WLAN Subsystem

Host Driver-Firmware Interface

IEEE 802.11a/g/b and draft-802.11n

Software Specification (Access Point, Station, Bridge)

Color	Description
Internal	Internal use only
Green	Fuji use only

Doc. No. MV-S800639-00, Rev. –

September 5, 2008

CONFIDENTIAL

Document Classification: Restricted

Marvell. Moving Forward Faster

Document Conventions

↘	**Note:** Provides related information or information of special importance.
!	**Caution:** Indicates potential damage to hardware or software, or loss of data.
ϟ	**Warning:** Indicates a risk of personal injury.

Document Status

Draft	For internal use. This document has not passed a complete technical review cycle and ECN signoff process.
Preliminary Tapeout (Advance)	This document contains design specifications for a product in its initial stage of design and development. A revision of this document or supplementary information may be published at a later date. Marvell may make changes to these specifications at any time without notice. Contact Marvell Field Application Engineers for more information.
Preliminary Information	This document contains preliminary specifications. A revision of this document or supplementary information may be published at a later date. Marvell may make changes to these specifications at any time without notice. . Contact Marvell Field Application Engineers for more information.
Complete Information	This document contains specifications for a product in its final qualification stages. Marvell may make changes to these specifications at any time without notice. Contact Marvell Field Application Engineers for more information.

Milestone Indicator:
Draft = 0.xx
Advance = 1.xx
Preliminary = 2.xx
Complete = 3.xx

X . Y Z

Work in Progress Indicator
Zero means document is released.

Various Revisions Indicator

Doc Status: 0.00	Technical Publication: 0.xx

For more information, visit our website at: www.marvell.com

Table of Contents

THIS PAGE INTENTIONALLY LEFT BLANK

1 Introduction

This document details the host-firmware interface commands for Marvell 88W8363-based WLAN SoCs used in Access Point, Sation, and Bridge applications.

The firmware interface consists of a list of command IDs, with their respective command structures. These Host Commands are used to configure the firmware for specific functionality and/or runtime operations.

For example:

Host command *HostCmd_CMD_BSS_START*—starts a 802.11 Basic Service Set (BSS) service

Host command *HostCmd_CMD_SET_HW_SPEC*—configures certain firmware/hardware capabilities with values supplied in its command structure.

2 Firmware Command Structures

2.1 Firmware Command Header

The standard command header for all firmware commands is as follows:

```
typedef PACK_START struct tagFWCmdHdr
{
    UINT16 Cmd;
    UINT16 Length;
    UINT8 SeqNum;
    UINT8 macid;
    UINT16 Result;
} PACK_END FWCmdHdr, *PFWCmdHdr;
```

where:

Parameter	Type	Description
Cmd	UINT16	Firmware command
Length	UINT16	Length of command including FWCmdHdr
SeqNum	UINT8	Command sequence number (not used)
macid	UINT8	ID of MAC interface (0,1) 0x00 to 00x7 = APs 0x08 to 0x1F = stations
Result	UINT16	Firmware command result 0x0000 = ok 0x0002 = command not valid 0x0003 = command pending (will be processed) 0x0004 = system busy (command ignored) 0x0005 = data buffer not big enough

2.2 List of Host Commands

Not all the commands are used/needed by the NetBSD drivers.

Table 1: Host Command List

Firmware Commands	Value
Section 2.3, General Firmware Attributes	
HostCmd_CMD_CODE_DNLD	0x0001
HostCmd_CMD_GET_HW_SPEC	0x0003
HostCmd_CMD_SET_HW_SPEC	0x0004
HostCmd_CMD_802_11_GET_STAT	0x0014
HostCmd_CMD_SET_KEEP_ALIVE	0x1112
Section 2.4, Firmware Memory and Register Access	
HostCmd_CMD_MAC_REG_ACCESS	0x0019
HostCmd_CMD_BBP_REG_ACCESS	0x001A
HostCmd_CMD_RF_REG_ACCESS	0x001B
HostCmd_CMD_MEM_ADDR_ACCESS	0x001D
Section 2.5, PHY Attributes and Power Control	
HostCmd_CMD_802_11_RADIO_CONTROL	0x001C
HostCmd_CMD_802_11_RF_TX_POWER	0x001E
HostCmd_CMD_802_11_RF_ANTENNA	0x0020
HostCmd_CMD_SET_RF_CHANNEL	0x010A
HostCmd_CMD_802_11H_DETECT_RADAR	0x0120
HostCmd_CMD_SET_REGION_POWER	0x0128
HostCmd_CMD_HT_GF_MODE	0x0140
HostCmd_CMD_HT_TX_STBC	0x0141
HostCmd_CMD_SET_SWITCH_CHANNEL	0x1121
HostCmd_CMD_SET_SPECTRUM_MGMT	0x1128
HostCmd_CMD_SET_POWER_CONSTRAINT	0x1129
HostCmd_CMD_GET_CALTABLE	0x1134
HostCmd_CMD_SET_REGION_CODE	0x1139
Section 2.6, Rate and Link Adaptation Attributes	

Table 1: Host Command List (Continued)

Firmware Commands	Value
HostCmd_CMD_802_11_RTS_THSD	0x0113
HostCmd_CMD_HT_GUARD_INTERVAL	0x0124
HostCmd_CMD_SET_FIXED_RATE	0x0126
HostCmd_CMD_SET_LINKADAPT_CS_MODE	0x0129
HostCmd_CMD_SET_RATE_ADAPT_MODE	0x0203
HostCmd_CMD_GET_RATETABLE	0x1137
HostCmd_CMD_AMPDU_RETRY_RATEDROP_MODE	0x1145
Section 2.7, BSS Attributes	
HostCmd_CMD_BROADCAST_SSID_ENABLE	0x0050
HostCmd_CMD_SET_BEACON	0x0100
HostCmd_CMD_SET_INFRA_MODE	0x010E
HostCmd_CMD_SET_G_PROTECT_FLAG	0x010F
HostCmd_CMD_SET_IES	0x0127
HostCmd_CMD_SET_MAC_ADDR	0x0202
HostCmd_CMD_DEL_MAC_ADDR	0x0206
HostCmd_CMD_BSS_START	0x1100
HostCmd_CMD_AP_BEACON	0x1101
HostCmd_CMD_SET_APMODE	0x1114
HostCmd_CMD_SET_COUNTRY_CODE	0x1130
HostCmd_CMD_SET_N_PROTECT_FLAGS	0x1131
HostCmd_CMD_SET_N_PROTECT_OPMODE	0x1132
HostCmd_CMD_SET_WSC_IE	0x1136
Section 2.8, Station Database and Security Attributes	
HostCmd_CMD_SET_AID	0x010D
HostCmd_CMD_SET_NEW_STN	0x1111
HostCmd_CMD_UPDATE_ENCRYPTION	0x1122
HostCmd_CMD_DWDS_ENABLE	0x1144
Section 2.9, QoS and AMPDU Attributes	
HostCmd_CMD_SET_EDCA_PARAMS	0x0115

Table 1: Host Command List (Continued)

Firmware Commands	Value
HostCmd_CMD_SET_WMM_MODE	0x0123
HostCmd_CMD_BASTREAM	0x1125
HostCmd_CMD_SET_RIFS	0x1126
HostCmd_CMD_SET_OPTIMIZATION_LEVEL	0x1133
HostCmd_CMD_GET_SEQNO	0x1143
HostCmd_CMD_CFEND_ENABLE	0x1146
Section 2.10, AP PowerSave Attributes	
HostCmd_CMD_SET_MIMOPSHT	0x1135
HostCmd_CMD_SET_POWERSAVESTATION	0x1140
HostCmd_CMD_SET_TIM	0x1141

2.3 General Firmware Attributes

2.3.1 HostCmd_CMD_CODE_DNLD

Firmware command to download firmware code.

Command passes code to firmware boot loader in blocks.

2.3.2 HostCmd_CMD_GET_HW_SPEC

Firmware command to get firmware hardware settings.

```
typedef PACK_START struct _HostCmd_DS_GET_HW_SPEC
{
    FWCmdHdr CmdHdr;
    UINT8 Version;
    UINT8 HostIf;
    UINT16 NumOfWCB;
    UINT16 NumOfMCastAdr;
    UINT8 PermanentAddr[6];
    UINT16 RegionCode;
    UINT16 NumberOfAntenna;
    UINT32 FWReleaseNumber;
    UINT32 WcbBase0;
    UINT32 RxPdWrPtr;
    UINT32 RxPdRdPtr;
    UINT32 ulFwAwakeCookie;
    UINT32 WcbBase1;
    UINT32 WcbBase2;
    UINT32 WcbBase3;
} PACK_END HostCmd_DS_GET_HW_SPEC, *PHostCmd_DS_GET_HW_SPEC;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Version	UINT8	Hardware version
HostIf	UINT8	Host interface
NumOfWCB	UINT16	Maximum number of WCB firmware can handle
NumofMCastAdr	UINT16	Maximum number of multicast address firmware can handle
PermanentAddr[6]	UINT8	MAC address
RegionCode	UINT16	Region code
NumberOfAntenna	UINT16	Number of antenna used
FWReleaseNumber	UINT32	4 bytes of firmware release number Example; 0x1234 = 1.2.3.4.

Parameter	**Type**	**Description**
WcbBase0	UINT32	WCB base 0 for Tx
RxPdWrPtr	UINT32	Rx firmware descriptor write position
RxPdRdPtr	UINT32	Rx firmware descriptor read position
ulFwAwakeCookie	UINT32	Firmware awake cookie (to ensure that the device is not in sleep mode)
WcbBase1	UINT32	Base address for Tx queue 1
WcbBase2	UINT32	Base address for Tx queue 2
WcbBase3	UINT32	Base address for Tx queue 3

2.3.3 HostCmd_CMD_SET_HW_SPEC

Firmware command to set hardware settings.

```
#define TOTAL_TX_QUEUES 4
typedef PACK_START struct _HostCmd_DS_SET_HW_SPEC
{
    FWCmdHdr CmdHdr;
    UINT8 Version;
    UINT8 HostIf;
    UINT16 NumOfMCastAdr;
    UINT8 PermanentAddr[6];
    UINT16 RegionCode;
    UINT32 FWReleaseNumber;
    UINT32 ulFwAwakeCookie;
    UINT32 DeviceCaps;
    UINT32 RxPdWrPtr;
    UINT32 NumTxQueues;
    UINT32 WcbBase[TOTAL_TX_QUEUES];
    UINT32 Flags;
    UINT32 TxWcbNumPerQueue;
    UINT32 TotalRxWcb;
} PACK_END HostCmd_DS_SET_HW_SPEC, *PHostCmd_DS_SET_HW_SPEC;
```

where:

Parameter	**Type**	**Description**
CmdHdr	FWCmdHdr	Standard firmware command header
Version	UINT8	Hardware version
HostIf	UINT8	Host interface
NumofMCastAdr	UINT16	Maximum number of multicast address firmware can handle
PermanentAddr[6]	UINT8	MAC address
RegionCode	UINT16	Region code

Parameter	Type	Description
FWReleaseNumber	UINT32	4 bytes of firmware release number Example; 0x1234 = 1.2.3.4.
ulFwAwakeCookie	UINT32	Firmware awake cookie (to ensure that the device is not in sleep mode)
DeviceCaps	UINT32	Not used, set to 0 (device capabilities)
RxPdWrPtr	UINT32	Rx shared memory queue
NumTxQueues	UINT32	Actual number of Tx queues on WcbBase array
WcbBase	UINT32	Tx WCB rings
Flags	UINT32	Each bit, when set to 1, enables a specific feature Currently supported features: Bit[31:8]: Reserved, set to 0 Bit[7]: Host encrypt/decrypt management packets Bit[6]: Host handles power save Bit[5]: Probe Response from host Bit[4]: Beacon from host Bit[3]: Disable MBSS Bits[2:0]: Reserved, set to 0
TxWcbNumPerQueue	UINT32	Number of WCBs per queue
TotalRxWcb	UINT32	Number of Rx queues Currently only one Rx queue is supported.

2.3.4 HostCmd_CMD_802_11_GET_STAT

Firmware command for setting firmware statistics.

```
typedef PACK_START struct _HostCmd_DS_802_11_GET_STAT
{
    FWCmdHdr CmdHdr;
    UINT32 TxRetrySuccesses;
    UINT32 TxMultipleRetrySuccesses;
    UINT32 TxFailures;
    UINT32 RTSSuccesses;
    UINT32 RTSFailures;
    UINT32 AckFailures;
    UINT32 RxDuplicateFrames;
    UINT32 FCSErrorCount;
    UINT32 TxWatchDogTimeouts;
    UINT32 RxOverflows
    UINT32 RxFragErrors;
    UINT32 RxMemErrors;
    UINT32 PointerErrors;
    UINT32 TxUnderflows;
    UINT32 TxDone;
    UINT32 TxDoneBufTryPut;
    UINT32 TxDoneBufPut;
    UINT32 Wait4TxBuf;
    UINT32 TxAttempts;
    UINT32 TxSuccesses;
    UINT32 TxFragments;
    UINT32 TxMulticasts;
    UINT32 RxNonCtlPkts;
    UINT32 RxMulticasts;
    UINT32 RxUndecryptableFrames;
    UINT32 RxICVErrors;
    UINT32 RxExcludedFrames;
} PACK_END HostCmd_DS_802_11_GET_STAT, *PHostCmd_DS_802_11_GET_STAT;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
TxRetrySuccesses	UINT32	Transmit retry successes counter
TxMultipleRetrySuccesses	UINT32	Transmit multiple retry success counter
TxFailures	UINT32	Transmit failure counter
RTSSuccesses	UINT32	Request to transmit success counter
RTSFailures	UINT32	Request to transmit failure counter
AckFailures	UINT32	Acknowledge failure counter
RxDuplicatesFrames	UINT32	Receive duplicate frame counter
FCSErrorCount	UINT32	Frame check sequence error counter
TxWatchDogTimeouts	UINT32	Transmit watchdog timeout counter
RxOverflows	UINT32	Receiver overflow counter
RxFragErrors	UINT32	Receive fragment error counter
RxMemErrors	UINT32	Receive memory error counter
PointerErrors	UINT32	Pointer error counter
TxUnderflows	UINT32	Transmit underflow counter
TxDone	UINT32	Transmit done counter
TxDoneBufTryPut	UINT32	Transmit done buffer try to put back counter
TxDoneBufPut	UINT32	Transmit done buffer put back counter
Wait4TxBuf	UINT32	Wait for transmit buffer counter
TxAttempts	UINT32	Transmit attempts counter
TxSuccesses	UINT32	Transmit successes counter
TxFragments	UINT32	Transmit fragments counter
TxMulticasts	UINT32	Transmit multicast counter
RxNonCtlPkts	UINT32	Receive non-control packet counter
RxMulticasts	UINT32	Receive multicast counter

Parameter	Type	Description
RxUndecryptableFrames	UINT32	Receive undecryptable frames counter
RxICVErrors	UINT32	Receive integrity check value error counter
RxExcludedFrames	UINT32	Receive excluded frames counter

2.3.5 HostCmd_CMD_SET_KEEP_ALIVE

Firmware command to check firmware status.

```
typedef PACK_START struct tagHostCmd_FW_SET_KEEP_ALIVE_TICK
{
    FWCmdHdr CmdHdr;
    UINT8 tick;
} PACK_END HostCmd_FW_SET_KEEP_ALIVE_TICK, *PHostCmd_FW_SET_KEEP_ALIVE_TICK;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
tick	UINT8	Tick Must be set to 0.

2.4 Firmware Memory and Register Access

2.4.1 HostCmd_CMD_MAC_REG_ACCESS

Firmware command for reading or writing MAC firmware registers.

```
typedef PACK_START struct _HostCmd_DS_MAC_REG_ACCESS
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT16 Offset;
    UINT32 Value;
    UINT16 Reserved;
} PACK_END HostCmd_DS_MAC_REG_ACCESS, *PHostCmd_MAC_RF_REG_ACCESS;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0000 = read register 0x0001 = write register
Offset	UINT16	Offset from base of all registers, 0x80000000
Value	UINT32	Value to write or value read from register memory location
Reserved	UINT16	Reserved, set to 0

2.4.2 HostCmd_CMD_BBP_REG_ACCESS

Firmware command for reading or writing baseband processor firmware registers.

```
typedef PACK_START struct _HostCmd_DS_BBP_REG_ACCESS
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT16 Offset;
    UINT8 Value;
    UINT8 Reserved[3];
} PACK_END HostCmd_DS_BBP_REG_ACCESS, *PHostCmd_DS_BBP_REG_ACCESS;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0000 = read register 0x0001 = write register
Offset	UINT16	Offset from base baseband processor register memory location
Value	UINT8	Value to write or value read from register memory location
Reserved[3]	UINT8	Reserved, set to 0

2.4.3 HostCmd_CMD_RF_REG_ACCESS

Firmware command for getting or setting RF firmware registers.

```
typedef PACK_START struct _HostCmd_DS_RF_REG_ACCESS
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT16 Offset;
    UINT8 Value;
    UINT8 Reserved[3];
} PACK_END HostCmd_DS_RF_REG_ACCESS, *PHostCmd_DS_RF_REG_ACCESS;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0000 = read register 0x0001 = write register
Offset	UINT16	Offset from base RF register memory location
Value	UINT8	Value to write or value read from register memory location
Reserved[3]	UINT8	Reserved, set to 0

2.4.4 HostCmd_CMD_MEM_ADDR_ACCESS

Firmware command for reading or writing firmware memory.

```
typedef PACK_START struct _HostCmd_DS_MEM_ADDR_ACCESS
{
    FWCmdHdr CmdHdr;
    UINT32 Address;
    UINT16 Length;
    UINT16 Reserved;
    UINT32 Value[64];
} PACK_END HostCmd_DS_MEM_ADDR_ACCESS, *PHostCmd_DS_MEM_ADDR_ACCESS;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Address	UINT32	Starting address of location (all accesses are 32-bit)
Length	UINT16	Length of memory to access
Reserved	UINT16	0x0000 = read block of memory Length (< 64) starting from location Address (32-bit words) 0x0001 = write single memory location Address with Value[0] (32-bit) 0x0002 = read 64 words of memory starting from location Address (32-bit words)
Value[64]	UINT32	Buffer of memory that was read or to be written

2.5 PHY Attributes and Power Control

2.5.1 HostCmd_CMD_802_11_RADIO_CONTROL

Firmware command for radio control.

```
typedef PACK_START struct _HostCmd_DS_801_11_RADIO_CONTROL
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT16 Control;
    UINT16 RadioOn;
} PACK_END HostCmd_DS_801_11_RADIO_CONTROL, *PHostCmd_DS_801_11_RADIO_CONTROL;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0001 = set radio control
Control	UINT16	Control 0x0001 = long preamble 0x0003 = short preamble 0x0005 = auto preamble
RadioOn	UINT16	Set to 0, no effect

2.5.2 HostCmd_CMD_802_11_RF_TX_POWER

Firmware command for setting transmit power level.

```
#define TX_POWER_LEVEL_TOTAL 8
typedef PACK_START struct _HostCmd_DS_802_11_RF_TX_POWER
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT16 SupportTxPowerLevel;
    UINT16 CurrentTxPowerLevel;
    UINT16 Reserved;
    UINT16 PowerLevelList[TX_POWER_LEVEL_TOTAL];
} PACK_END HostCmd_DS_802_11_RF_TX_POWER, *PHostCmd_DS_802_11_RF_TX_POWER;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0001 = use SupportTxPowerLevel 0x0002 = use PowerLevelList
SupportTxPowerLevel	UINT16	Power level (dBm)
CurrentTxPowerLevel	UINT16	Not used, set to 0
Reserved	UINT16	Reserved, set to 0
PowerLevelList	UINT16	Power level list (dBm)

2.5.3 HostCmd_CMD_802_11_RF_ANTENNA

Firmware command for setting antenna mode.

```
typedef PACK_START struct _HostCmd_DS_802_11_RF_ANTENNA
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT16 AntennaMode;
} PACK_END HostCmd_DS_802_11_RF_ANTENNA, *PHostCmd_DS_802_11_RF_ANTENNA;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0001 = set Rx antenna 0x0002 = set Tx antenna
AntennaMode	UINT16	Antenna A, B, and C For Rx antenna: 0x0001 = antenna A 0x0002 = antenna B 0x0003 = antenna C 0x0004 = antenna A, and B 0x0005 = antenna A, B, and C 0x0006 = antenna B and C 0x0007 = antenna A and C 0x0008 to 0xFFFF = reserved For Tx antenna: 0x0001 = antenna A 0x0002 = antenna B 0x0003 = antenna A and B 0x0004 = antenna C 0x0005 and 0x0006 = reserved 0x0007 = antenna A, B, and C 0x0008 to 0xFFFE = reserved 0xFFFF = antenna diversity

2.5.4 HostCmd_CMD_SET_RF_CHANNEL

Firmware command for setting RF channel.

```
typedef PACK_START struct tagHostCmd_FW_RF_CHANNEL
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT8 CurrentChannel;
    CHNL_FLAGS ChannelFlags;
} PACK_END HostCmd_FW_SET_RF_CHANNEL, *PHostCmd_FW_SET_RF_CHANNEL;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0001 = set channel
CurrentChannel	UINT8	Current channel 0x01 to 0x0E = 2.4 GHz 0x24 to 0xA5 = 5.0 GHz
ChannelFlags	CHNL_FLAGS	See Section 2.5.4.1, CHNL_FLAGS, on page 22 for details

2.5.4.1 CHNL_FLAGS

```
typedef PACK_START struct tagChnlFlags
{
    UINT32 FreqBand: 6;
    UINT32 ChnlWidth: 5;
    UINT32 ExtChnlOffset: 2;
    UINT32 Reserved: 19;
} CHNL_FLAGS, *PCHNL_FLAGS;
```

where:

Parameter	Type	Description

FreqBand	UINT32:6	Frequency band
		Bits[5:4] = reserved, set to 0
		Bit[3] = 1: 5.2 GHz
		Bit[2] = 1: 5.0 GHz
		Bit[1] = 1: 4.9 GHz
		Bit[0] = 1: 2.4 GHz

ChnlWidth	UINT32:5	Channel width
		Bits[10:9] = reserved, set to 0
		Bit[8] = 1: 40 MHz
		Bit[7] = 1: 20 MHz
		Bit[6] = 1: 10 MHz

Parameter	Type	Description
ExtChnlOffset	UINT32:2	Extension channel offset Bits[12:11]: 00 = no extension 01 = above 11 = below
Reserved	UINT32:19	Bits[31:13]: Reserved, set to 0

2.5.5 HostCmd_CMD_802_11H_DETECT_RADAR

Firmware command for radar detect.

```
typedef PACK_START struct _HostCmd_802_11h_Detect_Radar
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT16 RadarTypeCode;
} PACK_END HostCmd_802_11h_Detect_Radar, *PHostCmd_802_11h_Detect_Radar;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0000 = DFS disable 0x0001 = check channel available start 0x0002 = check channel available stop 0x0003 = in service monitor start
RadarTypeCode	UINT16	Radar type code 0x0000 = default 0x0035 = JP W53 0x0038 = JP W56 0x0083 = ETSI 1.3.1

2.5.6 HostCmd_CMD_SET_REGION_POWER

Firmware command to set region power.

```
typedef PACK_START struct _HostCmd_DS_SET_REGION_POWER
{
    FWCmdHdr CmdHdr;
    UINT16 MaxPowerLevel;
    UINT16 Reserved;
} PACK_END HostCmd_DS_SET_REGION_POWER, *PHostCmd_DS_SET_REGION_POWER;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
MaxPowerLevel	UINT16	Power level (dBm) Default maximum 30 dBm.
Reserved	UINT16	Reserved, set to 0

2.5.7 HostCmd_CMD_HT_GF_MODE

Firmware command to set greenfield mode. This command is supported with 88W8366 firmware. Not supported with 88W8363 firmware.

```
typedef struct tagHostCmd_FW_HT_GF_MODE
{
    FWCmdHdr CmdHdr;
    UINT32 Action;
    UINT32 Mode;
} HostCmd_FW_HT_GF_MODE, *PHostCmd_FW_HT_GF_MODE;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT32	Action 0x00000000 = get greenfield mode 0x00000001 = set greenfield mode
Mode	UINT32	Mode 0x00000000 = disable 0x00000001 = enable

2.5.8 HostCmd_CMD_HT_TX_STBC

Firmware command to set STBC mode. This command is supported with 88W8366 firmware. Not supported with 88W8363 firmware.

```
typedef struct tagHostCmd_FW_HT_STBC_TX
{
    FWCmdHdr CmdHdr;
    UINT32 Action;
    UINT32 Mode;
} HostCmd_FW_HT_STBC_TX, *PHostCmd_FW_HT_STBC_TX;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT32	Action 0x00000000 = get STBC mode 0x00000001 = set STBC mode
Mode	UINT32	Mode 0x00000000 = disable STBC 0x00000001 = enable STBC 0x00000002 = reserved (do not use) 0x00000003 = reserved (do not use)

2.5.9 HostCmd_CMD_SET_SWITCH_CHANNEL

Firmware command to switch channel.

```
typedef PACK_START struct _HostCmd_SET_SWITCH_CHANNEL
{
    FWCmdHdr CmdHdr;
    UINT32 Next11hChannel;
    UINT32 Mode;
    UINT32 InitialCount;
    CHNL_FLAGS ChannelFlags;
} PACK_END HostCmd_SET_SWITCH_CHANNEL, *PHostCmd_SET_SWITCH_CHANNEL;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Next11hChannel	UINT32	Next channel that AP will switch to
Mode	UINT32	Indicates if station(s) are restricted from transmitting packets until the scheduled channel switch 0x00000000 = no restriction 0x00000001 = restricted from transmitting

Parameter	Type	Description
InitialCount	UINT32	Starting count for switch (number of beacons)
ChannelFlags	CHNL_FLAGS	Channel flags type defined in Section 2.5.4.1, CHNL_FLAGS, on page 22.

2.5.10 HostCmd_CMD_SET_SPECTRUM_MGMT

Firmware command for setting spectrum management.

```
typedef PACK_START struct _HostCmd_SET_SPECTRUM_MGMT
{
    FWCmdHdr CmdHdr;
    UINT32 SpectrumMgmt;
} PACK_END HostCmd_SET_SPECTRUM_MGMT, *PHostCmd_SET_SPECTRUM_MGMT;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
SpectrumMgmt	UINT32	Spectrum management 0x00000000 = remove spectrum management capability from beacon and probe response 0x00000001 = add spectrum management capability to beacon and probe response

2.5.11 HostCmd_CMD_SET_POWER_CONSTRAINT

Firmware command for setting power constraint.

```
typedef PACK_START struct _HostCmd_SET_POWER_CONSTRAINT
{
    FWCmdHdr CmdHdr;
    SINT32 PowerConstraint;
} PACK_END HostCmd_SET_POWER_CONSTRAINT, *PHostCmd_SET_POWER_CONSTRAINT;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
PowerConstraint	SINT32	Power constraint (dBm) (Applies to 802.11a and draft-802.11n modes).

2.5.12 HostCmd_CMD_GET_CALTABLE

Firmware command for getting calibration table data.

```
#define CAL_TBL_SIZE 160
typedef PACK_START struct tagHostCmd_FW_GET_CALTABLE
{
    FWCmdHdr CmdHdr;
    UINT8 annex;
    UINT8 index;
    UINT8 len;
    UINT8 Reserved;
    UINT8 calTbl[CAL_TBL_SIZE];
} PACK_END HostCmd_FW_GET_CALTABLE, *PHostCmd_FW_GET_CALTABLE;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
annex	UINT8	Example of Annex supported by 88W8363 firmware 0x00 = hardware information 0x01 to 0x0D = reserved, set to 0 0x0E = 802.11g/b calibration table 0x0F = 802.11a calibration table 0x10 = power calibration table 0x11 = Rx LNA calibration table 0x12 = reserved, set to 0 0x13 = Rx IQ calibration for 802.11g/b 0x14 = reserved, set to 0 0x15 = Rx IQ calibration for 802.11a 0x16 to 0x19 = reserved, set to 0 0x1A = 802.11g/b signal calibration 0x1B to 0x1C = reserved, set to 0 0x1D = hardware band support 0x1E = Lo spurs 0x1F = RSSI 802.11g/b 0x20 = RSSI 802.11a 0x21 = power table 2.4 GHz 20 MHz 0x22 = power table 2.4 GHz 40 MHz 0x23 = power table 5 GHz 20 MHz 0x24 = power table 5 GHz 40 MHz 0x25 = 802.11g/b calibration data annex 37 0x26 to 0xFF = reserved, set to 0 0xFF = SPI header
index	UINT8	Index of related table
len	UINT8	Size of HostCmd_FW_GET_CALTABLE
Reserved	UINT8	Reserved, set to 0
calTbl	UINT8	Buffer for calibration table requested

2.5.13 HostCmd_CMD_SET_REGION_CODE

Firmware command for setting region code.

```
typedef PACK_START struct _HostCmd_SET_REGIONCODE_INFO
{
    FWCmdHdr CmdHdr;
    UINT16 regionCode;
} PACK_END HostCmd_SET_REGIONCODE_INFO, *PHostCmd_SET_REGIONCODE_INFO;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
regionCode	UINT16	Region code for country

2.6 Rate and Link Adaptation Attributes

2.6.1 HostCmd_CMD_802_11_RTS_THSD

Firmware command for setting RTS threshold.

```
typedef PACK_START struct _HostCmd_DS_802_11_RTS_THSD
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT16 Threshold;
} PACK_END HostCmd_DS_802_11_RTS_THSD, *PHostCmd_DS_802_11_RTS_THSD;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0001 = set RTS threshold 0x0002 = get RTS threshold
Threshold	UINT16	Threshold Value between 0xFF and 0x92B (default 0x92B).

2.6.2 HostCmd_CMD_HT_GUARD_INTERVAL

Firmware command for setting guard interval.

```
typedef PACK_START struct tagHostCmd_FW_HT_GUARD_INTERVAL
{
    FWCmdHdr CmdHdr;
    UINT32 Action;
    GI_TYPE GIType;
} PACK_END HostCmd_FW_HT_GUARD_INTERVAL, *PHostCmd_FW_HT_GUARD_INTERVAL;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT32	Action 0x00000000 = get guard interval 0x00000001 = set guard interval
GIType	GI_TYPE	See Section 2.6.2.1, GI_TYPE, on page 30

2.6.2.1 GI_TYPE

```
typedef PACK_START struct tagGI_TYPE
{
    UINT32 LongGI: 1;
    UINT32 ShortGI: 1;
    UINT32 RESV: 30;
} PACK_END GI_TYPE, *PGIType;
```

where:

Parameter	Type	Description
LongGI	UINT32:1	Long guard interval Bit[0] = 0: use long guard interval
ShortGI	UINT32:1	Short guard interval Bit[1] = 1: use short guard interval
RESV	UINT32:30	Bits[31:2]: Reserved, set to 0

2.6.3 HostCmd_CMD_SET_FIXED_RATE

Firmware command for setting fixed rate.

```
typedef struct tagHostCmd_FW_USE_FIXED_RATE
{
    FWCmdHdr CmdHdr;
    UINT32 Action;
    UINT32 AllowRateDrop;
    UINT32 EntryCount;
    FIXED_RATE_ENTRY FixedRateTable[4];
    UINT8 MulticastRate;
    UINT8 MultiRateTxType;
    UINT8 ManagementRate;
} PACK_END HostCmd_FW_USE_FIXED_RATE, *PHostCmd_FW_USE_FIXED_RATE;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT32	Action 0x00000001 = use fixed rate 0x00000002 = do not use fixed rate
AllowRateDrop	UINT32	Allow rate drop 0x00000000 = fixed rate with auto rate drop 0x00000001 = fixed rate without auto rate drop
EntryCount	UINT32	Entry count Up to 4 rates for auto rate drop; lower rate after the retry count.

Parameter	Type	Description
FixedRateTable[4]	FIXED_RATE_ENTRY	Not used, set to 0
MulticastRate	UINT8	Rate used for multicast packets
MultiRateTxType	UINT8	Multi rate Tx type 0x00 = MulticastRate is legacy rate type 0x01 = MulticastRate is HT rate type
ManagementRate	UINT8	Rate used for management packets

2.6.4 HostCmd_CMD_SET_LINKADAPT_CS_MODE

Firmware command for setting link adaptation Card Select (CS) mode.

```
typedef PACK_START struct _HostCmd_DS_SET_LINKADAPT_CS_MODE
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT16 CSMode;
} PACK_END HostCmd_DS_SET_LINKADAPT_CS_MODE, *PHostCmd_DS_SET_LINKADAPT_CS_MODE;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Set link adapt CS mode
CSMode	UINT16	CS mode 0x0000 = CS adapt conservative 0x0001 = CS adapt aggressive 0x0002 = CS adapt auto 0x0003 = CS adapt disabled

2.6.5 HostCmd_CMD_SET_RATE_ADAPT_MODE

Firmware command for setting rate adaptation mode.

```
typedef PACK_START struct _HostCmd_DS_SET_RATE_ADAPT_MODE
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT16 RateAdaptMode;
} PACK_END HostCmd_DS_SET_RATE_ADAPT_MODE, *PHostCmd_DS_SET_RATE_ADAPT_MODE;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0001 = set rate adapt mode
RateAdaptMode	UINT16	Rate adapt mode 0x0000 = rate adapt mode indoor 0x0001 = rate adapt mode outdoor

2.6.6 HostCmd_CMD_GET_RATETABLE

Firmware command for getting station rate table.

```
typedef PACK_START struct tagHostCmd_FW_GET_RATETABLE
{
    FWCmdHdr CmdHdr;
    UINT8 Addr[6];
    UINT16 SortedRatesIndexMap[100];
} PACK_END HostCmd_FW_GET_RATETABLE, *PHostCmd_FW_GET_RATETABLE;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Addr[6]	UINT8	MAC address of station
SortedRatesIndexMap[100]	UINT16	Sorted rates map of station

2.6.7 HostCmd_CMD_AMPDU_RETRY_RATEDROP_MODE

Firmware command to enable AMPDU retry aggressive ratedrop action. This option allows a client at range or has difficulty to hear the packets at high rates to receive the retried packets sooner with lower rates.

```
typedef PACK_START struct tagHostCmd_FW_AMPDU_RETRY_RATEDROP_MODE
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT32 Option;
    UINT32 Threshold;
} PACK_END HostCmd_FW_AMPDU_RETRY_RATEDROP_MODE,
*PHostCmd_FW_AMPDU_RETRY_RATEDROP_MODE;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0000 = get AMPDU retry ratedrop mode 0x0001 = set AMPDU retry ratedrop mode
Option	UINT32	AMPDU retry ratedrop mode 0x00000000 = disable (default) 0x00000001 = enable aggressive ratedrop for AMPDU
Threshold	UINT32	Threshold Range: 0x00000000 to 0x000000C8. Default is 0x00000050. When AMPDU transmits threshold packets with no acknowledgement for the packets after 10ms, a fast ratedrop will be triggered.

2.7 BSS Attributes

2.7.1 HostCmd_CMD_BROADCAST_SSID_ENABLE

Firmware command to enable/disable broadcast SSID.

```
typedef PACK_START struct tagHostCmd_BSS_START
{
    FWCmdHdr CmdHdr;
    UINT32 Enable;
} PACK_END HostCmd_BSS_START, *PHostCmd_BSS_START;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Enable	UINT32	SSID broadcast enable 0x00000000 = enable 0x00000001 = disable

2.7.2 HostCmd_CMD_SET_BEACON

Firmware command for setting AP beacon. This command copies the received beacon directly to the beacon buffer.

```
typedef PACK_START struct _HostCmd_FW_SET_BCN_CMD
{
    FWCmdHdr CmdHdr;
    UINT16 FrmBodyLen;
    UINT8 FrmBody[1];
} PACK_END HostCmd_FW_SET_BCN_CMD, *PHostCmd_FW_SET_BCN_CMD;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
FrmBodyLen	UINT16	Beacon frame body length
FrmBody[1]	UINT8	Beacon frame body starting address

2.7.3 HostCmd_CMD_SET_INFRA_MODE

Firmware command for setting up station mode.

```
typedef PACK_START struct tagHostCmd_FW_SET_INFRA_MODE
{
    FWCmdHdr CmdHdr;
} PACK_END HostCmd_FW_SET_INFRA_MODE, *PHostCmd_FW_SET_INFRA_MODE;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header

Note

No other parameters are required. Mode is set to station whenever command is received by firmware.

2.7.4 HostCmd_CMD_SET_G_PROTECT_FLAG

Firmware command for setting 802.11g protection enable Extended Rate PHY (ERP).

```
typedef PACK_START struct tagHostCmd_FW_SET_G_PROTECT_FLAG
{
    FWCmdHdr CmdHdr;
    UINT32 GProtectFlag;
} PACK_END HostCmd_FW_SET_G_PROTECT_FLAG, *PHostCmd_FW_SET_G_PROTECT_FLAG;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header

GProtectFlag	UINT32	802.11g protection flag
		Bits[31:2] = reserved, set to 0
		Bit[1] = enable protection
		Bit[0] = reserved, set to 0

2.7.5 HostCmd_CMD_SET_IES

Firmware command to set information elements to end of beacon/probe response. This command is not used for hostFormBeacon mode.

```
typedef struct tagHostCmd_FW_SetIEs
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT16 IeListLen;
    UINT8 IeList[100];
} HostCmd_FW_SetIEs, *PHostCmd_FW_SetIEs;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0000 = not set 0x0001 = set
IeListLen	UINT16	Size of information elements to add to beacon/probe
IeList[100]	UINT8	Information elements

2.7.6 HostCmd_CMD_SET_MAC_ADDR

Firmware command for setting MAC address of AP or station for receive packet filter table.

```
typedef PACK_START struct tagHostCmd_FW_SET_MAC
{
    FWCmdHdr CmdHdr;
    UINT16 MacType;
    UINT8 MacAddr[6];
} PACK_END HostCmd_DS_SET_MAC, *PHostCmd_DS_SET_MAC,
HostCmd_FW_SET_BSSID, *PHostCmd_FW_SET_BSSID,
HostCmd_FW_SET_MAC, *PHostCmd_FW_SET_MAC;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
MacType	UINT16	MAC type 0x0000 = primary client 0x0001 = secondary client 0x0002 = primary AP 0x0003 = secondary AP
MacAddr[6]	UINT8	Standard 6-byte MAC address

2.7.7 HostCmd_CMD_DEL_MAC_ADDR

Firmware command for deleting MAC address of AP or station from receive packet filter table.

```
typedef PACK_START struct tagHostCmd_FW_SET_MAC
{
    FWCmdHdr CmdHdr;
    UINT16 MacType;
    UINT8 MacAddr[6];
} PACK_END HostCmd_FW_SET_MAC, *PHostCmd_FW_SET_MAC;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
MacType	UINT16	Not used, set to 0
MacAddr[6]	UINT8	Standard 6-byte MAC address

2.7.8 HostCmd_CMD_BSS_START

Firmware command to enable/disable BSS.

```
typedef PACK_START struct tagBSS_START
{
    FWCmdHdr CmdHdr;
    UINT32 Enable;
} PACK_END HostCmd_BSS_START, *PHostCmd_BSS_START;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Enable	UINT32	Enable 0x00000000 = disable 0x00000001 = enable

2.7.9 HostCmd_CMD_AP_BEACON

Firmware command for configuring beacon.

```
typedef PACK_START struct tagHostCmd_AP_Beacon
{
    FWCmdHdr CmdHdr;
    IEEEtypes_StartCmd_t StartCmd;
} PACK_END HostCmd_AP_Beacon, *PHostCmd_AP_Beacon;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
StartCmd	IEEEtypes_StartCmd_t	Beacon configure structure IEEEtypes_StartCmd_t as defined in Section 2.7.9.1, IEEEtypes_StartCmd_t, on page 38. **NOTE:** Beacon provided in IEEEtypes_StartCmd_t structure can have Information Elements added to it. This structure is provided only as a basic template.

2.7.9.1 IEEEtypes_StartCmd_t

```
typedef struct IEEEtypes_StartCmd_t
{
    IEEEtypes_MacAddr_t StaMacAddr;
    IEEEtypes_SsId_t SsId;
    IEEEtypes_Bss_e BssType;
    IEEEtypes_BcnInterval_t BcnPeriod;
    IEEEtypes_DtimPeriod_t DtimPeriod;
    IEEEtypes_SsParamSet_t SsParamSet;
    IEEEtypes_PhyParamSet_t PhyParamSet;
    UINT16 ProbeDelay;
    IEEEtypes_CapInfo_t CapInfo;
    IEEEtypes_DataRate_t BssBasicRateSet[IEEEtypes_MAX_DATA_RATE_G];
    IEEEtypes_DataRate_t OpRateSet[IEEEtypes_MAX_DATA_RATE_G];
    IEEEtypes_RSN_IE_t thisStaRsnIE;
    IEEEtypes_RSN_IE48_t thisStaRsnIE48;
    WME_param_elem_t thisStaWMEparam;
    IEEEtypes_COUNTRY_IE_t Country;
    UINT32 ApRFType;
    IEEEtypes_PowerConstraintElement_t PowerConstriantIE;
} PACK_END IEEEtypes_StartCmd_t;
```

where:

Parameter	**Type**	**Description**
StaMacAddr	IEEEtypes_MACAddr_t	Station MAC address
SsId	IEEEtypes_SsId_t	SSID of AP (see Section 2.7.9.9, Other Parameters, on page 45)
BssType	IEEEtypes_Bss_e	BSS type 0x1 = infrastructure 0x2 = independent 0x3 = any See Section 2.7.9.2, IEEEtypes_Bss_e, on page 40.
BcnPeriod	IEEEtypes_BcnInterval_t	Beacon period (see Section 2.7.9.9, Other Parameters, on page 45)
DtimPeriod	IEEEtypes_DtimPeriod_t	DTIM period in beacons (see Section 2.7.9.9, Other Parameters, on page 45)
SsParamSet	IEEEtypes_SsParamSet_t	Parameters for IBSS or CF operation (see Section 2.7.9.3, IEEEtypes_SsParamSet_t, on page 40)
PhyParamSet	IEEEtypes_PhyParamSet_t	Physical RF parameter set (see Section 2.7.9.4, IEEEtypes_PhyParamSet_t, on page 41)
ProbeDelay	UINT16	Probe delay (not used, set to 0)
CapInfo	IEEEtypes_CapInfo_t	Capability information (see Section 2.7.9.5, IEEEtypes_CapInfo_t, on page 41)
BssBasicRateSet	IEEEtypes_DataRate_t	BSS base rate (see Section 2.7.9.9, Other Parameters, on page 45)
OpRateSet	IEEEtypes_DataRate_t	Optional rate set (see Section 2.7.9.9, Other Parameters, on page 45)
thisStaRsnIE	IEEEtypes_RSN_IE_t	WPA RSN information element (see Section 2.7.9.6, IEEEtypes_RSN_IE_t, on page 43)
thisStaRSNIE48	IEEEtypes_RSN_IE48_t	WPA2 RSN information element (see Section 2.7.9.7, IEEEtypes_RSN_IE48_t, on page 43)
thisStaWMEparam	IEEEtypes_WME_param_elem_t	WME information element (see Section 2.7.9.8, IEEEtypes_WME_param_elem_t, on page 44)
Country	IEEEtypes_COUNTRY_IE_t	Country information (see Section 2.7.9.10, IEEEtypes_COUNTRY_IE_t, on page 45)
ApRFType	UINT32	RF type of AP 0x00000000 = 802.11b 0x00000001 = 802.11g 0x00000002 = mixed 0x00000003 = 802.11a 0x00000004 = 802.11j (10 MHz)
PowerConstraintIE	IEEEtypes_PowerConstraintElement_t	Power constraint information element (see Section 2.7.9.11, IEEEtypes_PowerConstraintElement_t, on page 45)

2.7.9.2 IEEEtypes_Bss_e

```
typedef enum
{
    BSS_INFRASTRUCUTURE = 1,
    BSS_INDEPENDENT,
    BSS_ANY
} PACK_END IEEEtypes_Bss_e;
```

2.7.9.3 IEEEtypes_SsParamSet_t

```
typedef union IEEEtypes_SsParamSet_t
{
    IEEEtypes_CfParamSet_t CfParamSet;
    IEEEtypes_IbssParamSet_t IbssParamSet;
} PACK_END IEEEtypes_SsParamSet_t;
typedef struct IEEEtypes_CfParamSet_t
{
    IEEEtypes_ElementId_e ElementId;
    IEEEtypes_Len_t Len;
    UINT8 CfpCnt;
    UINT8 CfpPeriod;
    UINT16 CfpMaxDuration;
    UINT16 CfpDurationRemaining;
} PACK_END IEEEtypes_CfParamSet_t;
typedef struct IEEEtypes_IbssParamSet_t
{
    IEEEtypes_ElementId_e ElementId;
    IEEEtypes_Len_t Len;
    UINT16 AtimWindow;
} PACK_END IEEEtypes_IbssParamSet_t;
```

2.7.9.4 IEEEtypes_PhyParamSet_t

```
typedef union IEEEtypes_PhyParamSet_t
{
    IEEEtypes_FhParamSet_t FhParamSet;
    IEEEtypes_DsParamSet_t DsParamSet;
} PACK_END IEEEtypes_PhyParamSet_t;
typedef union IEEEtypes_FhParamSet_t
{
    IEEEtypes_ElementId_e ElementId;
    IEEEtypes_Len_t Len;
    UINT16 DwellTime;
    UINT8 HopSet;
    UINT8 HopPattern;
    UINT8 HopIndex;
} PACK_END IEEEtypes_FhParamSet_t;
typedef struct IEEEtypes_DsParamSet_t
{
    IEEEtypes_ElementId_e ElementId;
    IEEEtypes_Len_t Len;
    UINT8 CurrentChan;
} PACK_END IEEEtypes_DsParamSet_t;
```

2.7.9.5 IEEEtypes_CapInfo_t

Capabilities information.

```
typedef struct IEEEtypes_CapInfo_t
{
    UINT16 Ess: 1;
    UINT16 Ibss: 1;
    UINT16 CfPollable: 1;
    UINT16 CfPollRqst: 1;
    UINT16 Privacy: 1;
    UINT16 ShortPreamble: 1;
    UINT16 Pbcc: 1
    UINT16 ChangAgility: 1;
    UINT16 SpectrumMgmt: 1;
    UINT16 QoS: 1;
    UINT16 ShortSlotTime: 1;
    UINT16 APSD: 1;
    UINT16 Rsrvd1: 1;
    UINT16 DsssOfdm: 1;
    UINT16 BlckAck: 1;
    UINT16 Rsrvd2: 1;
} PACK_END IEEEtypes_CapInfo_t;
```

where:

Parameter	Type	Description
Ess	UINT16:1	Bit[0]: ESS 0 = not ESS 1 = ESS type
Ibss	UINT16:1	Bit[1]: IBSS 0 = not IBSS 1 = IBSS type
CfPollable	UINT16:1	Bit[2]: CF pollable 0 = not CF pollable 1 = CF pollable
CfPollRqst	UINT16:1	Bit[3]: CF poll request 0 = not requested 1 = requested
Privacy	UINT16:1	Bit[4]: Privacy 0 = disabled 1 = enabled
ShortPreamble	UINT16:1	Bit[5]: Short preamble 0 = disabled 1 = enabled
Pbcc	UINT16:1	Bit[6]: PBCC 0 = not allowed 1 = allowed
ChanAgility	UINT16:1	Bit[7]: Channel agility 0 = not used 1 = used
SpectrumMgmt	UINT16:1	Bit[8]: Spectrum management 0 = disabled 1 = enabled
QoS	UINT16:1	Bit[9]: QoS 0 = QoS not supported 1 = QoS supported
ShortSlotTime	UINT16:1	Bit[10]: 802.11g mode short slot time 0 = disabled 1 = enabled
APSD	UINT16:1	Bit[11]: APSD 0 = not supported 1 = supported
Rsrvd1	UINT16:1	Bit[12]: Reserved, set to 0
DsssOfdm	UINT16:1	Bit[13]: DSSS-OFDM 0 = not allowed 1 = allowed

Parameter	Type	Description
BlckAck	UINT16:1	Bit[14]: Delayed block acknowledgement 0 = not allowed 1 = allowed
Rsrvd2	UINT16:1	Bit[15]: Reserved, set to 0

2.7.9.6 IEEEtypes_RSN_IE_t

```
typedef PACK_START struct IEEEtypes_RSN_IE_t
{
    UINT8 ElemId;
    UINT8 Len;
    UINT8 OuiType[4];
    UINT8 Ver[2];
    UINT8 GrpKeyCipher[4];
    UINT8 PwsKeyCnt[2];
    UINT8 PwsKeyCipherList[4]
    UINT8 AuthKeyCnt[2];
    UINT8 AuthKeyList[4];
    UINT8 RsnCap[2];
} PACK_END IEEEtypes_RSN_IE_t;
```

2.7.9.7 IEEEtypes_RSN_IE48_t

```
typedef PACK_START struct IEEEtypes_RSN_IE48_t
{
    UINT8 ElemId;
    UINT8 Len;
    UINT8 Ver[2];
    UINT8 GrpKeyCipher[4];
    UINT8 PwsKeyCnt[2];
    UINT8 PwsKeyCipherList[4]
    UINT8 AuthKeyCnt[2];
    UINT8 AuthKeyList[4];
    UINT8 RsnCap[2];
} PACK_END IEEEtypes_RSN_IE48_t;
```

2.7.9.8 IEEEtypes_WME_param_elem_t

```
typedef PACK_START struct IEEEtypes_WME_param_elem_t
{
    UINT8 ElemId;
    UINT8 Len;
    OUI_t OUI;
    UINT8 version;
    QoS_Info_t Qos_info;
    UINT8 rsvd;
    AC_param_rcd_t AC_BE;
    AC_param_rcd_t AC_BK;
    AC_param_rcd_t AC_VI;
    AC_param_rcd_t AC_VO;
} PACK_END IEEEtypes_WME_param_elem_t;
typedef PACK_START struct AC_param_rcd_t
{
    ACI_AIFSN_field_t ACI_AIFSN;
    ECW_min_max_field_t ECW_min_max;
    UINT16 TXOP_lim;
} PACK_END AC_param__rcd_t;
typedef PACK_START struct ACI_AIFSN_field_t
{
    UINT8 AIFSN: 4;
    UINT8 ACM: 1;
    UINT8 ACI: 2;
    UINT8 rsvd: 1;
} PACK_END ACI_AIFSN_field_t;
typedef PACK_START struct ECW_min_max_field_t
{
    UINT8 ECW_min: 4;
    UINT8 ECW_max: 4;
} PACK_END ECW_min_max_field_t;
```

2.7.9.9 Other Parameters

```
typedef UINT8 IEEEtypes_SsId_t[IEEEtypes_SSID_SIZE];

typedef UINT16 IEEEtypes_BcnInterval_t;

typedef UINT8 IEEEtypes_DtimPeriod_t;

typedef UINT8 IEEEtypes_DataRate_t;
```

2.7.9.10 IEEEtypes_COUNTRY_IE_t

```
typedef struct IEEEtypes_COUNTRY_IE_t
{
    IEEEtypes_ElementId_e ElemId;
    UINT8 Len;
    UINT8 CountryCode[3];
    UINT8 DomainEntry[100];
} PACK_END IEEEtypes_COUNTRY_IE_t;
```

where:

Parameter	Type	Description
ElemId	IEEEtypes_ElementId_t	Country Set to 0x07.
Len	UINT8	Length of element structure DomainEntry is variable sized.
CountryCode[3]	UINT8	3-byte country code
DomainEntry[100]	UINT8	List of channel information for a particular country

2.7.9.11 IEEEtypes_PowerConstraintElement_t

```
typedef PACK_START struct IEEEtypes_PowerConstraintElement_t
{
    IEEEtypes_ElementId_e ElementId;
    IEEEtypes_Len_t Len;
    SINT8 powerConstraint;
} PACK_END IEEEtypes_PowerConstraintElement_t;
```

2.7.10 HostCmd_CMD_SET_APMODE

Firmware command for setting AP mode.

```
typedef PACK_START struct tagHostCmd_FW_SET_APMODE
{
    FWCmdHdr CmdHdr;
    UINT8 ApMode;
} PACK_END HostCmd_FW_SET_APMODE, *PHostCmd_FW_SET_APMODE;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
ApMode	UINT8	AP mode 0x00 = reserved, set to 0 0x01 = 802.11b mode 0x02 = 802.11g mode 0x03 = 802.11g/b modes 0x04 = draft-802.11n mode 0x05 = 802.11b and draft-802.11n modes 0x06 = 802.11g and draft-802.11n modes 0x07 = 802.11g/b and draft-802.11n modes 0x08 = 802.11a mode 0x09 to 0x0B = reserved, set to 0 0x0C = 802.11a and draft-802.11n modes 0x0D to 0x0F = reserved, set to 0

2.7.11 HostCmd_CMD_SET_COUNTRY_CODE

Firmware command for setting country code.

```
typedef PACK_START struct _HostCmd_SET_COUNTRY_INFO
{
    FWCmdHdr CmdHdr;
    UINT32 Action;
    DomainCountryInfo DomainInfo;
} PACK_END HostCmd_SET_COUNTRY_INFO, *PHostCmd_SET_COUNTRY_INFO;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT32	Action 0x00000000 = not set 0x00000001 = set
DomainInfo	DomainCountryInfo	See Section 2.7.11.1, DomainCountryInfo, on page 47.

2.7.11.1 DomainCountryInfo

```
typedef PACK_START struct _DomainCountryInfo
{
    UINT8 CountryString[3];
    UINT8 GChannelLen;
    DomainChannelEntry DomainEntryG[1]; /** Assume only 1 G zone **/
    UINT8 AChannelLen;
    DomainChannelEntry DomainEntryA[20]; /** Assume max of 5 A zone **/
} PACK_END DomainCountryInfo;

typedef PACK_START struct _DomainChannelEntry
{
    UINT8 FirstChannelNo;
    UINT8 NoofChannel;
    UINT8 MaxTransmitPw;
} PACK_END DomainCountryEntry;
```

2.7.12 HostCmd_CMD_SET_N_PROTECT_FLAGS

Firmware command for setting draft-802.11n protection flag.

```
typedef PACK_START struct tagHostCmd_FW_SET_N_PROTECT_FLAG
{
    FWCmdHdr CmdHdr;
    UINT32 NProtectFlag;
} PACK_END HostCmd_FW_SET_N_PROTECT_FLAG, *PHostCmd_FW_SET_N_PROTECT_FLAG;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
NProtectFlag	UINT32	Draft-802.11n protection flag 0x00000000 = disabled 0x00000001 = enabled

2.7.13 HostCmd_CMD_SET_N_PROTECT_OPMODE

Firmware command for setting draft-802.11n protection mode.

```
typedef PACK_START struct tagHostCmd_FW_SET_N_PROTECT_OPMODE
{
    FWCmdHdr CmdHdr;
    UINT8 NProtectOpMode;
} PACK_END HostCmd_FW_SET_N_PROTECT_OPMODE, *PHostCmd_FW_SET_N_PROTECT_OPMODE;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
NProtectOpMode	UINT8	Draft-802.11n protection mode 0x00 = disabled 0x01 = enabled

2.7.14 HostCmd_CMD_SET_WSC_IE

Firmware command for setting Wi-Fi Simple configuration (WSC) information element. This command is not used for hostFormBeacon mode.

```
#define WSC_BEACON_IE 0
#define WSC_PROBE_RESP_IE 1
typedef PACK_START struct _HostCmd_SET_WSC_IE
{
    FWCmdHdr CmdHdr;
    UINT16 ieType;
    WSC_COMB_IE_t wscIE;
} PACK_END HostCmd_SET_WSC_IE, *PHostCmd_SET_WSC_IE;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
ieType	UINT16	Type 0x0000 = beacon 0x0001 = probe response
wscIE	WSC_COMB_IE_t	See Section 2.7.14.1, WSC_COMB_IE_t, on page 49

2.7.14.1 WSC_COMB_IE_t

```
typedef union
{
    WSC_BeaconIE_t beaconIE;
    IEEEtypes_WSC_ProbeRespIE_t probeRespIE;
} PACK WSC_COMB_IE_t;
```

where:

Parameter	Type	Description
beaconIE	WSC_BeaconIE_t	Beacon information element as defined in Section 2.7.14.2, WSC_BeaconIE_t, on page 49
wscIE	IEEEtypes_WSC _ProbeRespIE_t	Probe response information element as defined in Section 2.7.14.3, IEEEtypes_WSC_ProbeRespIE_t, on page 49

2.7.14.2 WSC_BeaconIE_t

```
WSC_BEACON_IE_MAX_LENGTH : 68
WSC_OUI_LENGTH : 4
typedef struct
{
    UINT8 ElementId;
    UINT8 Len;
    UINT8 OUI[WSC_OUI_LENGTH];
    UINT8 WSCData[WSC_BEACON_IE_MAX_LENGTH];
} PACK WSC_BeaconIE_t;
```

2.7.14.3 IEEEtypes_WSC_ProbeRespIE_t

```
WSC_PROBERESP_IE_MAX_LENGTH : 251
WSC_OUI_LENGTH : 4
typedef struct
{
    UINT8 ElementId;
    UINT8 Len;
    UINT8 OUI[WSC_OUI_LENGTH];
    UINT8 WSCData[WSC_PROBERESP_IE_MAX_LENGTH];
} PACK IEEEtypes_WSC_BeaconIE_t;
```

2.8 Station Database and Security Attributes

2.8.1 HostCmd_CMD_SET_AID

Firmware command for setting AP information for client mode.

```
#define RATE_INDEX_MAX_ARRAY 14
typedef PACK_START struct tagHostCmd_FW_SET_AID
{
    FWCmdHdr CmdHdr;
    UINT16 AssocID;
    UINT8 MacAddr[6];
    UINT32 GProtection;
    UINT8 ApRates[RATE_INDEX_MAX_ARRAY];
} PACK_END HostCmd_FW_SET_AID, *PHostCmd_FW_SET_AID;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
AssocID	UINT16	Not used, set to 0
MacAddr[6]	UINT8	AP's MAC address, standard 6-byte MAC address
GProtection	UINT32	Not used, set to 0
ApRates	UINT8	Not used, set to 0

2.8.2 HostCmd_CMD_SET_NEW_STN

Firmware command for adding new station to firmware station database.

```
typedef PACK_START struct tagHostCmd_FW_SET_NEW_STN
{
    FWCmdHdr CmdHdr;
    UINT16 AID;
    UINT8 MacAddr[6];
    UINT16 StnId;
    UINT16 ActionType;
    UINT16 Reserved;
    PeerInfo_t PeerInfo;
    Qos_WmeInfo_Info_t Qosinfo;
    UINT8 isQosSta;
    UINT32 FwStaPtr;
} PACK_END HostCmd_FW_SET_NEW_STN, *PHostCmd_FW_SET_NEW_STN;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
AID	UINT16	Station AID
MacAddr[6]	UINT8	Standard 6-byte MAC address
StnId	UINT16	Station ID
ActionType	UINT16	Action 0x0000 = add station 0x0001 = modify station 0x0002 = remove station
Reserved	UINT16	Reserved, set to 0
PeerInfo	PeerInfo_t	See Section 2.8.2.1, PeerInfo_t, on page 51
Qosinfo	Qos_WmeInfo_Info_t	See Section 2.8.2.7, Qos_WmeInfo_Info_t, on page 54
isQosSta	UINT8	Station QoS 0x00 = station does not support QoS 0x01 = station supports QoS
FwStaPtr	UINT32	Firmware station pointer For add station: Zero: add station failure, station not added to firmware database. Non-zero: Add station success, firmware pointer to station database entry. For remove station: Zero: remove station success, station removed from firmware database. Non-zero: remove station failure, station not removed from firmware database. For modify station: Zero: station modify failure, station not in database. Non-zero: station modify success, firmware pointer to station database entry.

2.8.2.1 PeerInfo_t

Peer information of station.

```
typedef struct
{
    UINT32 LegacyRateBitMap;
    UINT32 HTRateBitMap;
    IEEEtypes_CapInfo_t CapInfo;
    IEEEtypes_HT_Cap_t HTCapabilitiesInfo;
    UINT8 MacHTParamInfo;
    UINT8 Rev;
    IEEEtypes_Add_HT_INFO_t AddHtInfo;
} PACK_END PeerInfo_t;
```

where:

Parameter	Type	Description
LegacyRateBitMap	UINT32	Bits correspond to legacy supported rates 1, 2, 5.5, 11, 6, 9, 12, 18, 24, 36, 48, 54 Mbps corresponds to bits 0 to 12.
HTRateBitMap	UINT32	Bits correspond to MCS0 to MCS15+ rates
CapInfo	IEEEtypes_CapInfo_t	See Section 2.7.9.5, IEEEtypes_CapInfo_t, on page 41
HTCapabilitiesInfo	IEEEtypes_HT_Cap_t	See Section 2.8.2.2, IEEEtypes_HT_Cap_t, on page 52
MacHTParamInfo	UINT8	Bits[7:5]: Reserved, set to 0 Bits[4:2]: Minimum MPDU start spacing 0x0 = no restriction 0x1 = 0.25 µs 0x2 = 0.5 µs 0x3 = 1 µs 0x4 = 2 µs 0x5 = 4 µs 0x6 = 8 µs 0x7 = 16 µs Bits[1:0]: Maximum AMPDU length 0x0 = 8K 0x1 = 16K 0x2 = 32K 0x3 = 64K
Rev	UINT8	Revision 0x00 = non-Marvell station 0x01 = Marvell station
AddHtInfo	IEEEtypes_Add_HT_INFO_t	See Section 2.8.2.3, IEEEtypes_Add_HT_INFO_t, on page 53.

2.8.2.2 IEEEtypes_HT_Cap_t

High throughput capabilities parameter definition.

```
typedef struct IEEEtypes_HT_Cap_t
{
    UINT16 AdvCoding: 1;
    UINT16 SupChanWidth: 1;
    UINT16 MIMOPwSave: 2;
    UINT16 GreenField: 1;
    UINT16 SGI20MHz: 1;
    UINT16 SGI40MHz: 1;
    UINT16 TxSTBC: 1
    UINT16 RxSTBC: 2;
    UINT16 DelayedBA: 1;
    UINT16 MaxAMSDUSize: 1;
    UINT16 DssCck40MHz: 1;
    UINT16 PSMP: 1;
    UINT16 STBCCtrlFrm: 1;
    UINT16 LSIGTxopProc: 1;
} PACK_END IEEEtypes_HT_Cap_t;
```

2.8.2.3 IEEEtypes_Add_HT_INFO_t

High throughput information parameter definition.

```
typedef struct IEEEtypes_Add_HT_INFO_t
{
    UINT16 ControlChan;
    IEEEtypes_Add_HT_Chan_t AddChan;
    IEEEtypes_Add_HT_OpMode_t OpMode;
    IEEEtypes_Add_HT_STBC_t stbc;
} PACK_END IEEEtypes_Add_HT_INFO_t;
```

where:

Parameter	Type	Description
ControlChan	UINT16	Primary channel
AddChan	IEEEtypesAdd_HT_Chan_t	Additional HT channel information (see Section 2.8.2.4, IEEEtypesAdd_HT_Chan_t, on page 53)
OpMode	IEEEtypesAdd_HT_OpMode_t	Operation mode (see Section 2.8.2.5, IEEEtypesAdd_HT_OpMode_t, on page 54)
stbc	IEEEtypes_Add_HT_STBC_t	Space time block coding information (see Section 2.8.2.6, IEEEtypes_Add_HT_STBC_t, on page 54)

2.8.2.4 IEEEtypesAdd_HT_Chan_t

Additional high throughput channel information.

```
typedef struct IEEEtypes_Add_HT_Chan_t
{
    UINT8 ExtChanOffset: 2;
    UINT8 RcmTxWidthSet: 1;
    UINT8 RIFSMode: 1;
    UINT8 CtrledAccssOnly: 1;
    UINT8 SrvcIntvlGran: 3
} PACK_END IEEEtypes_Add_HT_Chan_t;
```

2.8.2.5 IEEEtypesAdd_HT_OpMode_t

High throughput operation mode.

```
typedef struct IEEEtypes_Add_HT_OpMode_t
{
    UINT16 OpMode: 2;
    UINT16 Rsrv: 14;
} PACK_END IEEEtypes_Add_HT_OpMode_t;
```

2.8.2.6 IEEEtypes_Add_HT_STBC_t

High throughput space time block code information.

```
typedef struct IEEEtypes_Add_HT_STBC_t
{
    UINT16 BscSTBC: 7;
    UINT16 DualSTBCProc: 1;
    UINT16 ScdBcn: 1;
    UINT16 LSIGTxopProcFullSup: 1;
    UINT16 PCOActive: 1;
    UINT16 PCOPhase: 1;
    UINT16 Rsrv: 4;
} PACK_END IEEEtypes_Add_HT_STBC_t;
```

2.8.2.7 Qos_WmeInfo_t

```
typedef struct Qos_WmeInfo_Info_t
{
    UINT8 Uapsd_ac_vo: 1;
    UINT8 Uapsd_ac_vi: 1;
    UINT8 Uapsd_ac_bk: 1;
    UINT8 Uapsd_ac_be: 1;
    UINT8 Reserved: 1;
    UINT8 Max_Sp: 2;
    UINT8 Reserved2: 1;
} PACK_END Qos_WmeInfo_Info_t;
```

where:

Parameter	Type	Description
Upasd_ac_vo	UINT8:1	Bit[0]: Power save mode for voice
Upasd_ac_vi	UINT8:1	Bit[1]: Power save mode for video
Upasd_ac_bk	UINT8:1	Bit[2]: Power save mode for background
Upasd_ac_be	UINT8:1	Bit[3]: Power save mode for best effort

Parameter	Type	Description
Reserved	UINT8:1	Bit[4]: Reserved, set to 0
Max_Sp	UINT8:2	Maximum number of buffered MSDUs or MPDUs that the AP may deliver to the station during any power save trigger Only applicable if at least one of the power save bits above is 1. Bits[6:5]: 0x0 = AP may deliver all packets 0x1 = AP may deliver maximum of 2 packets per trigger 0x2 = AP may deliver maximum of 4 packets per trigger 0x3 = AP may deliver maximum of 8 packets per trigger
Reserved2	UINT8:1	Bit[7]: Reserved, set to 0

2.8.3 HostCmd_CMD_UPDATE_ENCRYPTION

Firmware command to enable encryption and set keys with two structures.

- HostCmd_UPDATE_ENCRYPTION—for enabling encryption
- HostCmd_UPDATE_ENCRYPTION_SET_KEY—for setting encryption keys

2.8.3.1 HostCmd_UPDATE_ENCRYPTION

```
typedef PACK_START struct tagHostCmd_FW_ENCRYPTION
{
    FWCmdHdr CmdHdr;
    UINT32 ActionType;
    UINT32 DataLength;
    UINT8 macaddr[6];
    UINT8 ActionData[1];
} PACK_END HostCmd_FW_UPDATE_ENCRYPTION, *PHostCmd_FW_UPDATE_ENCRYPTION;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
ActionType	UINT32	Action type 0x00000000 = enable encryption
DataLength	UINT32	Length of encryption data structure
macaddr[6]	UINT8	MAC address of station in database
ActionData[1]	UINT8	Action data 0x0000 = WEP 0x0001 = disable, no encryption 0x0004 = TKIP 0x0008 = AES 0x0007 = mixed mode

2.8.3.2 HostCmd_UPDATE_ENCRYPTION_SET_KEY

```
MAX_ENCR_KEY_LENGTH 16 // maximum 128 bits-depends on type
MIC_KEY_LENGTH 8 // size of Tx/Rx MIC key-8 bytes
typedef PACK_START struct tagHostCmd_FW_ENCRYPTION_SET_KEY
{
    FWCmdHdr CmdHdr;
    UINT32 ActionType;
    UINT32 DataLength;
    KEY_PARAM_SET KeyParam;
} PACK_END HostCmd_FW_UPDATE_ENCRYPTION_SET_KEY,
*PHostCmd_FW_UPDATE_ENCRYPTION_SET_KEY;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
ActionType	UINT32\	Action type 0x00000001 = set key 0x00000002 = remove key 0x00000003 = set group key
DataLength	UINT32	Length of KeyParam
KeyParam	KEY_PARAM_SET	See Section 2.8.3.3, KEY_PARAM_SET, on page 56

2.8.3.3 KEY_PARAM_SET

Encryption key definitions.

```
typedef PACK_START struct _KEY_PARAM_SET
{
    UINT16 Length;
    UINT16 KeyTypeId;
    UINT32 KeyInfo;
    UINT32 KeyIndex;
    UINT16 KeyLen;
    PACK_STRUCT union
    {
        WEP_TYPE_KEY WepKey;
        TKIP_TYPE_KEY TkipKey;
        AES_TYPE_KEY AesKey;
    } PACK_END Key;
    UINT8 Macaddr[6];
} PACK_END KEY_PARAM_SET, *PKEY_PARAM_SET;
```

where:

Parameter	Type	Description
Length	UINT16	Total length of this structure (key is variable size array)
KeyTypeId	UINT16	Key type 0x0000 = WEP 0x0001 = TKIP 0x0002 = AES

KeyInfo	UINT32	Key flags
		Bit[8] = use Tx/Rx MIC keys in TKIP_TYPE_KEY structure
		Bit[7] = WEP Tx key
		Bit[6] = use sequence counters in TKIP_TYPE_KEY structure
		Bits[5:4] = reserved, set to 0
		Bit[3] = pairwise key
		Bit[2] = Tx group key
		Bit[1] = Rx group key
		Bit[0] = reserved, set to 0

KeyIndex	UINT32	For WEP only—actual key index
KeyLen	UINT16	Size of key
Key	--	Union—key material (variable size array) **Item** / **Description** WepKey: WEP_TYPE_KEY (see Section 2.8.3.4, WEP_TYPE_KEY, on page 57) TkipKey: TKIP_TYPE_KEY (see Section 2.8.3.5, TKIP_TYPE_KEY, on page 58) AesKey: AES_TYPE_KEY (see Section 2.8.3.6, AES_TYPE_KEY, on page 59)
Macaddr[6[	UINT8	MAC address of station

2.8.3.4 WEP_TYPE_KEY

WEP key material definition.

```
typedef PACK_START struct _WEP_TYPE_KEY
{
    UINT8 KeyMaterial[MAX_ENCR_KEY_LENGTH];
} PACK_END WEP_TYPE_KEY, *PWEP_TYPE_KEY;
```

where:

Parameter	Type	Description
KeyMaterial	UINT8	An array of MAX_ENCR_KEY_LENGTH bytes **NOTE:** 152-bit WEP keys not supported.

2.8.3.5 TKIP_TYPE_KEY

TKIP key material definition.

TKIP sequence counter is 24 bits. Incremented on each fragment MPDU.

```
typedef PACK_START struct _TKIP_TYPE_KEY
{
    UINT8 KeyMaterial[MAX_ENCR_KEY_LENGTH];
    UINT8 TkipTxMicKey[MAX_KEY_LENGTH];
    UINT8 TkipRxMicKey[MAX_KEY_LENGTH];
    ENCR_TKIPSEQCNT TkipRsc;
    ENCR_TKIPSEQCNT TkipTsc;
} PACK_END TKIP_TYPE_KEY, *PTKIP_TYPE_KEY;
```

where:

Parameter	Type	Description
KeyMaterial	UINT8	Key of size up to MAX_ENCR_KEY_LENGTH bytes
TkipTxMicKey	UINT8	Tx MIC key used if KeyInfo flag is set
TkipRxMicKey	UINT8	Rx MIC key used if KeyInfo flag is set
TkipRsc	ENCR_TKIPSEQCNT	Rx sequence counter used if KeyInfo flag is set
TkipTsc	ENCR_TKIPSEQCNT	Tx sequence counter used if KeyInfo flag is set

```
typedef PACK_START struct tagENCR_TKIPSEQCNT
{
    UINT16 low;
    UINT32 high;
} PACK_END ENCR_TKIPSEQCNT, *PENCR_TKIPSEQCNT;
```

where:

Parameter	Type	Description
low	UINT16	Lower order sequence number
high	UINT32	Higher order sequence number

2.8.3.6 AES_TYPE_KEY

AES-CCMP key material definition.

```
typedef PACK_START struct _AES_TYPE_KEY
{
    UINT8 KeyMaterial[MAX_ENCR_KEY_LENGTH];
} PACK_END AES_TYPE_KEY, *PAES_TYPE_KEY;
```

where:

Parameter	Type	Description
KeyMaterial	UINT8	Key of size up to MAX_ENCR_KEY_LENGTH bytes

2.8.4 HostCmd_CMD_DWDS_ENABLE

Firmware command to enable/disable Dynamic Wireless Distribution System (DWDS) mode.

```
typedef PACK_START struct tagHostCmd_DWDS_ENABLE
{
    FWCmdHdr CmdHdr;
    UINT32 Enable;
} PACK_END HostCmd_DWDS_ENABLE, *PHostCmd_DWDS_ENABLE;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Enable	UINT32	Enable DWDS 0x00000000 = disable 0x00000001 = enable

2.9 QoS and AMPDU Attributes

2.9.1 HostCmd_CMD_SET_EDCA_PARAMS

Firmware command to set queue parameters for QoS.

```
typedef struct tagHostCmd_FW_SET_EDCA_PARAMS
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
    UINT16 TxOP;
    UINT32 CWMax;
    UINT32 CWMin;
    UINT8 AIFSN;
    UINT8 TxQNum;
} HostCmd_FW_SET_EDCA_PARAMS, *PHostCmd_FW_SET_EDCA_PARAMS;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0000 = get all 0x0001 = set CWMin/CWMax 0x0002 = set TxOP 0x0004 = set AIFSN
TxOP	UINT16	Tx OP In units of 32 µs.
CWMax	UINT32	Contention window maximum 0 to 1023
CWMin	UINT32	Contention window minimum 0 to 1023
AIFSN	UINT8	Arbitrary inter-frame spacing number in slot times
TxQNum	UINT8	Tx queue number

2.9.2 HostCmd_CMD_SET_WMM_MODE

Firmware command to enable/disable Wi-Fi Multimedia (WMM) mode.

```
typedef struct tagHostCmd_FW_SetWMMMode
{
    FWCmdHdr CmdHdr;
    UINT16 Action;
} HostCmd_FW_SetWMMMode, *PHostCmd_FW_SetWMMMode;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Action	UINT16	Action 0x0000 = disable 0x0001 = enable

2.9.3 HostCmd_CMD_BASTREAM

Firmware command to create/destroy block acknowledge stream for AMPDU.

```
typedef PACK_START struct tagHostCmd_FW_BASTREAM
{
    FWCmdHdr CmdHdr;
    UINT32 ActionType;
    PACK_START union
    {
        BASTREAM_CREATE_STREAM CreateParams;
        BASTREAM_UPDATE_STREAM UpdtSeqNum;
        BASTREAM_STREAM_INFO DestroyParams;
        BASTREAM_STREAM_INFO FlushParams;
    } PACK_END BaInfo;
} PACK_END HostCmd_FW_BASTREAM, *PHostCmd_FW_BASTREAM;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header

Parameter	Type	Description
ActionType	UINT32	Action type 0x00000000 = create stream 0x00000001 = update stream 0x00000002 = destroy stream 0x00000003 = flush stream 0x00000004 = check create stream
BaInfo	--	Union parameter depends on ActionType: (see nested table below)

Item	Description
CreateParams	Create stream BASTREAM_CREATE_STREAM (see Section 2.9.3.1, BASTREAM_CREATE_STREAM, on page 62)
UpdtSeqNum	Update starting/new sequence number BASTREAM_UPDATE_STREAM (see Section 2.9.3.3, BASTREAM_UPDATE_STREAM, on page 64)
DestroyParams	Destroy existing stream parameters BASTREAM_STREAM_INFO (see Section 2.9.3.4, BASTREAM_STREAM_INFO, on page 65)
FlushParams	Flush parameters BASTREAM_STREMA_INFO (see Section 2.9.3.4, BASTREAM_STREAM_INFO, on page 65)

2.9.3.1 BASTREAM_CREATE_STREAM

Parameter for block acknowledge create, destroy, and update.

```
typedef PACK_START struct tagCreateBaParams
{
    BASTREAM_FLAGS Flags;
    UINT32 IdleThrs;
    UINT32 BarThrs;
    UINT32 WindowSize;
    UINT8 PeerMacAddr[6];
    UINT8 DialogToken;
    UINT8 Tid
    UINT8 QueueId;
    UINT8 ParamInfo;
    BASTREAM_CONTEXT FwBaContext;
    UINT8 ResetrSeqNo;
    UINT16 CurrentSeq;
    UINT8 StaSrcMacAddr[6];
} PACK_END BASTREAM_CREATE_STREAM;
```

where:

Parameter	Type	Description
Flags	BASTREAM _FLAGS	BA creation flags See Section 2.9.3.2, BASTREAM_FLAGS, on page 64.
IdleThrs	UINT32	Idle threshold
BarThrs	UINT32	Block acknowledge transmit threshold
WindowSize	UINT32	Receiver window size
PeerMacAddr[6]	UINT8	MAC address of the BA partner
DialogToken	UINT8	Dialog token
Tid	UINT8	TID for the traffic stream in this BA
QueueId	UINT8	Shared memory queue ID
ParamInfo	UINT8	Parameter information

Bit	Description
[7:5]	Reserved, set to 0
[4:2]	Minimum MPDU start spacing 0x0 = no restriction 0x1 = 0.25 µs 0x2 = 0.5 µs 0x3 = 1 µs 0x4 = 2 µs 0x5 = 4 µs 0x6 = 8 µs 0x7 = 16 µs
[1:0]	Maximum AMPDU length that STA can receive This field is an integer in the range 0 to 3. The length defined by this field is equal to $2^{(13 + \text{Maximum AMPDU Length})}$ - 1 octets.

Parameter	Type	Description
FwBaContext	BASTREAM _CONTEXT	See Section 2.9.3.5, BASTREAM_CONTEXT, on page 65
ResetSeqNo	UINT8	Reset sequence number 0x00 = do not reset sequence number 0x01 = reset sequence number
CurrentSeq	UINT16	Sequence number to start stream with
StaSrcMacAddr[6]	UINT8	Used for proxy station mode only MAC address of proxy station.

2.9.3.2 BASTREAM_FLAGS

Block acknowledge stream flags.

```
typedef PACK_START struct tagBAStreamFlags
{
    UINT32 BaType: 1;
    UINT32 BaDirection: 3;
    UINT32 Reserved: 24;
} PACK_END BASTREAM_FLAGS;
```

where:

Parameter	Type	Description
BaType	UINT32:1	Bit[0]: BA type 0 = delayed block acknowledge 1 = immediate block acknowledge
BaDirection	UINT32:3	Bits[3:1]: BA direction 0x0 = upstream 0x1 = downstream 0x2 = reserved, set to 0 0x3 = bidirectional
Reserved	UINT32:24	Bits[27:4]: Reserved, set to 0
NOTE: Bits[31:28] are not used, set to 0.		

2.9.3.3 BASTREAM_UPDATE_STREAM

Transmit sequence number information.

```
typedef PACK_START struct tagBaUpdateSeqNum
{
    BASTREAM_FLAGS Flags;
    BASTREAM_CONTEXT FwBaContext;
    UINT16 BaSeqNum;
} PACK_END BASTREAM_UPDATE_STREAM;
```

where:

Parameter	Type	Description
Flags	BASTREAM_FLAGS	See Section 2.9.3.2, BASTREAM_FLAGS, on page 64
FwBaContext	BASTREAM_CONTEXT	See Section 2.9.3.5, BASTREAM_CONTEXT, on page 65
BaSeqNum	UINT16	New sequence number for this block acknowledge stream

2.9.3.4 BASTREAM_STREAM_INFO

Transmit block acknowledge stream information.

```
typedef PACK_START struct tagBaStreamContext
{
    BASTREAM_FLAGS Flags;
    BASTREAM_CONTEXT FwBaContext;
} PACK_END BASTREAM_STREAM_INFO;
```

where:

Parameter	Type	Description
Flags	BASTREAM_FLAGS	See Section 2.9.3.2, BASTREAM_FLAGS, on page 64
FwBaContext	BASTREAM_CONTEXT	See Section 2.9.3.5, BASTREAM_CONTEXT, on page 65

2.9.3.5 BASTREAM_CONTEXT

Firmware reference pointer for block acknowledge stream.

```
typedef PACK_START struct tagBAContext
{
    UINT32 Context;
} PACK_END BASTREAM_CONTEXT;
```

where:

Parameter	Type	Description
Context	UINT32	Stream identifier (0,1) hardware 2-7 software block acknowledge streams

2.9.4 HostCmd_CMD_SET_RIFS

Firmware command for setting Reduced Interfame Space (RIFS).

```
typedef PACK_START struct tagHostCmd_FW_SET_RIFS
{
    FWCmdHdr CmdHdr;
    UINT8 QNum;
} PACK_END HostCmd_FW_SET_RIFS, *PHostCmd_FW_SET_RIFS;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
QNum	UINT8	Enable RIFS for queue number

2.9.5 HostCmd_CMD_SET_OPTIMIZATION_LEVEL

Firmware command for setting optimization level.

```
typedef PACK_START struct tagHostCmd_FW_SET_OPTIMIZATION_LEVEL
{
    FWCmdHdr CmdHdr;
    UINT8 Optlevel;
} PACK_END HostCmd_FW_SET_OPTIMIZATION_LEVEL,
*PHostCmd_FW_SET_OPTIMIZATION_LEVEL;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
OptLevel	UINT8	Optimization level 0x00 = normal operation 0x01 = high performance (or burst mode)

2.9.6 HostCmd_CMD_GET_SEQNO

Firmware command for getting the sequence number of station.

```
typedef PACK_START struct _HostCmd_GET_SEQNO
{
    FWCmdHdr CmdHdr;
    UINT8 MacAddr[6];
    UINT8 TID;
    UINT16 SeqNo;
    UINT8 reserved
} PACK_END HostCmd_GET_SEQNO, *PHostCmd_GET_SEQNO;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
MacAddr[6]	UINT8	MAC address of station
TID	UINT8	TID of traffic
SeqNo	UINT16	Sequence number
reserved	UINT8	Reserved, set to 0

2.9.7 HostCmd_CMD_CFEND_ENABLE

Firmware command to enable/disable CFEND transmit at the end of EDCA TXOP.

```
typedef PACK_START struct _HostCmd_CFEND_ENABLE
{
    FWCmdHdr CmdHdr;
    UINT32 Enable
} PACK_END HostCmd_CFEND_ENABLE, *PHostCmd_CFEND_ENABLE;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Enable	UINT32	Enable CFEND transmit 0x00000000 = disable 0x00000001 = enable

2.10 AP PowerSave Attributes

2.10.1 HostCmd_CMD_SET_MIMOPSHT

Firmware command for setting MIMO powersave high throughput parameter.

```
typedef PACK_START struct tagHostCmd_FW_SET_MIMOPSHT
{
    FWCmdHdr CmdHdr;
    UINT8 Addr[6];
    UINT8 Enable;
    UINT8 Mode;
} PACK_END HostCmd_FW_SET_MIMOPSHT, *PHostCmd_FW_SET_MIMOPSHT;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Addr[6]	UINT8	MAC address of station
Enable	UINT8	Enable MIMO power save
Mode	UINT8	Mode 0x00 = MIMO power save static 0x01 = MIMO power save dynamic

2.10.2 HostCmd_CMD_SET_POWERSAVESTATION

Firmware command for setting the number of stations in power save mode.

```
typedef PACK_START struct tagHostCmd_SET_POWERSAVESTATION
{
    FWCmdHdr CmdHdr;
    UINT8 NumberofPowersave;
    UINT8 reserved;
} PACK_END HostCmd_SET_POWERSAVESTATION, *PHostCmd_SET_POWERSAVESTATION;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
NumberofPowersave	UINT8	Number of stations in power save
reserved	UINT8	Reserved, set to 0

2.10.3 HostCmd_CMD_SET_TIM

Firmware command for setting the TIM bit of a specified station.

```
typedef PACK_START struct tagHostCmd_SET_TIM
{
   FWCmdHdr CmdHdr;
   UINT16 Aid;
   UINT32 Set;
   UINT8 reserved
} PACK_END HostCmd_SET_TIM, *PHostCmd_SET_TIM;
```

where:

Parameter	Type	Description
CmdHdr	FWCmdHdr	Standard firmware command header
Aid	UINT16	Association ID of station
Set	UINT32	Set TIM 0x00000000 = clear station TIM 0x00000001 = set station TIM
reserved	UINT8	Reserved, set to 0

THIS PAGE INTENTIONALLY LEFT BLANK

A Acronyms and Abberivations

Table 2: Acronyms and Abbreivations

Term	Definition
AES	Advanced Encryption Standard
AID	Association Identifier
AMPDU	Aggregate MAC Protocol Data Unit
APSDU	Aggregate MAC Service Data Unit
AP	Access Point
APSD	Automatic Power Save Delivery
BA	Block Acknowledgement
BSS	Basic Service Set
CCMP	Counter mode with Cipher Block Chaining Message protocol
CFEND	End of Contention-Free Period
CS	Card Select
DFS	Dynamic Frequency Selection
DSSS	Direct Sequence Spread Spectrum
DTIM	Delivery Traffic Indication Message
DWDS	Dynamic Wireless Distribution System
ERP	Extended Rate PHY
ESS	Extended Service Set
GF	Greenfield
HT	High Throughput
IBSS	Independent Basic Service Set (“Ad-Hoc”)
ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
IQ	In-phase/Quadrature
LNA	Low Noise Amplifier
MAC	Medium/Media Access Controller
MIC	Message Integrity Code
MIMO	Multiple Input, Multiple Output
MPDU	MAC Protocol Data Unit
MSDU	MAC Service Data Unit
NetBSD	Net–Berkeley Software Distribution

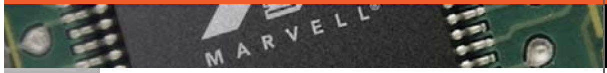
Table 2: Acronyms and Abbreviations (Continued)

Term	Definition
OFDM	Orthogonal Frequency Division Multiplexing
PBCC	Packet Binary Convolution Code
PHY	Physical Layer
QoS	Quality of Service
RIFS	Reduced Interframe Space
RF	Radio Frequency
RSSI	Received Signal Strength Indicator
RTS	Request to Send
Rx	Receive
SINT32	32-bit signed integer
SPI	Serial Peripheral Interface
SSID	Service Set ID A 32-character unique identifier attached to the header of packets sent over a WLAN that acts as a password when a mobile device tries to connect to the BSS.
TID	Traffic Stream Identifier
TIM	Traffic Information Map
TKIP	Temporal Key Integrity Protocol
Tx	Transmit
UINT16	16-bit unsigned integer
UINT32	32-bit unsigned integer
UINT8	8-bit unsigned integer
WCB	Wireless Control Block
WEP	Wired Equivalent Privacy
WLAN	Wireless Local Area Network
WMM	Wi-Fi Multimedia
WSC	Wi-Fi Simple Configuration

B Revision History

Table 3: Revision History

Document Type	Document Revision
Release	Rev. –
First release.	



Marvell Semiconductor, Inc.
5488 Marvell Lane
Santa Clara, CA 95054, USA

Tel: 1.408.222.2500
Fax: 1.408.752.9028

www.marvell.com

Marvell. Moving Forward Faster