryption response PDU.
LL_PDU_START_ENC_REQ	Start encryption request PDU.

Enumerator

LL_PDU_START_ENC_RSP	Start encryption response PDU.
LL_PDU_UNKNOWN_RSP	Unknown response PDU.
LL_PDU_FEATURE_REQ	Feature request PDU.
LL_PDU_FEATURE_RSP	Feature response PDU.
LL_PDU_PAUSE_ENC_REQ	Pause encryption request PDU.
LL_PDU_PAUSE_ENC_RSP	Pause encryption response PDU.
LL_PDU_VERSION_IND	Version indication PDU.
LL_PDU_REJECT_IND	Reject indication PDU.
LL_PDU_SLV_FEATURE_REQ	Slave feature request PDU.
LL_PDU_CONN_PARAM_REQ	Connection parameter request PDU.
LL_PDU_CONN_PARAM_RSP	Connection parameter response PDU.
LL_PDU_REJECT_EXT_IND	Reject extended indication PDU.
LL_PDU_PING_REQ	Ping request PDU.
LL_PDU_PING_RSP	Ping response PDU.
LL_PDU_LENGTH_REQ	Data length request PDU.
LL_PDU_LENGTH_RSP	Data length response PDU.
LL_PDU_PHY_REQ	PHY request PDU.
LL_PDU_PHY_RSP	PHY response PDU.
LL_PDU_PHY_UPDATE_IND	PHY update indication PDU.
LL_PDU_MIN_USED_CHAN_IND	Minimum used channels indication PDU.
LL_PDU_PERIODIC_SYNC_IND	Periodic sync indication PDU.
LL_PDU_PEER_SCA_REQ	Peer SCA request PDU.
LL_PDU_PEER_SCA_RSP	Peer SCA response PDU.
LL_PDU_CIS_REQ	CIS request PDU.
LL_PDU_CIS_RSP	CIS response PDU.
LL_PDU_CIS_IND	CIS indication PDU.
LL_PDU_CIS_TERM_IND	CIS terminate indication PDU.
LL_PDU_PWR_CTRL_REQ	Power Control request.
LL_PDU_PWR_CTRL_RSP	Power Control response.
LL_PDU_PWR_CHANGE_IND	Transmit power change indication PDU.
LL_PDU_UNSPECIFIED	Unspecified PDU.

Definition at line 192 of file ll_defs.h.

```
{
  /* --- Core Spec 4.0 --- */
  LL_PDU_CONN_UPDATE_IND         = 0x00, /*!< Connection update indication PDU. */
  LL_PDU_CHANNEL_MAP_IND         = 0x01, /*!< Channel map indication PDU. */
  LL_PDU_TERMINATE_IND           = 0x02, /*!< Terminate indication PDU. */
  LL_PDU_ENC_REQ                 = 0x03, /*!< Encryption request PDU. */
  LL_PDU_ENC_RSP                 = 0x04, /*!< Encryption response PDU. */
  LL_PDU_START_ENC_REQ           = 0x05, /*!< Start encryption request PDU. */
  LL_PDU_START_ENC_RSP           = 0x06, /*!< Start encryption response PDU. */
  LL_PDU_UNKNOWN_RSP             = 0x07, /*!< Unknown response PDU. */
  LL_PDU_FEATURE_REQ             = 0x08, /*!< Feature request PDU. */
  LL_PDU_FEATURE_RSP             = 0x09, /*!< Feature response PDU. */
  LL_PDU_PAUSE_ENC_REQ           = 0x0A, /*!< Pause encryption request PDU. */
  LL_PDU_PAUSE_ENC_RSP           = 0x0B, /*!< Pause encryption response PDU. */
  LL_PDU_VERSION_IND             = 0x0C, /*!< Version indication PDU. */
  LL_PDU_REJECT_IND              = 0x0D, /*!< Reject indication PDU. */
  /* --- Core Spec 4.2 --- */
  LL_PDU_SLV_FEATURE_REQ         = 0x0E, /*!< Slave feature request PDU. */
  LL_PDU_CONN_PARAM_REQ          = 0x0F, /*!< Connection parameter request PDU. */
  LL_PDU_CONN_PARAM_RSP          = 0x10, /*!< Connection parameter response PDU. */
  LL_PDU_REJECT_EXT_IND          = 0x11, /*!< Reject extended indication PDU. */
  LL_PDU_PING_REQ               = 0x12, /*!< Ping request PDU. */
  LL_PDU_PING_RSP               = 0x13, /*!< Ping response PDU. */
  LL_PDU_LENGTH_REQ             = 0x14, /*!< Data length request PDU. */
  LL_PDU_LENGTH_RSP             = 0x15, /*!< Data length response PDU. */
  /* --- Core Spec 5.0 --- */
  LL_PDU_PHY_REQ                = 0x16, /*!< PHY request PDU. */
  LL_PDU_PHY_RSP                = 0x17, /*!< PHY response PDU. */
  LL_PDU_PHY_UPDATE_IND         = 0x18, /*!< PHY update indication PDU. */
  LL_PDU_MIN_USED_CHAN_IND      = 0x19, /*!< Minimum used channels indication PDU. */
  /* --- Core Spec 5.1 --- */
  /* 0x1A 0x1B for AOA AOD, 0x1C for PAST 0x20 RFU */
  LL_PDU_PERIODIC_SYNC_IND      = 0x1C, /*!< Periodic sync indication PDU. */
  /* --- Core Spec 5.2 --- */
  LL_PDU_PEER_SCA_REQ           = 0x1D, /*!< Peer SCA request PDU. */
  LL_PDU_PEER_SCA_RSP           = 0x1E, /*!< Peer SCA response PDU. */
  LL_PDU_CIS_REQ                = 0x1F, /*!< CIS request PDU. */
  LL_PDU_CIS_RSP                = 0x20, /*!< CIS response PDU. */
  LL_PDU_CIS_IND                = 0x21, /*!< CIS indication PDU. */
  LL_PDU_CIS_TERM_IND           = 0x22, /*!< CIS terminate indication PDU. */
  LL_PDU_PWR_CTRL_REQ           = 0x23, /*!< Power Control request. */
  LL_PDU_PWR_CTRL_RSP           = 0x24, /*!< Power Control response. */
  LL_PDU_PWR_CHANGE_IND         = 0x25, /*!< Transmit power change indication PDU. */

  LL_PDU_UNSPECIFIED            = 0xFF  /*!< Unspecified PDU. */
};
```

4.2.3.5 anonymous enum

anonymous enum

Data PDU LLID types.

Enumerator

LL_LLID_VS_PDU	Vendor specific PDU.
LL_LLID_EMPTY_PDU	Empty PDU.
LL_LLID_CONT_PDU	Data PDU: continuation fragment of an L2CAP message.
LL_LLID_START_PDU	Data PDU: start of an L2CAP message or a complete L2CAP message with no fragmentation.
LL_LLID_CTRL_PDU	Control PDU.

Definition at line 291 of file ll_defs.h.

```
{
  LL_LLID_VS_PDU                    = 0x00,   /*!< Vendor specific PDU. */
  /* N.B. next two enumerations intentionally use identical values. */
  LL_LLID_EMPTY_PDU                 = 0x01,   /*!< Empty PDU. */
  LL_LLID_CONT_PDU                  = 0x01,   /*!< Data PDU: continuation fragment of an L2CAP message. */
  LL_LLID_START_PDU                 = 0x02,   /*!< Data PDU: start of an L2CAP message or a complete L2CAP message with no fragmentation. */
  LL_LLID_CTRL_PDU                  = 0x03,   /*!< Control PDU. */
};
```

4.2.3.6 LlIsoLlid_t

enum LlIsoLlid_t

ISO PDU LLID types.

Enumerator

LL_LLID_ISO_UNF_END_PDU	Unframed ISO Data PDU; end fragment of a SDU.
LL_LLID_ISO_UNF_CONT_PDU	Unframed ISO Data PDU; start or continuation fragment of a SDU.
LL_LLID_ISO_FRA_PDU	Framed ISO Data PDU; one or more segments of a SDU.
LL_LLID_ISO_EMPTY_PDU	ISO Empty PDU.
LL_LLID_BIG_CTRL_PDU	BIG Control PDU.

Definition at line 417 of file ll_defs.h.

```
427 {
```

4.2.3.7 LlFraming_t

```
enum LlFraming_t
```

ISO PDU type.

Enumerator

LL_ISO_PDU_TYPE_UNFRAMED	Unframed PDU type.
LL_ISO_PDU_TYPE_FRAMED	Framed PDU type.

Definition at line 430 of file ll_defs.h.

```
434 {
```

4.2.3.8 anonymous enum

```
anonymous enum
```

ISO test payload types.

Enumerator

LL_ISO_TEST_PL_LEN_ZERO	Zero length test payload
LL_ISO_TEST_PL_LEN_VAR	Variable length test payload
LL_ISO_TEST_PL_LEN_MAX	Maximum length test payload

Definition at line 437 of file ll_defs.h.

```
442 {
```

4.2.3.9 anonymous enum

```
anonymous enum
```

ISOAL SDU assembly states.

Enumerator

LL_ISO_SDU_STATE_NEW	New SDU state.
LL_ISO_SDU_STATE_CONT	SDU pending state.

Definition at line 445 of file ll_defs.h.

```
499 {
```

4.2.3.10 anonymous enum

```
anonymous enum
```

ACAD opcodes.

Enumerator

LL_ACAD_OPCODE_CHAN_MAP_UPD	Opcode for ACAD Channel Map Update Indication.
LL_ACAD_OPCODE_BIG_INFO	Opcode for ACAD BIG Info.

Definition at line 502 of file ll_defs.h.

```
516 {
```

4.2.3.11 anonymous enum

```
anonymous enum
```

BIG Control opcodes.

Enumerator

LL_BIG_OPCODE_CHAN_MAP_IND	Opcode for BIG Channel Map Indication.
LL_BIG_OPCODE_BIG_TERM_IND	Opcode for BIG Terminate Indication.
LL_BIG_OPCODE_MAX	Maximum number of BIG opcodes.

Definition at line 519 of file ll_defs.h.

```
536 {
```

4.2.3.12 anonymous enum

```
anonymous enum
```

Action parameter.

Enumerator

LL_MODIFY_SCA_MORE_ACCURATE	Modify to more accurate clock accuracy.
LL_MODIFY_SCA_LESS_ACCURATE	Modify to less accurate clock accuracy.
LL_MODIFY_SCA_NO_ACTION	No action (Used for request sca tester command).

Definition at line 539 of file ll_defs.h.

```
562 {
```

4.2.3.13 anonymous enum

```
anonymous enum
```

Path loss monitoring zones.

Enumerator

LL_PATH_LOSS_ZONE_LOW	Low path loss zone.
LL_PATH_LOSS_ZONE_MID	Medium path loss zone.
LL_PATH_LOSS_ZONE_HIGH	High path loss zone.

Definition at line 565 of file ll_defs.h.

```
570 {
```

4.2.3.14 anonymous enum

```
anonymous enum
```

brief Power control reporting reasons table.

Enumerator

LL_POWER_REPORT_REASON_LOCAL	Local power change report.
LL_POWER_REPORT_REASON_REMOTE	Remote power change report.
LL_POWER_REPORT_REASON_READ_REMOTE	Remote transmit power read report.

Definition at line 573 of file ll_defs.h.

```
583 {
```

4.2.3.15 anonymous enum

```
anonymous enum
```

Codec transport types.

Enumerator

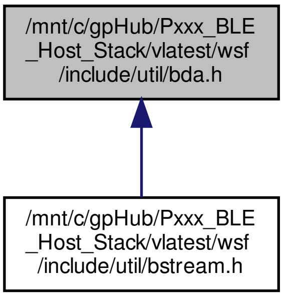
LL_CODEC_TRANS_CIS_BIT	Codec supported over LE CIS.
LL_CODEC_TRANS_BIS_BIT	Codec supported over LE BIS.

Definition at line 586 of file ll_defs.h.

4.3 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/bda.h File Reference

Bluetooth device address utilities.

This graph shows which files directly or indirectly include this file:


/mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/bda.h
/mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/bstream.h

Macros

- #define BDA_ADDR_LEN 6

  BD address length.
- #define BDA_ADDR_STR_LEN (BDA_ADDR_LEN ∗ 2)

  BD address string length.
- #define BDA_ADDR_IS_RPA(bda) (((bda)[5] & 0xC0) == 0x40)

  BDA RPA check.
- #define BDA_ADDR_IS_NRPA(bda) (((bda)[5] & 0xC0) == 0x00)

  BDA NRPA check.
- #define BDA_ADDR_IS_STATIC(bda) (((bda)[5] & 0xC0) == 0xC0)

  BDA static random check.
- #define BDA64_ADDR_IS_RPA(bda64) ((((bda64) >> 40) & 0xC0) == 0x40)

  BDA64 RPA check.
- #define BDA64_ADDR_IS_NRPA(bda64) ((((bda64) >> 40) & 0xC0) == 0x00)

  BDA64 NRPA check.
- #define BDA64_ADDR_IS_STATIC(bda64) ((((bda64) >> 40) & 0xC0) == 0xC0)

  BDA64 static random check.

Typedefs

- typedef uint8_t bdAddr_t[BDA_ADDR_LEN]

  BD address data type.

Functions

- void BdaCpy (uint8_t ∗pDst, const uint8_t ∗pSrc)

  Copy a BD address from source to destination.
- bool_t BdaCmp (const uint8_t ∗pAddr1, const uint8_t ∗pAddr2)

  Compare two BD addresses.
- uint8_t ∗ BdaClr (uint8_t ∗pDst)

  Set a BD address to all zeros.
- bool_t BdaIsZeros (const uint8_t ∗pAddr)

  Check if a BD address is all zeros.
- char ∗ Bda2Str (const uint8_t ∗pAddr)

  Convert a BD address to a string.

4.3.1 Detailed Description

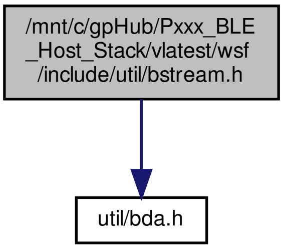
Bluetooth device address utilities.

4.4 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/bstream.h File Reference

Byte stream to integer conversion macros.

```
#include "util/bda.h"
```

Include dependency graph for bstream.h:



Macros

Macros for converting a little endian byte buffer to integers.

- #define BYTES_TO_INT16(n, p) {n = ((int16_t)(p)[0] + ((int16_t)(p)[1] << 8));}
  convert little endian byte buffer to int16_t.
- #define BYTES_TO_UINT16(n, p) {n = ((uint16_t)(p)[0] + ((uint16_t)(p)[1] << 8));}
  convert little endian byte buffer to uint16_t.
- #define BYTES_TO_UINT24(n, p)
  convert little endian byte buffer to uint24_t.
- #define BYTES_TO_UINT32(n, p)
  convert little endian byte buffer to uint32_t.
- #define BYTES_TO_UINT40(n, p)
  convert little endian byte buffer to uint40_t.
- #define BYTES_TO_UINT64(n, p)
  convert little endian byte buffer to uint64_t.

Macros for converting a big endian byte buffer to integers.

- #define BYTES_BE_TO_UINT16(n, p) {n = ((uint16_t)(p)[1] + ((uint16_t)(p)[0] << 8));}
  convert big endian byte buffer to uint16_t.
- #define BYTES_BE_TO_UINT24(n, p)
  convert big endian byte buffer to 24-bit uint32_t (MSB 0).
- #define BYTES_BE_TO_UINT32(n, p)
  convert big endian byte buffer to uint32_t.

Macros for converting little endian integers to array of bytes

- #define UINT16_TO_BYTES(n) ((uint8_t) (n)), ((uint8_t)((n) >> 8))
  convert little endian uint16_t to array of bytes.

- #define UINT32_TO_BYTES(n) ((uint8_t) (n)), ((uint8_t)((n) >> 8)), ((uint8_t)((n) >> 16)), ((uint8_t)((n) >> 24))

  convert little endian uint32_t to array of bytes.

Macros for converting big endian integers to array of bytes

- #define UINT16_TO_BE_BYTES(n) ((uint8_t)((n) >> 8)), ((uint8_t) (n))

  convert big endian uint16_t to array of bytes.
- #define UINT24_TO_BE_BYTES(n) ((uint8_t)((n) >> 16)), ((uint8_t)((n) >> 8)), ((uint8_t) (n))

  convert 24-bit big endian uint32_t (MSB 0) to array of bytes.
- #define UINT32_TO_BE_BYTES(n) ((uint8_t)((n) >> 24)), ((uint8_t)((n) >> 16)), ((uint8_t)((n) >> 8)), ((uint8_t) (n))

  convert big endian uint32_t to array of bytes.

Macros for converting little endian integers to single bytes

- #define UINT16_TO_BYTE0(n) ((uint8_t) (n))

  convert little endian uint16_t to byte 0.
- #define UINT16_TO_BYTE1(n) ((uint8_t) ((n) >> 8))

  convert little endian uint16_t to byte 1.
- #define UINT32_TO_BYTE0(n) ((uint8_t) (n))

  convert little endian uint32_t to byte 0.
- #define UINT32_TO_BYTE1(n) ((uint8_t) ((n) >> 8))

  convert little endian uint32_t to byte 1.
- #define UINT32_TO_BYTE2(n) ((uint8_t) ((n) >> 16))

  convert little endian uint32_t to byte 2.
- #define UINT32_TO_BYTE3(n) ((uint8_t) ((n) >> 24))

  convert little endian uint32_t to byte 3.

Macros for converting a little endian byte stream to integers, with increment.

- #define BSTREAM_TO_INT8(n, p) {n = (int8_t)(∗(p)++);}

  convert little endian byte stream to uint8_t, incrementing one byte.
- #define BSTREAM_TO_UINT8(n, p) {n = (uint8_t)(∗(p)++);}

  convert little endian byte stream to int8_t, incrementing one byte.
- #define BSTREAM_TO_INT16(n, p) {BYTES_TO_INT16(n, p); p += 2;}

  convert little endian byte stream to int16_t, incrementing two bytes.
- #define BSTREAM_TO_UINT16(n, p) {BYTES_TO_UINT16(n, p); p += 2;}

  convert little endian byte stream to uint16_t, incrementing two bytes.
- #define BSTREAM_TO_UINT24(n, p) {BYTES_TO_UINT24(n, p); p += 3;}

  convert little endian byte stream to uint24_t, incrementing three bytes.
- #define BSTREAM_TO_UINT32(n, p) {BYTES_TO_UINT32(n, p); p += 4;}

  convert little endian byte stream to uint32_t, incrementing four bytes.
- #define BSTREAM_TO_UINT40(n, p) {BYTES_TO_UINT40(n, p); p += 5;}

  convert little endian byte stream to uint40_t, incrementing five bytes.
- #define BSTREAM_TO_UINT64(n, p) {n = BstreamToUint64(p); p += 8;}

  convert little endian byte stream to uint64_t, incrementing eigth bytes.
- #define BSTREAM_TO_BDA(bda, p) {BdaCpy(bda, p); p += BDA_ADDR_LEN;}

  convert little endian byte stream to six byte Bluetooth device address, incrementing six bytes.
- #define BSTREAM_TO_BDA64(bda, p) {bda = BstreamToBda64(p); p += BDA_ADDR_LEN;}

  convert little endian byte stream to eight byte Bluetooth device address, incrementing eight bytes.

Macros for converting a big endian byte stream to integers, with increment.

- #define BSTREAM_BE_TO_UINT16(n, p) {BYTES_BE_TO_UINT16(n, p); p += 2;}

  convert big endian byte stream to uint16_t, incrementing one byte.

- #define BSTREAM_BE_TO_UINT24(n, p) {BYTES_BE_TO_UINT24(n, p); p += 3;}

  convert big endian byte stream to 24-bit uint32_t (MSB 0), incrementing one byte.

Macros for converting integers to a little endian byte stream, with increment.

- #define UINT8_TO_BSTREAM(p, n) {∗(p)++ = (uint8_t)(n);}

  convert uint8_t to little endian byte stream, incrementing one byte.
- #define UINT16_TO_BSTREAM(p, n) {∗(p)++ = (uint8_t)(n); ∗(p)++ = (uint8_t)((n) >> 8);}

  convert uint16_t to little endian byte stream, incrementing two bytes.
- #define UINT24_TO_BSTREAM(p, n)

  convert uint24_t to little endian byte stream, incrementing three bytes.
- #define UINT32_TO_BSTREAM(p, n)

  convert uint32_t to little endian byte stream, incrementing four bytes.
- #define UINT40_TO_BSTREAM(p, n)

  convert uint40_t to little endian byte stream, incrementing five bytes.
- #define UINT64_TO_BSTREAM(p, n) {Uint64ToBstream(p, n); p += sizeof(uint64_t);}

  convert uint64_t to little endian byte stream, incrementing eight bytes.
- #define BDA_TO_BSTREAM(p, bda) {BdaCpy(p, bda); p += BDA_ADDR_LEN;}

  convert six byte Bluetooth device address to little endian byte stream, incrementing six bytes.
- #define BDA64_TO_BSTREAM(p, bda) {Bda64ToBstream(p, bda); p += BDA_ADDR_LEN;}

  convert eight byte Bluetooth device address to little endian byte stream, incrementing eigth bytes.

Macros for converting integers to a big endian byte stream, with increment.

- #define UINT16_TO_BE_BSTREAM(p, n) {∗(p)++ = (uint8_t)((n) >> 8); ∗(p)++ = (uint8_t)(n);}

  convert uint16_t to big endian byte stream, incrementing one byte.
- #define UINT32_TO_BE_BSTREAM(p, n)

  convert uint32_t to big endian byte stream, incrementing one byte.

Macros for converting integers to a little endian byte stream, without increment.

- #define UINT16_TO_BUF(p, n) {(p)[0] = (uint8_t)(n); (p)[1] = (uint8_t)((n) >> 8);}

  convert uint16_t to little endian byte stream.
- #define UINT24_TO_BUF(p, n)

  convert uint24_t to little endian byte stream.
- #define UINT32_TO_BUF(p, n)

  convert uint32_t to little endian byte stream.
- #define UINT40_TO_BUF(p, n)

  convert uint40_t to little endian byte stream.

Macros for converting integers to a big endian byte stream, without increment.

- #define UINT16_TO_BE_BUF(p, n) {(p)[0] = (uint8_t)((n) >> 8); (p)[1] = (uint8_t)(n);}

  convert uint16_t to big endian byte stream.
- #define UINT24_TO_BE_BUF(p, n)

  convert 24-bit uint32_t (MSB 0) to big endian byte stream.
- #define UINT32_TO_BE_BUF(p, n)

  convert uint32_t to big endian byte stream.

Macros for comparing a little endian byte buffer to integers.

- #define BYTES_UINT16_CMP(p, n) ((p)[1] == UINT16_TO_BYTE1(n) && (p)[0] == UINT16_TO_BYTE0(n))

  compare 2 byte little endian buffer with a uint16_t.

Macros for IEEE FLOAT type: exponent = byte 3, mantissa = bytes 2-0

- #define FLT_TO_UINT32(m, e) ((m) | ((int32_t)(e) ∗ 16777216))

  Convert float to uint32.
- #define UINT32_TO_FLT(m, e, n) {m = UINT32_TO_FLT_M(n); e = UINT32_TO_FLT_E(n);}

  Convert uint32_t to float.
- #define UINT32_TO_FLT_M(n)

  Convert uint32_t to float mantissa component.
- #define UINT32_TO_FLT_E(n) ((int8_t)(n >> 24))

  Convert uint32_t to float exponent component.

Macros for IEEE SFLOAT type: exponent = bits 15-12, mantissa = bits 11-0

- #define SFLT_TO_UINT16(m, e) ((m) | (0xF000 & ((int16_t)(e) ∗ 4096)))

  Convert sfloat to uint16_t.
- #define UINT16_TO_SFLT(m, e, n) {m = UINT16_TO_SFLT_M(n); e = UINT16_TO_SFLT_E(n);}

  Convert uint16_t to sfloat.
- #define UINT16_TO_SFLT_M(n)

  Convert uint16_T to sfloat mantissa component.
- #define UINT16_TO_SFLT_E(n)

  Convert uint16_T to sfloat exponent component.

Functions

- uint64_t BstreamToBda64 (const uint8_t ∗p)

  Convert bstream to BDA64.
- uint64_t BstreamToUint64 (const uint8_t ∗p)

  Convert bstream to uint64_t.
- void Bda64ToBstream (uint8_t ∗p, uint64_t bda)

  Convert BDA64 to bstream.
- void Uint64ToBstream (uint8_t ∗p, uint64_t n)

  Convert uint64_t to bstream.

4.4.1 Detailed Description

Byte stream to integer conversion macros.

4.5 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/calc128.h File Reference

128-bit integer utilities.

Macros

- #define CALC128_LEN 16

  128-bit integer length in bytes

Functions

- void Calc128Cpy (uint8_t *pDst, uint8_t *pSrc)

  Copy a 128-bit integer from source to destination.
- void Calc128Cpy64 (uint8_t *pDst, uint8_t *pSrc)

  Copy a 64-bit integer from source to destination.
- void Calc128Xor (uint8_t *pDst, uint8_t *pSrc)

  Exclusive-or two 128-bit integers and return the result in pDst.

Variables

- const uint8_t calc128Zeros [CALC128_LEN]

  128-bit zero value

4.5.1 Detailed Description

128-bit integer utilities.

4.6 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/crc32.h File Reference

CRC-32 utilities.

Functions

- uint32_t CalcCrc32 (uint32_t crcInit, uint32_t len, const uint8_t ∗pBuf)

  Calculate the CRC-32 of the given buffer.

4.6.1 Detailed Description

CRC-32 utilities.

4.7 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/fcs.h File Reference

FCS utilities (3GPP TS 27.010).

Functions

- uint8_t FcsCalc (uint32_t len, const uint8_t ∗pBuf)

  Calculate the FCS of the given buffer.
- void FcsAddByte (uint8_t ∗pFcs, uint8_t byte)

  Computes resultant CRC by appending one byte.

4.7.1 Detailed Description

FCS utilities (3GPP TS 27.010).

4.7.2 Function Documentation

4.7.2.1 FcsCalc()

```
uint8_t FcsCalc (
             uint32_t len,
             const uint8_t * pBuf )
```

Calculate the FCS of the given buffer.

Parameters

len	Length of the buffer.
pBuf	Buffer to compute the CRC.

Returns

FCS value.

4.7.2.2 FcsAddByte()

```
void FcsAddByte (
            uint8_t * pFcs,
            uint8_t byte )
```

Computes resultant CRC by appending one byte.

Parameters

pFcs	CRC value on which to append the byte.
byte	Byte to be added to CRC computation.

Returns

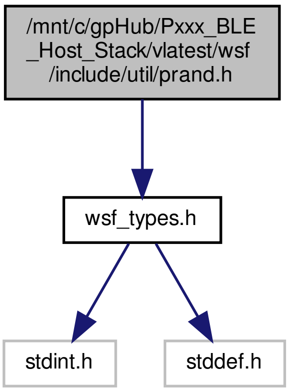
FCS value.

4.8 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/prand.h File Reference

Pseudo-random number generator interface.

```
#include "wsf_types.h"
```

Include dependency graph for prand.h:



Functions

- void PrandInit (void)

  Initialize random number generator.
- void PrandGen (uint8_t ∗pBuf, uint16_t len)

  Generate random data.

4.8.1 Detailed Description

Pseudo-random number generator interface.

Copyright (c) 2016-2018 Arm Ltd. All Rights Reserved. ARM confidential and proprietary.

Copyright (c) 2019-2020 Packetcraft, Inc.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

```
http://www.apache.org/licenses/LICENSE-2.0
```

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

4.8.2 Function Documentation

4.8.2.1 PrandGen()

```
void PrandGen (
             uint8_t * pBuf,
             uint16_t len )
```

Generate random data.

Parameters

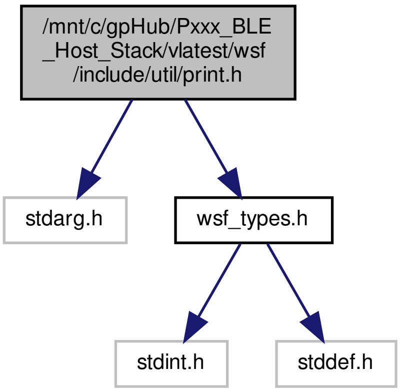
pBuf	Storage for random data.
len	Length of data to generate, in bytes.

4.9 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/print.h File Reference

Print functions.

```
#include <stdarg.h>
#include "wsf_types.h"
```

Include dependency graph for print.h:



Macros

- #define PRINT_ATTRIBUTE(a, b)

  Print function attributes.

Functions

- uint32_t PrintVsn (char ∗pStr, uint32_t size, const char ∗pFmt, va_list ap) PRINT_ATTRIBUTE(3

  Print a trace message.

4.9.1 Detailed Description

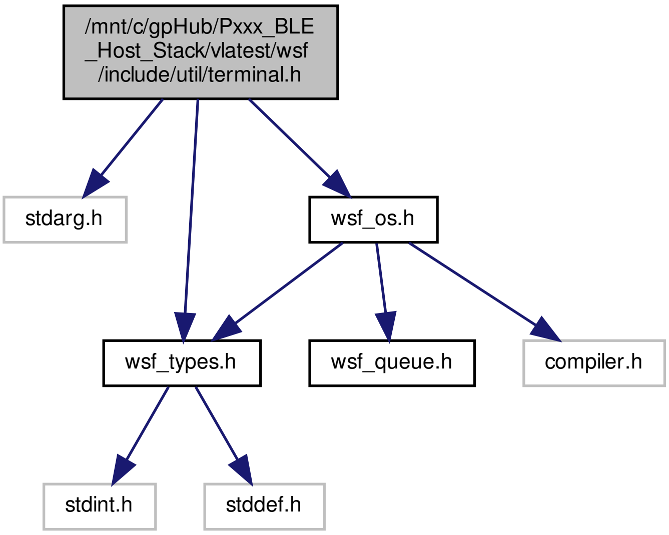
Print functions.

4.10 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/terminal.h File Reference

Terminal handler.

```
#include <stdarg.h>
#include "wsf_types.h"
#include "wsf_os.h"
```

Include dependency graph for terminal.h:


/mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/terminal.h
stdarg.h
wsf_os.h
wsf_types.h
wsf_queue.h
compiler.h
stdint.h
stddef.h

Data Structures

- struct terminalCommand_t

  Terminal command.

Macros

- #define TERMINAL_MAX_ARGC 8u

  Maximum number of arguments to any command.
- #define TERMINAL_MAX_COMMAND_LEN 100u

  Maximum length of command line.
- #define TERMINAL_PRINTF_MAX_LEN 256u

  Maximum length of any printed output.
- #define TERMINAL_STRING_PROMPT "> "

  Prompt string.
- #define TERMINAL_STRING_ERROR "ERROR: "

  Error prefix.
- #define TERMINAL_STRING_USAGE "USAGE: "

  Usage prefix.
- #define TERMINAL_STRING_NEW_LINE "\r\n"

  New line string.

Typedefs

- typedef uint8_t(∗ terminalHandler_t) (uint32_t argc, char ∗∗argv)

  Handler for a terminal command.
- typedef bool_t(∗ terminalUartTx_t) (const uint8_t ∗pBuf, uint32_t len)

  Handler for transmit.

Enumerations

- enum {
  TERMINAL_ERROR_OK = 0,
  TERMINAL_ERROR_BAD_ARGUMENTS = 1,
  TERMINAL_ERROR_TOO_FEW_ARGUMENTS = 2,
  TERMINAL_ERROR_TOO_MANY_ARGUMENTS = 3,
  TERMINAL_ERROR_EXEC = 4 }

  Terminal command error codes.

Functions

- void TerminalInit (wsfHandlerId_t handlerId)

  Initialize terminal.
- void TerminalRegisterUartTxFunc (terminalUartTx_t uartTxFunc)

  Register the UART Tx Function for the platform.
- void TerminalRegisterCommand (terminalCommand_t ∗pCommand)

  Register command with terminal.
- void TerminalHandler (wsfEventMask_t event, wsfMsgHdr_t ∗pMsg)

  Handler for terminal messages.

- void TerminalRx (uint8_t dataByte)

  Called by application when a data byte is received.
- void TerminalTxStr (const char ∗pStr)

  Called by application to transmit string.
- void TerminalTxChar (char c)

  Called by application to transmit character.
- void TerminalTxPrint (const char ∗pStr,...)

  Called by application to print formatted data.
- void TerminalTx (const uint8_t ∗pData, uint16_t len)

  Application function to transmit data..

4.10.1 Detailed Description

Terminal handler.

4.11 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/wstr.h File Reference

String manipulation functions.

Macros

- #define WSTR_IS_HEX_FORMAT(c)
- #define WSTR_IS_BIN_FORMAT(c) (((char)(c)[0] == '0') && ((char)(c)[1] == 'b'))

Functions

- void WstrnCpy (char ∗pBuf, const char ∗pData, uint8_t n)

  Copies a string up to a given length.
- void WStrReverseCpy (uint8_t ∗pBuf1, const uint8_t ∗pBuf2, uint16_t len)

  Byte by byte reverse and copy a buffer.
- void WStrReverse (uint8_t ∗pBuf, uint8_t len)

  Byte by byte reverse a buffer.
- void WStrFormatHex (char ∗pBuf, uint32_t val, uint8_t len)

  Format a hex value.
- void WStrHexToArray (const char ∗pStr, uint8_t ∗pBuf, uint16_t len)

  Convert a formatted string to int array, zero out space after the string length.

4.11.1 Detailed Description

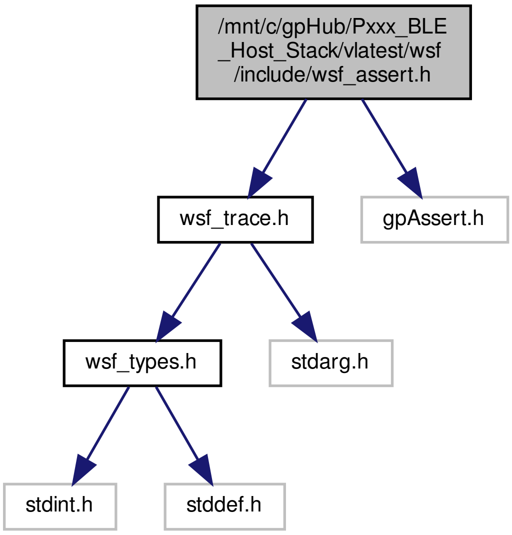
String manipulation functions.

4.12 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_assert.h File Reference

Assert macro.

```
#include "wsf_trace.h"
#include "gpAssert.h"
```

Include dependency graph for wsf_assert.h:



Macros

- #define **GP_COMPONENT_ID** GP_COMPONENT_ID_QORVOBLEHOST
- #define WSF_ASSERT_ENABLED FALSE

  Enable assertion statements.
- #define WSF_ASSERT(expr) (void)(expr);

  Run-time assert macro. The assert executes when the expression is FALSE.
- #define WSF_CT_ASSERT(expr) extern char wsf_ct_assert[(expr) ? 1 : -1]

  Compile-time assert macro. This macro causes a compiler error when the expression is FALSE. Note that this macro is generally used at file scope to test constant expressions. Errors may result of it is used in executing code.

Functions

- void **WsfAssert** (const char ∗pFile, uint16_t line)
- uint16_t WsfAssertNum (void)

  Get number of asserts.
- void WsfAssertTrapEnable (bool_t enaAssertTrap)

  Enable assert trap.
- void WsfAssertRegister (void(∗cback)(void))

  Register assert handler.

4.12.1 Detailed Description

Assert macro.

4.13 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_buf.h File Reference

Buffer pool service.

Data Structures

- struct wsfBufPoolDesc_t

  Buffer pool descriptor structure.
- struct WsfBufPoolStat_t

  Pool statistics.
- struct wsfBufDiagAllocFail_t

  WSF buffer diagnostics - buffer allocation failure.
- struct WsfBufDiag_t

  WSF buffer diagnostics message.

Macros

- #define WSF_BUF_FREE_CHECK_ASSERT TRUE

  Check if trying to free a buffer that is already free.
- #define WSF_BUF_ALLOC_BEST_FIT_FAIL_ASSERT FALSE

  Assert on best-fit buffer allocation failure.
- #define WSF_BUF_ALLOC_FAIL_ASSERT FALSE

  Assert on buffer allocation failure.
- #define WSF_BUF_STATS_HIST FALSE

  Buffer histogram stats.
- #define WSF_BUF_STATS_MAX_LEN 128

  Length of the buffer statistics array.
- #define WSF_BUF_STATS_MAX_POOL 32

  Max number of pools can allocate.
- #define WSF_BUF_ALLOC_FAILED 1

  Failure Codes.
- #define WSF_BUF_STATS FALSE

  Enable buffer allocation statistics.

Typedefs

- typedef void(∗ WsfBufDiagCback_t) (WsfBufDiag_t ∗pInfo)

  Callback providing WSF buffer diagnostic messages.

Functions

- uint32_t WsfBufCalcSize (uint8_t numPools, wsfBufPoolDesc_t ∗pDesc)

  Calculate size required by the buffer pool.
- uint32_t WsfBufInit (uint32_t bufMemLen, uint8_t ∗pBufMem, uint8_t numPools, wsfBufPoolDesc_t ∗pDesc)

  Initialize the buffer pool service. This function should only be called once upon system initialization.
- bool_t CheckWsfBufAlloc (uint16_t len)

  Verify whether a buffer with required length is available.
- void ∗ WsfBufAlloc (uint16_t len)

  Allocate a buffer.
- void WsfBufFree (void ∗pBuf)

  Free a buffer.
- uint8_t ∗ WsfBufGetAllocStats (void)

  Diagnostic function to get the buffer allocation statistics.
- uint8_t ∗ WsfBufGetPoolOverFlowStats (void)

  Diagnostic function to get the number of overflow times for each pool.
- uint8_t WsfBufGetNumPool (void)

  Get number of pools.
- void WsfBufGetPoolStats (WsfBufPoolStat_t ∗pStat, uint8_t numPool)

  Get statistics for each pool.
- void WsfBufDiagRegister (WsfBufDiagCback_t callback)

  Called to register the buffer diagnostics callback function.
- uint32_t WsfBufNumOutstanding (void)

  Get the number of outstanding memory pool buffers.

4.13.1 Detailed Description

Buffer pool service.

4.14 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_bufio.h File Reference

Buffer I/O service.

Typedefs

- typedef void(∗ WsfBufIoUartRxCback_t) (uint8_t rxByte)

  Buffer IO UART Rx callback.

Functions

- uint32_t WsfBufIoUartInit (void ∗pBuf, uint32_t size)

  Initialize the platform UART.
- void WsfBufIoUartRegister (WsfBufIoUartRxCback_t rxCback)

  Register the platform UART RX callback.
- bool_t WsfBufIoWrite (const uint8_t ∗pBuf, uint32_t len)

  Transmit buffer on platform UART.

4.14.1 Detailed Description

Buffer I/O service.

4.15 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_cs.h File Reference

Critical section macros.

Macros

- #define WSF_CS_STATS FALSE

  Use CS statistics hooks.
- #define WSF_CS_INIT(cs)

  Initialize critical section. This macro may define a variable.
- #define WSF_CS_ENTER(cs) WsfCsEnter()

  Enter a critical section.
- #define WSF_CS_EXIT(cs) WsfCsExit()

  Exit a critical section.

Functions

- uint32_t WsfCsStatsGetCsWaterMark (void)

  Get critical section duration watermark level.
- void WsfCsEnter (void)

  Enter a critical section.
- void WsfCsExit (void)

  Exit a critical section.

4.15.1 Detailed Description

Critical section macros.

4.16 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_detoken.h File Reference

Token trace decode header file.

Macros

- #define WSF_DETOKEN_VS_SET_EVENT_MASK_OPCODE HCI_OPCODE(HCI_OGF_VENDOR_SPEC, 0x3E1)

  Vendor specific event mask opcode.
- #define WSF_DETOKEN_ENABLE_BIT (1<<1)

  Event mask bits.
- #define WSF_DETOKEN_VS_EVT_TOKEN 0xFFF0

  Vendor specific token event.

Platform identifiers

- #define **WSF_DETOKEN_TRACE_NORDIC** 1

Parameter mask bits

- #define **WSF_DETOKEN_PARAM_VARIABLE** 0
- #define **WSF_DETOKEN_PARAM_STRING** 1

Functions

- void WsfDetokenInit (void)

  Initialize detoken trace.
- void WsfDetokenEnable (bool_t enable)

  Enable/disable detoken trace.
- bool_t WsfDetokenProcessHciEvent (uint16_t len, uint8_t ∗pBuffer)

  Process vendor specific HCI events and decode token trace events from the LL.

4.16.1 Detailed Description

Token trace decode header file.

4.16.2 Function Documentation

4.16.2.1 WsfDetokenEnable()

```
void WsfDetokenEnable (
            bool_t enable )
```

Enable/disable detoken trace.

Parameters

enable	TRUE to enable, FALSE to disable.

4.16.2.2 WsfDetokenProcessHciEvent()

```
bool_t WsfDetokenProcessHciEvent (
            uint16_t len,
            uint8_t * pBuffer )
```

Process vendor specific HCI events and decode token trace events from the LL.

Parameters

len	Length of pBuffer in bytes.
pBuffer	Buffer containing HCI event.

Returns

TRUE if VS HCI message is a token, else FALSE.

4.17 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_efs.h File Reference

Embedded File System service.

Data Structures

- struct wsfEsfAttributes_t

  File attributes data type.
- struct wsfEfsControl_t

  File control block data type.
- struct wsfEfsFileInfo_t

  File Listing Information.
- struct wsfEfsMedia_t

  Media Control data type.

Macros

- #define WSF_EFS_MAX_FILES 6

  Max Number of Files and Media.
- #define WSF_EFS_MAX_MEDIA 4

  Max Number of Media.
- #define WSF_EFS_FILE_OFFSET_ANY 0xFFFFFFFF

  Offset to WsfEfsAddFile indicating any file offset can be used.

- #define WSF_EFS_NAME_LEN 16

  File name length in bytes.
- #define WSF_EFS_VERSION_LEN 16

  File version length in bytes.
- #define WSF_EFS_USER_CMD 0x80

  Media Specific Command Identifiers reserved for applications begin at 0x80.

Status Codes

- #define WSF_EFS_SUCCESS 0

  Success.
- #define WSF_EFS_FAILURE 1

  Failure.
- #define WSF_EFS_CBACK_REQUIRED 2

  File sytem callback required.
- #define WSF_EFS_GET_FAILED 0xFFFF

  Get operation failure.
- #define WSF_EFS_PUT_FAILED 0xFFFF

  PUT operation failure.

Invalid Parameter Identifiers

- #define WSF_EFS_INVALID_HANDLE 0xFFFF

  Invalid Handle.
- #define WSF_EFS_INVALID_OFFSET 0xFFFFFFFF

  Invalid Offset.
- #define WSF_EFS_INVALID_SIZE 0xFFFFFFFF

  Invalid Size.
- #define WSF_EFS_INVALID_MEDIA 0xFF

  Invalid Media.

File Types

- #define WSF_EFS_FILE_TYPE_BULK 0

  Bulk File Type.
- #define WSF_EFS_FILE_TYPE_STREAM 1

  Stream File Type.

File Permissions

- #define WSF_EFS_REMOTE_PERMISSIONS_MASK 0xFF

  Remote Permissions.
- #define WSF_EFS_REMOTE_GET_PERMITTED 0x01

  Remote Get Permitted.
- #define WSF_EFS_REMOTE_PUT_PERMITTED 0x02

  Remote Put Permitted.
- #define WSF_EFS_REMOTE_ERASE_PERMITTED 0x04

  Remote Erase Permitted.
- #define WSF_EFS_REMOTE_VERIFY_PERMITTED 0x08

  Remote Verify Permitted.
- #define WSF_EFS_LOCAL_GET_PERMITTED 0x0100

  Local Get Permitted.
- #define WSF_EFS_LOCAL_PUT_PERMITTED 0x0200

  Local Put Permitted.
- #define WSF_EFS_LOCAL_ERASE_PERMITTED 0x0400

  Local Erase Permitted.

- #define WSF_EFS_REMOTE_VISIBLE 0x0800
  File Visible via Remote WDXS.

Standard Media Specific Command Identifiers

- #define WSF_EFS_WDXS_PUT_COMPLETE_CMD 0x00
  Put Complete.
- #define WSF_EFS_VALIDATE_CMD 0x01
  Validate Req for the file.
- #define WSF_EFS_RESTORE_ATTRS_CMD 0x02
  Restore file attributes.
- #define WSF_EFS_GET_FS_CBACK_MODE_CMD 0x03
  Filesystem callback check.
- #define WSF_EFS_CFG_FOTA_CMD 0x04
  Configure FOTA and reboot.

Typedefs

- typedef uint16_t wsfEfsHandle_t
  File handle data type.
- typedef uint8_t wsfMediaInitFunc_t(void)
  Media Init function, called when media is registered.
- typedef uint8_t wsfMediaEraseFunc_t(uint32_t address, uint32_t size)
  Media Erase function.
- typedef uint8_t wsfMediaReadFunc_t(uint8_t ∗pBuf, uint32_t address, uint32_t size)
  Media Read function.
- typedef uint8_t wsfMediaWriteFunc_t(const uint8_t ∗pBuf, uint32_t address, uint32_t size)
  Media Write function.
- typedef uint8_t wsfMediaHandleCmdFunc_t(uint8_t cmd, uint32_t param)
  Media Specific Command handler.
- typedef const wsfEfsMedia_t ∗ pWsfEfsMedia_t
  Pointer to Media Control data type.

Functions

- void WsfEfsInit (void)
  Initialise the embedded file system.
- wsfEfsHandle_t WsfEfsAddFile (uint32_t maxSize, uint8_t media, wsfEsfAttributes_t ∗pAttr, uint32_t offset)
  Create a file in the embedded file system.
- uint8_t WsfEfsRemoveFile (wsfEfsHandle_t handle)
  Deletes a file in the embedded file system.
- uint8_t WsfEfsErase (wsfEfsHandle_t handle)
  Clears the contents of a file without deleting the file.
- uint8_t WsfEfsGetAttributes (wsfEfsHandle_t handle, wsfEsfAttributes_t ∗pAttr)
  Gets the attributes of a file.
- uint8_t WsfEfsSetAttributes (wsfEfsHandle_t handle, wsfEsfAttributes_t ∗pInfo)
  Updates the attributes of a file.
- uint16_t WsfEfsGet (wsfEfsHandle_t handle, uint32_t offset, uint8_t ∗pBuffer, uint16_t len)
  Copies data from a file.
- uint16_t WsfEfsPut (wsfEfsHandle_t handle, uint32_t offset, const uint8_t ∗pBuffer, uint16_t len)
  Writes data to a file.

- uint8_t WsfEfsRegisterMedia (const wsfEfsMedia_t ∗pMediaCtrl, uint8_t mediaID)

  Registers a File Storage Medium with the Embedded File System.
- wsfEfsControl_t ∗ WsfEfsGetFileByHandle (wsfEfsHandle_t handle)

  Returns the file control block for the given handle.
- char ∗ WsfEfsGetFileName (wsfEfsHandle_t handle)

  Get the name of a file.
- char ∗ WsfEfsGetFileVersion (wsfEfsHandle_t handle)

  Get the version of a file.
- uint32_t WsfEfsGetFileSize (wsfEfsHandle_t handle)

  Get the size of a file.
- uint32_t WsfEfsGetFileMaxSize (wsfEfsHandle_t handle)

  Get the number of bytes of memory reserved for use by a file.
- uint8_t WsfEfsGetFileType (wsfEfsHandle_t handle)

  Get the type of a file.
- uint16_t WsfEfsGetFilePermissions (wsfEfsHandle_t handle)

  Get the permissions of a file.
- uint8_t WsfEfsMediaSpecificCommand (wsfEfsHandle_t handle, uint8_t cmd, uint32_t param)

  Execute a media specific command on a file.

4.17.1 Detailed Description

Embedded File System service.

4.18 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_heap.h File Reference

Buffer heap service.

Functions

- uint32_t WsfHeapCountAvailable (void)

  Get heap available.
- uint32_t WsfHeapCountUsed (void)

  Get heap used.
- void WsfHeapAlloc (uint32_t size)

  Reserve heap memory.
- void ∗ WsfHeapGetFreeStartAddress (void)

  Get next available heap memory.

4.18.1 Detailed Description

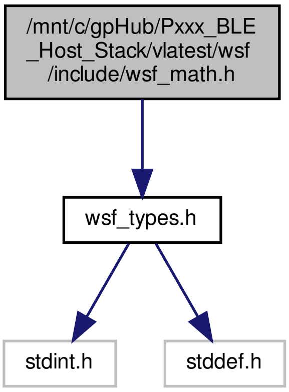
Buffer heap service.

4.19 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_math.h File Reference

Common math utilities.

```
#include "wsf_types.h"
```

Include dependency graph for wsf_math.h:



Macros

- #define WSF_MIN(a, b) ((a) < (b) ? (a) : (b))

  Returns the minimum of two values.
- #define WSF_MAX(a, b) ((a) > (b) ? (a) : (b))

  Returns the maximum of two values.

4.19.1 Detailed Description

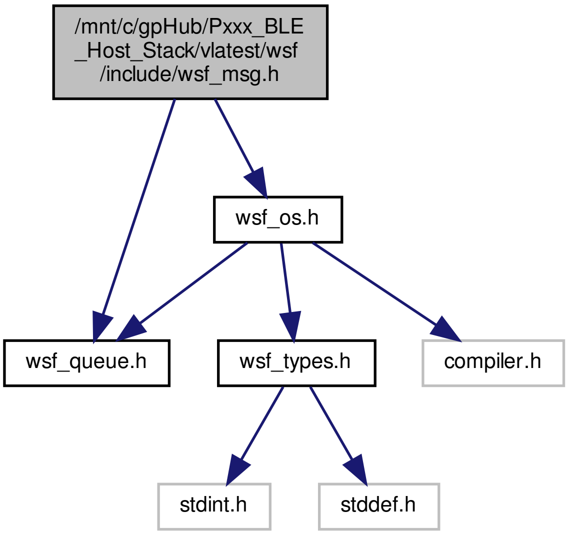
Common math utilities.

4.20 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_msg.h File Reference

Message passing service.

```
#include "wsf_queue.h"
#include "wsf_os.h"
```

Include dependency graph for wsf_msg.h:



Functions

- bool_t CheckWsfMsgDataAlloc (uint16_t len, uint8_t tailroom)

  Verify whether a data buffer with required length is available to send a message buffer with WsfMsgSend().
- void ∗ WsfMsgDataAlloc (uint16_t len, uint8_t tailroom)

  Allocate a data message buffer to be sent with WsfMsgSend().
- bool_t CheckWsfMsgAlloc (uint16_t len)

  Verify whether a buffer with required length is available to send a message buffer with WsfMsgSend().
- void ∗ WsfMsgAlloc (uint16_t len)

  Allocate a message buffer to be sent with WsfMsgSend().
- void WsfMsgFree (void ∗pMsg)

  Free a message buffer allocated with WsfMsgAlloc().
- void WsfMsgSend (wsfHandlerId_t handlerId, void ∗pMsg)

  Send a message to an event handler.
- void WsfMsgEnq (wsfQueue_t ∗pQueue, wsfHandlerId_t handlerId, void ∗pMsg)

  Enqueue a message.
- void ∗ WsfMsgDeq (wsfQueue_t ∗pQueue, wsfHandlerId_t ∗pHandlerId)

  Dequeue a message.
- void ∗ WsfMsgPeek (wsfQueue_t ∗pQueue, wsfHandlerId_t ∗pHandlerId)

  Get the next message without removing it from the queue.
- void ∗ WsfMsgNPeek (wsfQueue_t ∗pQueue, uint8_t n, wsfHandlerId_t ∗pHandlerId)

  Get the Nth message without removing it from the queue.

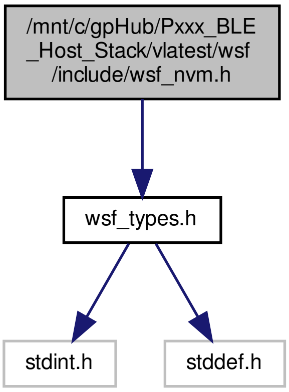
4.20.1 Detailed Description

Message passing service.

4.21 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_nvm.h File Reference

NVM service.

`#include "wsf_types.h"`
Include dependency graph for wsf_nvm.h:



Typedefs

- typedef void(∗ WsfNvmCompEvent_t) (bool_t status)

  Operation completion callback.

Functions

- static uint64_t WsfNvmConvertChar8to64Bit (char ∗charId)

  Read data.
- void WsfNvmInit (void)

  Initialize the WSF NVM.
- bool_t WsfNvmReadData (uint64_t id, uint8_t ∗pData, uint16_t len, WsfNvmCompEvent_t compCback)

  Read data.
- bool_t WsfNvmWriteData (uint64_t id, const uint8_t ∗pData, uint16_t len, WsfNvmCompEvent_t compCback)

  Write data.
- bool_t WsfNvmEraseData (uint64_t id, WsfNvmCompEvent_t compCback)

  Erase data.
- void WsfNvmEraseDataAll (WsfNvmCompEvent_t compCback)

  Erase all data located in NVM storage.

4.21.1 Detailed Description

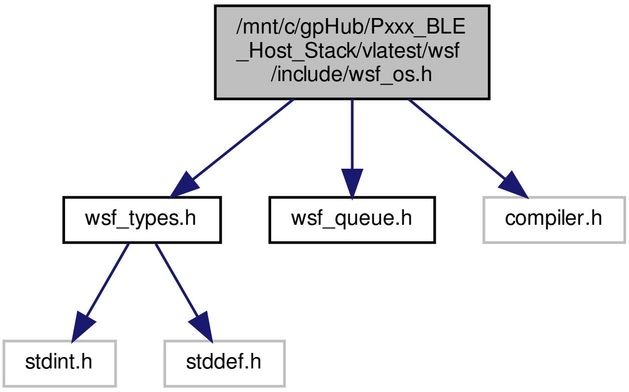
NVM service.

4.22 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_os.h File Reference

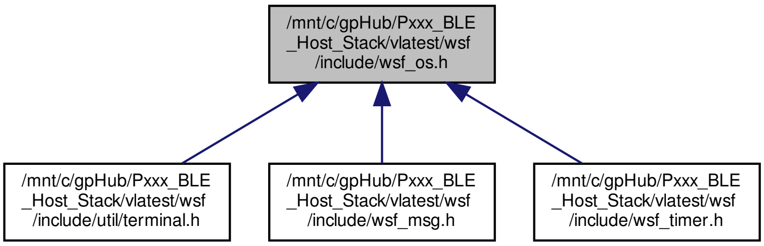
Software foundation OS API.

```
#include "wsf_types.h"
#include "wsf_queue.h"
#include "compiler.h"
```

Include dependency graph for wsf_os.h:


/mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_os.h
wsf_types.h
wsf_queue.h
compiler.h
stdint.h
stddef.h

This graph shows which files directly or indirectly include this file:


/mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_os.h
/mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/util/terminal.h
/mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_msg.h
/mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_timer.h

Data Structures

- struct wsfMsgHdr_t

  Common message structure passed to event handler.

Macros

- #define WSF_OS_DIAG FALSE

  OS Diagnostics.
- #define WSF_TASK_FROM_ID(handlerID) (((handlerID) >> 4) & 0x0F)

  Derive task from handler ID.
- #define WSF_HANDLER_FROM_ID(handlerID) ((handlerID) & 0x0F)

  Derive handler from handler ID.
- #define WSF_INVALID_TASK_ID 0xFF

  Invalid Task Identifier.
- #define WSF_OS_GET_ACTIVE_HANDLER_ID() WSF_INVALID_TASK_ID

  Get Diagnostic Task Identifier.

WSF Task Events

- #define WSF_MSG_QUEUE_EVENT 0x01

  Message queued for event handler.
- #define WSF_TIMER_EVENT 0x02

  Timer expired for event handler.
- #define WSF_HANDLER_EVENT 0x04

  Event set for event handler.

Typedefs

- typedef uint8_t wsfHandlerId_t

  Event handler ID data type.
- typedef uint16_t wsfEventMask_t

  Event handler event mask data type.
- typedef wsfHandlerId_t wsfTaskId_t

Task ID data type.
- typedef uint8_t wsfTaskEvent_t

  Task event mask data type.
- typedef bool_t(∗ WsfOsIdleHandler_t) (void)

  Idle check function.
- typedef void(∗ wsfEventHandler_t) (wsfEventMask_t event, wsfMsgHdr_t ∗pMsg)

  Event handler callback function.

Functions

- void WsfSetEvent (wsfHandlerId_t handlerId, wsfEventMask_t event)

  Set an event for an event handler.
- void WsfTaskLock (void)

  Lock task scheduling.
- void WsfTaskUnlock (void)

  Unlock task scheduling.
- void WsfTaskSetReady (wsfHandlerId_t handlerId, wsfTaskEvent_t event)

  Set the task used by the given handler as ready to run.
- wsfQueue_t ∗ WsfTaskMsgQueue (wsfHandlerId_t handlerId)

  Return the task message queue used by the given handler.
- wsfHandlerId_t WsfOsSetNextHandler (wsfEventHandler_t handler)

  Set the next WSF handler function in the WSF OS handler array. This function should only be called as part of the OS initialization procedure.
- void WsfOsInit (void)

  Initialize OS control structure.
- bool_t WsfOsReadyToSleep (void)

  Check if WSF is ready to sleep.
- void WsfOsDispatcher (void)

  Event dispatched. Designed to be called repeatedly from infinite loop.
- void WsfOsEnterMainLoop (void)

  OS starts main loop.
- void WsfOsRegisterIdleTask (WsfOsIdleHandler_t func)

  Register service check functions.

Variables

- wsfHandlerId_t WsfActiveHandler

  Diagnostic Task Identifier.

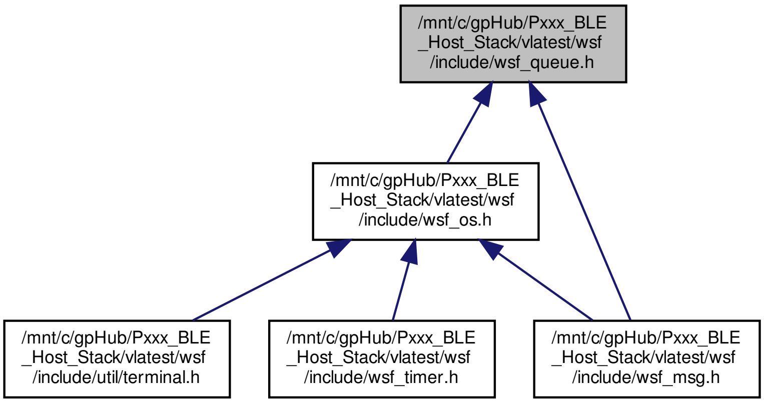
4.22.1 Detailed Description

Software foundation OS API.

4.23 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_queue.h File Reference

General purpose queue service.

This graph shows which files directly or indirectly include this file:


/mnt/c/gpHub/Pxxx_BLE _Host_Stack/vlatest/wsf /include/wsf_queue.h
/mnt/c/gpHub/Pxxx_BLE _Host_Stack/vlatest/wsf /include/wsf_os.h
/mnt/c/gpHub/Pxxx_BLE _Host_Stack/vlatest/wsf /include/util/terminal.h
/mnt/c/gpHub/Pxxx_BLE _Host_Stack/vlatest/wsf /include/wsf_timer.h
/mnt/c/gpHub/Pxxx_BLE _Host_Stack/vlatest/wsf /include/wsf_msg.h

Data Structures

- struct wsfQueue_t

  Queue structure.

Macros

- #define WSF_QUEUE_INIT(pQueue) {(pQueue)->pHead = NULL; (pQueue)->pTail = NULL;}

  Initialize a queue.

Functions

- void WsfQueueEnq (wsfQueue_t ∗pQueue, void ∗pElem)

  Enqueue an element to the tail of a queue.
- void ∗ WsfQueueDeq (wsfQueue_t ∗pQueue)

  Dequeue an element from the head of a queue.
- void WsfQueuePush (wsfQueue_t ∗pQueue, void ∗pElem)

  Push an element to the head of a queue.
- void WsfQueueInsert (wsfQueue_t ∗pQueue, void ∗pElem, void ∗pPrev)

  Insert an element into a queue. This function is typically used when iterating over a queue.
- void WsfQueueRemove (wsfQueue_t ∗pQueue, void ∗pElem, void ∗pPrev)

  Remove an element from a queue. This function is typically used when iterating over a queue.
- uint16_t WsfQueueCount (wsfQueue_t ∗pQueue)

  Count the number of elements in a queue.
- bool_t WsfQueueEmpty (wsfQueue_t ∗pQueue)

  Return TRUE if queue is empty.
- bool_t WsfIsQueueDepthOne (wsfQueue_t ∗pQueue)

  Check for a queue depth of 1 element.

4.23.1 Detailed Description

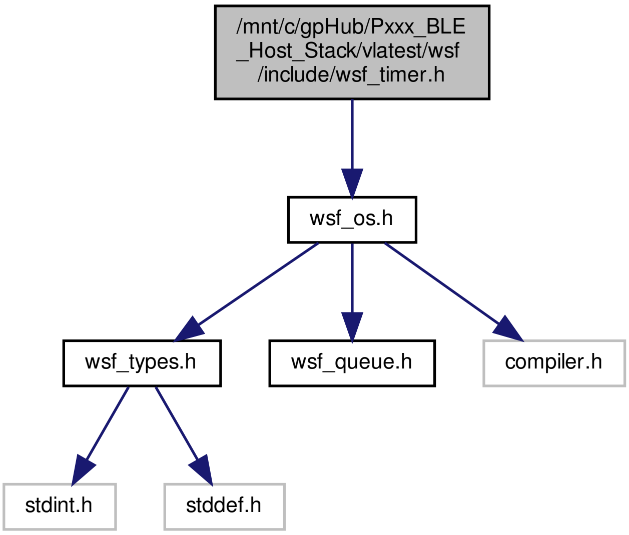
General purpose queue service.

4.24 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_timer.h File Reference

Timer service.

`#include "wsf_os.h"`

Include dependency graph for wsf_timer.h:



Data Structures

- struct wsfTimer_t

  Timer structure.

Macros

- #define WSF_MS_PER_TICK 1

  Default milliseconds per tick rate.

Typedefs

- typedef uint32_t wsfTimerTicks_t

  Timer ticks data type.

Functions

- void WsfTimerInit (void)

  Initialize the timer service. This function should only be called once upon system initialization.
- void WsfTimerStartSec (wsfTimer_t ∗pTimer, wsfTimerTicks_t sec)

  Start a timer in units of seconds. Before this function is called parameter pTimer->handlerId must be set to the event handler for this timer and parameter pTimer->msg must be set to any application-defined timer event parameters.
- void WsfTimerStartMs (wsfTimer_t ∗pTimer, wsfTimerTicks_t ms)

  Start a timer in units of milliseconds.
- void WsfTimerStop (wsfTimer_t ∗pTimer)

  Stop a timer.
- void WsfTimerUpdate (wsfTimerTicks_t ticks)

  Update the timer service with the number of elapsed ticks. This function is typically called only from timer porting code.
- wsfTimerTicks_t WsfTimerNextExpiration (bool_t ∗pTimerRunning)

  Return the number of ticks until the next timer expiration. Note that this function can return zero even if a timer is running, indicating a timer has expired but has not yet been serviced.
- wsfTimer_t ∗ WsfTimerServiceExpired (wsfTaskId_t taskId)

  Service expired timers for the given task. This function is typically called only WSF OS porting code.

4.24.1 Detailed Description

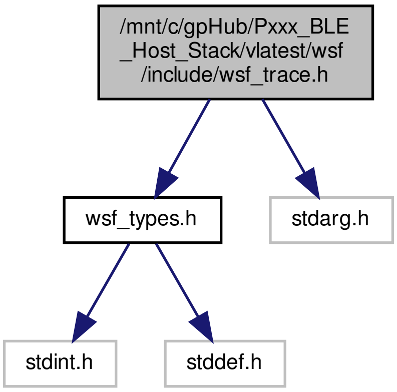
Timer service.

4.25 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_trace.h File Reference

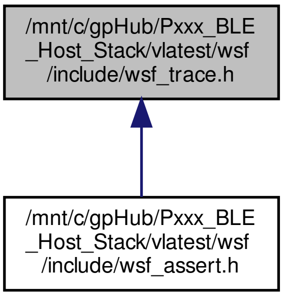
Trace message interface.

```
#include "wsf_types.h"
#include <stdarg.h>
```

Include dependency graph for wsf_trace.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define WSF_TRACE_ENABLED FALSE

  Trace enable flag (default is disabled, override with compile-time directive).
- #define WSF_TOKEN_ENABLED FALSE

  Tokenized tracing enable flag (default is disabled, override with compile-time directive).

- #define LL_TRACE_ENABLED FALSE

  Trace enabled for controller.
- #define **AUD_TRACE_ENABLED** FALSE
- #define WSF_TRACE_INFO0(msg)

  0 argument WSF info trace.
- #define WSF_TRACE_INFO1(msg, var1)

  1 argument WSF info trace.
- #define WSF_TRACE_INFO2(msg, var1, var2)

  2 argument WSF info trace.
- #define WSF_TRACE_INFO3(msg, var1, var2, var3)

  3 argument WSF info trace.
- #define WSF_TRACE_INFO4(msg, var1, var2, var3, var4)

  4 argument WSF info trace.
- #define WSF_TRACE_INFO5(msg, var1, var2, var3, var4, var5)

  5 argument WSF info trace.
- #define WSF_TRACE_INFO6(msg, var1, var2, var3, var4, var5, var6)

  6 argument WSF info trace.
- #define WSF_TRACE_WARN0(msg) WSF_TRACE0("WSF", "WARN", msg)

  0 argument WSF warning trace.
- #define WSF_TRACE_WARN1(msg, var1) WSF_TRACE1("WSF", "WARN", msg, var1)

  1 argument WSF warning trace.
- #define WSF_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("WSF", "WARN", msg, var1, var2)

  2 argument WSF warning trace.
- #define WSF_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("WSF", "WARN", msg, var1, var2, var3)

  3 argument WSF warning trace.
- #define WSF_TRACE_WARN4(msg, var1, var2, var3, var4) WSF_TRACE4("WSF", "WARN", msg, var1, var2, var3, var4)

  5 argument WSF warning trace.
- #define WSF_TRACE_WARN5(msg, var1, var2, var3, var4, var5) WSF_TRACE5("WSF", "WARN", msg, var1, var2, var3, var4, var5)

  5 argument WSF warning trace.
- #define WSF_TRACE_WARN6(msg, var1, var2, var3, var4, var5, var6) WSF_TRACE6("WSF", "WARN", msg, var1, var2, var3, var4, var5, var6)

  6 argument WSF warning trace.
- #define WSF_TRACE_ERR0(msg) WSF_TRACE0("WSF", "ERR", msg)

  0 argument WSF error trace.
- #define WSF_TRACE_ERR1(msg, var1) WSF_TRACE1("WSF", "ERR", msg, var1)

  1 argument WSF error trace.
- #define WSF_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("WSF", "ERR", msg, var1, var2)

  2 argument WSF error trace.
- #define WSF_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("WSF", "ERR", msg, var1, var2, var3)

  3 argument WSF error trace.
- #define WSF_TRACE_ERR4(msg, var1, var2, var3, var4) WSF_TRACE4("WSF", "ERR", msg, var1, var2, var3, var4)

  4 argument WSF error trace.
- #define WSF_TRACE_ERR5(msg, var1, var2, var3, var4, var5) WSF_TRACE5("WSF", "ERR", msg, var1, var2, var3, var4, var5)

  5 argument WSF error trace.
- #define WSF_TRACE_ERR6(msg, var1, var2, var3, var4, var5, var6) WSF_TRACE6("WSF", "ERR", msg, var1, var2, var3, var4, var5, var6)

  5 argument WSF error trace.

- #define WSF_TRACE_ALLOC0(msg)

  0 argument WSF buffer allocation trace.
- #define WSF_TRACE_ALLOC1(msg, var1)

  1 argument WSF buffer allocation trace.
- #define WSF_TRACE_ALLOC2(msg, var1, var2)

  2 argument WSF buffer allocation trace.
- #define WSF_TRACE_ALLOC3(msg, var1, var2, var3)

  3 argument WSF buffer allocation trace.
- #define WSF_TRACE_FREE0(msg)

  0 argument WSF buffer free trace.
- #define WSF_TRACE_FREE1(msg, var1)

  1 argument WSF buffer free trace.
- #define WSF_TRACE_FREE2(msg, var1, var2)

  2 argument WSF buffer free trace.
- #define WSF_TRACE_FREE3(msg, var1, var2, var3)

  3 argument WSF buffer free trace.
- #define WSF_TRACE_MSG0(msg)

  0 argument WSF message trace.
- #define WSF_TRACE_MSG1(msg, var1)

  1 argument WSF message trace.
- #define WSF_TRACE_MSG2(msg, var1, var2)

  2 argument WSF message trace.
- #define WSF_TRACE_MSG3(msg, var1, var2, var3)

  3 argument WSF message trace.
- #define HCI_TRACE_INFO0(msg)

  0 argument HCI info trace.
- #define HCI_TRACE_INFO1(msg, var1)

  1 argument HCI info trace.
- #define HCI_TRACE_INFO2(msg, var1, var2)

  2 argument HCI info trace.
- #define HCI_TRACE_INFO3(msg, var1, var2, var3)

  3 argument HCI info trace.
- #define HCI_TRACE_WARN0(msg) WSF_TRACE0("HCI", "WARN", msg)

  0 argument HCI warning trace.
- #define HCI_TRACE_WARN1(msg, var1) WSF_TRACE1("HCI", "WARN", msg, var1)

  1 argument HCI warning trace.
- #define HCI_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("HCI", "WARN", msg, var1, var2)

  2 argument HCI warning trace.
- #define HCI_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("HCI", "WARN", msg, var1, var2, var3)

  3 argument HCI warning trace.
- #define HCI_TRACE_ERR0(msg) WSF_TRACE0("HCI", "ERR", msg)

  0 argument HCI error trace.
- #define HCI_TRACE_ERR1(msg, var1) WSF_TRACE1("HCI", "ERR", msg, var1)

  1 argument HCI error trace.
- #define HCI_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("HCI", "ERR", msg, var1, var2)

  2 argument HCI error trace.
- #define HCI_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("HCI", "ERR", msg, var1, var2, var3)

  3 argument HCI error trace.
- #define HCI_PDUMP_CMD(len, pBuf)

  HCI PDUMP on command.
- #define HCI_PDUMP_EVT(len, pBuf)

HCI PDUMP on event.

- #define HCI_PDUMP_TX_ACL(len, pBuf)

  HCI PDUMP on transmitted ACL message.
- #define HCI_PDUMP_RX_ACL(len, pBuf)

  HCI PDUMP on Received ACL message.
- #define HCI_PDUMP_TX_ISO(len, pBuf)

  HCI PDUMP on transmitted ISO message.
- #define HCI_PDUMP_RX_ISO(len, pBuf)

  HCI PDUMP on Received ISO message.
- #define DM_TRACE_INFO0(msg)

  0 argument DM info trace.
- #define DM_TRACE_INFO1(msg, var1)

  1 argument DM info trace.
- #define DM_TRACE_INFO2(msg, var1, var2)

  2 argument DM info trace.
- #define DM_TRACE_INFO3(msg, var1, var2, var3)

  3 argument DM info trace.
- #define DM_TRACE_WARN0(msg) WSF_TRACE0("DM", "WARN", msg)

  0 argument DM warning trace.
- #define DM_TRACE_WARN1(msg, var1) WSF_TRACE1("DM", "WARN", msg, var1)

  1 argument DM warning trace.
- #define DM_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("DM", "WARN", msg, var1, var2)

  2 argument DM warning trace.
- #define DM_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("DM", "WARN", msg, var1, var2, var3)

  3 argument DM warning trace.
- #define DM_TRACE_ERR0(msg) WSF_TRACE0("DM", "ERR", msg)

  0 argument DM error trace.
- #define DM_TRACE_ERR1(msg, var1) WSF_TRACE1("DM", "ERR", msg, var1)

  1 argument DM error trace.
- #define DM_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("DM", "ERR", msg, var1, var2)

  2 argument DM error trace.
- #define DM_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("DM", "ERR", msg, var1, var2, var3)

  3 argument DM error trace.
- #define DM_TRACE_ALLOC0(msg) WSF_TRACE0("DM", "ALLOC", msg)

  0 argument DM buffer allocation trace.
- #define DM_TRACE_ALLOC1(msg, var1) WSF_TRACE1("DM", "ALLOC", msg, var1)

  1 argument DM buffer allocation trace.
- #define DM_TRACE_ALLOC2(msg, var1, var2) WSF_TRACE2("DM", "ALLOC", msg, var1, var2)

  2 argument DM buffer allocation trace.
- #define DM_TRACE_ALLOC3(msg, var1, var2, var3) WSF_TRACE3("DM", "ALLOC", msg, var1, var2, var3)

  3 argument DM buffer allocation trace.
- #define DM_TRACE_FREE0(msg) WSF_TRACE0("DM", "FREE", msg)

  0 argument DM buffer free trace.
- #define DM_TRACE_FREE1(msg, var1) WSF_TRACE1("DM", "FREE", msg, var1)

  1 argument DM buffer free trace.
- #define DM_TRACE_FREE2(msg, var1, var2) WSF_TRACE2("DM", "FREE", msg, var1, var2)

  2 argument DM buffer free trace.
- #define DM_TRACE_FREE3(msg, var1, var2, var3) WSF_TRACE3("DM", "FREE", msg, var1, var2, var3)

  3 argument DM buffer free trace.
- #define L2C_TRACE_INFO0(msg)

  0 argument L2C info trace.

- #define L2C_TRACE_INFO1(msg, var1)

  1 argument L2C info trace.
- #define L2C_TRACE_INFO2(msg, var1, var2)

  2 argument L2C info trace.
- #define L2C_TRACE_INFO3(msg, var1, var2, var3)

  3 argument L2C info trace.
- #define L2C_TRACE_WARN0(msg) WSF_TRACE0("L2C", "WARN", msg)

  0 argument L2C warning trace.
- #define L2C_TRACE_WARN1(msg, var1) WSF_TRACE1("L2C", "WARN", msg, var1)

  1 argument L2C warning trace.
- #define L2C_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("L2C", "WARN", msg, var1, var2)

  2 argument L2C warning trace.
- #define L2C_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("L2C", "WARN", msg, var1, var2, var3)

  3 argument L2C warning trace.
- #define L2C_TRACE_ERR0(msg) WSF_TRACE0("L2C", "ERR", msg)

  0 argument L2C error trace.
- #define L2C_TRACE_ERR1(msg, var1) WSF_TRACE1("L2C", "ERR", msg, var1)

  1 argument L2C error trace.
- #define L2C_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("L2C", "ERR", msg, var1, var2)

  2 argument L2C error trace.
- #define L2C_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("L2C", "ERR", msg, var1, var2, var3)

  3 argument L2C error trace.
- #define ATT_TRACE_INFO0(msg)

  0 argument ATT info trace.
- #define ATT_TRACE_INFO1(msg, var1)

  1 argument ATT info trace.
- #define ATT_TRACE_INFO2(msg, var1, var2)

  2 argument ATT info trace.
- #define ATT_TRACE_INFO3(msg, var1, var2, var3)

  3 argument ATT info trace.
- #define ATT_TRACE_WARN0(msg) WSF_TRACE0("ATT", "WARN", msg)

  0 argument ATT warning trace.
- #define ATT_TRACE_WARN1(msg, var1) WSF_TRACE1("ATT", "WARN", msg, var1)

  1 argument ATT warning trace.
- #define ATT_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("ATT", "WARN", msg, var1, var2)

  2 argument ATT warning trace.
- #define ATT_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("ATT", "WARN", msg, var1, var2, var3)

  3 argument ATT warning trace.
- #define ATT_TRACE_ERR0(msg) WSF_TRACE0("ATT", "ERR", msg)

  0 argument ATT error trace.
- #define ATT_TRACE_ERR1(msg, var1) WSF_TRACE1("ATT", "ERR", msg, var1)

  1 argument ATT error trace.
- #define ATT_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("ATT", "ERR", msg, var1, var2)

  2 argument ATT error trace.
- #define ATT_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("ATT", "ERR", msg, var1, var2, var3)

  3 argument ATT error trace.
- #define EATT_TRACE_INFO0(msg) WSF_TRACE0("EATT", "INFO", msg)

  0 argument EATT info trace.
- #define EATT_TRACE_INFO1(msg, var1) WSF_TRACE1("EATT", "INFO", msg, var1)

  1 argument EATT info trace.
- #define EATT_TRACE_INFO2(msg, var1, var2) WSF_TRACE2("EATT", "INFO", msg, var1, var2)

2 argument EATT info trace.

- #define EATT_TRACE_INFO3(msg, var1, var2, var3) WSF_TRACE3("EATT", "INFO", msg, var1, var2, var3)

  3 argument EATT info trace.
- #define EATT_TRACE_WARN0(msg) WSF_TRACE0("EATT", "WARN", msg)

  0 argument EATT warning trace.
- #define EATT_TRACE_WARN1(msg, var1) WSF_TRACE1("EATT", "WARN", msg, var1)

  1 argument EATT warning trace.
- #define EATT_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("EATT", "WARN", msg, var1, var2)

  2 argument EATT warning trace.
- #define EATT_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("EATT", "WARN", msg, var1, var2, var3)

  3 argument EATT warning trace.
- #define EATT_TRACE_ERR0(msg) WSF_TRACE0("EATT", "ERR", msg)

  0 argument EATT error trace.
- #define EATT_TRACE_ERR1(msg, var1) WSF_TRACE1("EATT", "ERR", msg, var1)

  1 argument EATT error trace.
- #define EATT_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("EATT", "ERR", msg, var1, var2)

  2 argument EATT error trace.
- #define EATT_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("EATT", "ERR", msg, var1, var2, var3)

  3 argument EATT error trace.
- #define SMP_TRACE_INFO0(msg)

  0 argument SMP info trace.
- #define SMP_TRACE_INFO1(msg, var1)

  1 argument SMP info trace.
- #define SMP_TRACE_INFO2(msg, var1, var2)

  2 argument SMP info trace.
- #define SMP_TRACE_INFO3(msg, var1, var2, var3)

  3 argument SMP info trace.
- #define SMP_TRACE_WARN0(msg) WSF_TRACE0("SMP", "WARN", msg)

  0 argument SMP warning trace.
- #define SMP_TRACE_WARN1(msg, var1) WSF_TRACE1("SMP", "WARN", msg, var1)

  1 argument SMP warning trace.
- #define SMP_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("SMP", "WARN", msg, var1, var2)

  2 argument SMP warning trace.
- #define SMP_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("SMP", "WARN", msg, var1, var2, var3)

  3 argument SMP warning trace.
- #define SMP_TRACE_ERR0(msg) WSF_TRACE0("SMP", "ERR", msg)

  0 argument SMP error trace.
- #define SMP_TRACE_ERR1(msg, var1) WSF_TRACE1("SMP", "ERR", msg, var1)

  1 argument SMP error trace.
- #define SMP_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("SMP", "ERR", msg, var1, var2)

  2 argument SMP error trace.
- #define SMP_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("SMP", "ERR", msg, var1, var2, var3)

  3 argument SMP error trace.
- #define **SMP_TRACE_BUF**(msg)
- #define **APP_TRACE_DEBUG**(msg, ...)
- #define APP_TRACE_INFO(msg, ...)

  Variadic argument App info trace.
- #define APP_TRACE_INFO0(msg)

  0 argument App info trace.

- #define APP_TRACE_INFO1(msg, var1)

    1 argument App info trace.
- #define APP_TRACE_INFO2(msg, var1, var2)

    2 argument App info trace.
- #define APP_TRACE_INFO3(msg, var1, var2, var3)

    3 argument App info trace.
- #define APP_TRACE_INFO4(msg, var1, var2, var3, var4)

    4 argument App info trace.
- #define APP_TRACE_INFO5(msg, var1, var2, var3, var4, var5)

    5 argument App info trace.
- #define APP_TRACE_INFO6(msg, var1, var2, var3, var4, var5, var6)

    6 argument App info trace.
- #define APP_TRACE_INFO7(msg, var1, var2, var3, var4, var5, var6, var7)

    7 argument App info trace.
- #define APP_TRACE_INFO8(msg, var1, var2, var3, var4, var5, var6, var7, var8)

    8 argument App info trace.
- #define APP_TRACE_INFO9(msg, var1, var2, var3, var4, var5, var6, var7, var8, var9)

    9 argument App info trace.
- #define APP_TRACE_INFO12(msg, var1, var2, var3, var4, var5, var6, var7, var8, var9, var10, var11, var12)

    12 argument App info trace.
- #define APP_TRACE_WARN(msg, ...) WSF_TRACE("APP", "WARN", msg, ##__VA_ARGS__)

    Variadic argument App warning trace.
- #define APP_TRACE_WARN0(msg) WSF_TRACE0("APP", "WARN", msg)

    0 argument App warning trace.
- #define APP_TRACE_WARN1(msg, var1) WSF_TRACE1("APP", "WARN", msg, var1)

    1 argument App warning trace.
- #define APP_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("APP", "WARN", msg, var1, var2)

    2 argument App warning trace.
- #define APP_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("APP", "WARN", msg, var1, var2, var3)

    3 argument App warning trace.
- #define APP_TRACE_ERR(msg, ...) WSF_TRACE("APP", "ERR", msg, ##__VA_ARGS__)

    Variadic argument App error trace.
- #define APP_TRACE_ERR0(msg) WSF_TRACE0("APP", "ERR", msg)

    0 argument App error trace.
- #define APP_TRACE_ERR1(msg, var1) WSF_TRACE1("APP", "ERR", msg, var1)

    1 argument App error trace.
- #define APP_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("APP", "ERR", msg, var1, var2)

    2 argument App error trace.
- #define APP_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("APP", "ERR", msg, var1, var2, var3)

    3 argument App error trace.
- #define **LL_TRACE_INFO0**(msg)
- #define LL_TRACE_INFO1(msg, var1)

    1 argument LL info trace.
- #define LL_TRACE_INFO2(msg, var1, var2)

    2 argument LL info trace.
- #define LL_TRACE_INFO3(msg, var1, var2, var3)

    3 argument LL info trace.
- #define LL_TRACE_WARN0(msg)

    0 argument LL warning trace.
- #define LL_TRACE_WARN1(msg, var1)

    1 argument LL warning trace.

- #define LL_TRACE_WARN2(msg, var1, var2)

  2 argument LL warning trace.
- #define LL_TRACE_WARN3(msg, var1, var2, var3)

  3 argument LL warning trace.
- #define LL_TRACE_ERR0(msg)

  0 argument LL error trace.
- #define LL_TRACE_ERR1(msg, var1)

  1 argument LL error trace.
- #define LL_TRACE_ERR2(msg, var1, var2)

  2 argument LL error trace.
- #define LL_TRACE_ERR3(msg, var1, var2, var3)

  3 argument LL error trace.
- #define AUD_TRACE_INFO0(msg)

  0 argument audio info trace.
- #define AUD_TRACE_INFO1(msg, var1)

  1 argument audio info trace.
- #define AUD_TRACE_INFO2(msg, var1, var2)

  2 argument audio info trace.
- #define AUD_TRACE_INFO3(msg, var1, var2, var3)

  3 argument audio info trace.
- #define AUD_TRACE_WARN0(msg)

  0 argument audio warning trace.
- #define AUD_TRACE_WARN1(msg, var1)

  1 argument audio warning trace.
- #define AUD_TRACE_WARN2(msg, var1, var2)

  2 argument audio warning trace.
- #define AUD_TRACE_WARN3(msg, var1, var2, var3)

  3 argument audio warning trace.
- #define AUD_TRACE_ERR0(msg)

  0 argument audio error trace.
- #define AUD_TRACE_ERR1(msg, var1)

  1 argument audio error trace.
- #define AUD_TRACE_ERR2(msg, var1, var2)

  2 argument audio error trace.
- #define AUD_TRACE_ERR3(msg, var1, var2, var3)

  3 argument audio error trace.
- #define **MESH_TRACE_DEBUG**(msg, ...)
- #define MESH_TRACE_INFO(msg, ...)

  Variadic argument MESH info trace.
- #define MESH_TRACE_INFO0(msg)

  0 argument MESH info trace.
- #define MESH_TRACE_INFO1(msg, var1)

  1 argument MESH info trace.
- #define MESH_TRACE_INFO2(msg, var1, var2)

  2 argument MESH info trace.
- #define MESH_TRACE_INFO3(msg, var1, var2, var3)

  3 argument MESH info trace.
- #define MESH_TRACE_WARN(msg, ...) WSF_TRACE("MESH", "WARN", msg, ##__VA_ARGS__)

  Variadic argument MESH warning trace.
- #define MESH_TRACE_WARN0(msg) WSF_TRACE0("MESH", "WARN", msg)

  0 argument MESH warning trace.

- #define MESH_TRACE_WARN1(msg, var1) WSF_TRACE1("MESH", "WARN", msg, var1)

  1 argument MESH warning trace.
- #define MESH_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("MESH", "WARN", msg, var1, var2)

  2 argument MESH warning trace.
- #define MESH_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("MESH", "WARN", msg, var1, var2, var3)

  3 argument MESH warning trace.
- #define MESH_TRACE_ERR(msg, ...) WSF_TRACE("MESH", "ERR", msg, ##__VA_ARGS__)

  Variadic argument MESH warning trace.
- #define MESH_TRACE_ERR0(msg) WSF_TRACE0("MESH", "ERR", msg)

  0 argument MESH error trace.
- #define MESH_TRACE_ERR1(msg, var1) WSF_TRACE1("MESH", "ERR", msg, var1)

  1 argument MESH error trace.
- #define MESH_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("MESH", "ERR", msg, var1, var2)

  2 argument MESH error trace.
- #define MESH_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("MESH", "ERR", msg, var1, var2, var3)

  3 argument MESH error trace.
- #define **MMDL_TRACE_DEBUG**(msg, ...)
- #define MMDL_TRACE_INFO(msg, ...)

  Variadic argument MMDL info trace.
- #define MMDL_TRACE_INFO0(msg)

  0 argument MMDL info trace.
- #define MMDL_TRACE_INFO1(msg, var1)

  1 argument MMDL info trace.
- #define MMDL_TRACE_INFO2(msg, var1, var2)

  2 argument MMDL info trace.
- #define MMDL_TRACE_INFO3(msg, var1, var2, var3)

  3 argument MMDL info trace.
- #define MMDL_TRACE_WARN(msg, ...) WSF_TRACE("MMDL", "WARN", msg, ##__VA_ARGS__)

  Variadic argument MMDL info trace.
- #define MMDL_TRACE_WARN0(msg) WSF_TRACE0("MMDL", "WARN", msg)

  0 argument MMDL warning trace.
- #define MMDL_TRACE_WARN1(msg, var1) WSF_TRACE1("MMDL", "WARN", msg, var1)

  1 argument MMDL warning trace.
- #define MMDL_TRACE_WARN2(msg, var1, var2) WSF_TRACE2("MMDL", "WARN", msg, var1, var2)

  2 argument MMDL warning trace.
- #define MMDL_TRACE_WARN3(msg, var1, var2, var3) WSF_TRACE3("MMDL", "WARN", msg, var1, var2, var3)

  3 argument MMDL warning trace.
- #define MMDL_TRACE_ERR(msg, ...) WSF_TRACE("MMDL", "ERR", msg, ##__VA_ARGS__)

  Variadic argument MMDL info trace.
- #define MMDL_TRACE_ERR0(msg) WSF_TRACE0("MMDL", "ERR", msg)

  0 argument MMDL error trace.
- #define MMDL_TRACE_ERR1(msg, var1) WSF_TRACE1("MMDL", "ERR", msg, var1)

  1 argument MMDL error trace.
- #define MMDL_TRACE_ERR2(msg, var1, var2) WSF_TRACE2("MMDL", "ERR", msg, var1, var2)

  2 argument MMDL error trace.
- #define MMDL_TRACE_ERR3(msg, var1, var2, var3) WSF_TRACE3("MMDL", "ERR", msg, var1, var2, var3)

  3 argument MMDL error trace.
- #define LL_TRACE_ENABLE(ena)

  Enable LL trace.

Trace macros

- #define **WSF_TRACE**(subsys, stat, msg, ...)
- #define **WSF_TRACE0**(subsys, stat, msg)
- #define **WSF_TRACE1**(subsys, stat, msg, var1)
- #define **WSF_TRACE2**(subsys, stat, msg, var1, var2)
- #define **WSF_TRACE3**(subsys, stat, msg, var1, var2, var3)
- #define **WSF_TRACE4**(subsys, stat, msg, var1, var2, var3, var4)
- #define **WSF_TRACE5**(subsys, stat, msg, var1, var2, var3, var4, var5)
- #define **WSF_TRACE6**(subsys, stat, msg, var1, var2, var3, var4, var5, var6)
- #define **WSF_TRACE7**(subsys, stat, msg, var1, var2, var3, var4, var5, var6, var7)
- #define **WSF_TRACE8**(subsys, stat, msg, var1, var2, var3, var4, var5, var6, var7, var8)
- #define **WSF_TRACE9**(subsys, stat, msg, var1, var2, var3, var4, var5, var6, var7, var8, var9)
- #define **WSF_TRACE12**(subsys, stat, msg, var1, var2, var3, var4, var5, var6, var7, var8, var9, var10, var11, var12)
- #define **WSF_TRACE_PRINT**(msg, ...)
- #define **WSF_TRACE_FLUSH**()

Typedefs

- typedef bool_t(∗ WsfTraceHandler_t) (const uint8_t ∗pBuf, uint32_t len)

  Token event handler.
- typedef void(∗ WsfBt4TraceCback_t) (const char ∗pStr, va_list args)

  BT4 Platform trace callback.

Functions

- void WsfToken (uint32_t tok, uint32_t var)

  Output tokenized message.
- void WsfTraceEnable (bool_t enable)

  Enable trace messages.
- void WsfTrace (const char ∗pStr,...)

  Output trace message.
- void WsfTraceRegisterHandler (WsfTraceHandler_t traceCback)

  Register trace handler.
- void WsfTraceRegister (WsfBt4TraceCback_t cback)

  Register BT4 platform trace callback function.
- bool_t WsfTokenService (void)

  Service the trace ring buffer.

4.25.1 Detailed Description

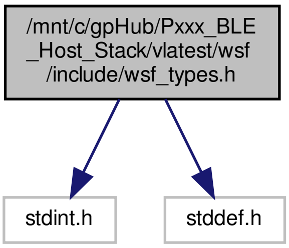
Trace message interface.

4.26 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/wsf/include/wsf_types.h File Reference

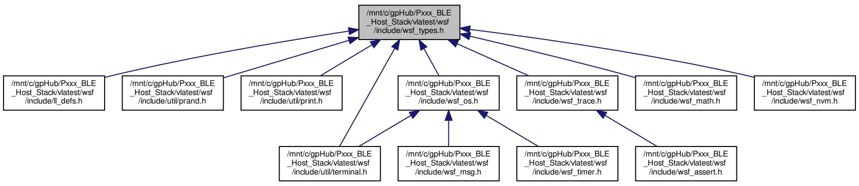
Platform-independent data types.

```
#include <stdint.h>
#include <stddef.h>
```

Include dependency graph for wsf_types.h:



This graph shows which files directly or indirectly include this file:



Macros

Integer Data Types

- #define **bool_t** uint8_t
- #define **FALSE** 0
- #define **TRUE** (!FALSE)
- #define **UINT64_C**(x) x##ULL
- #define **UINT32_C**(x) x##UL
- #define **UINT8_C**(x) (x)

4.26.1 Detailed Description

Platform-independent data types.

Index