

# WM-RP-0xS

## サンプルプログラム解説 (AP-RX111-0A)

第1版 2013年12月13日

### 目次

1. 概要 .....	1
1. 1 概要 .....	1
1. 2 動作環境 .....	2
1. 3 ネットワーク構成イメージ図 .....	3
1. 4 動作モード .....	4
1. 5 開発環境について .....	6
1. 6 ワークスペースについて .....	6
2. サンプルプログラムの構成 .....	7
2. 1 フォルダ構成 .....	7
2. 2 ファイル構成 .....	8
3. TCP/IP 通信サンプルプログラム .....	12
3. 1 ビルド・デバッグ方法 .....	12
3. 2 動作説明 (TCP/IP 通信) .....	13
3. 2. 1 サンプルプログラム概要 (TCP/IP 通信 アドホックモード) .....	13
3. 2. 2 サンプルプログラム概要 (TCP/IP 通信 インフラストラクチャモード) .....	15
3. 2. 3 TCP/IP 通信エコーバックサーバ動作 .....	16
3. 3 メモリマップ (TCP/IP 通信サンプルプログラム共通) .....	18
4. UDP 通信サンプルプログラム .....	19
4. 1 ビルド・デバッグ方法 (UDP 通信サンプルプログラム) .....	19
4. 2 動作説明 (UDP 通信) .....	20
4. 2. 1 サンプルプログラム概要 (UDP 通信 アドホックモード) .....	20
4. 2. 2 サンプルプログラム概要 (UDP 通信 インフラストラクチャモード) .....	22
4. 2. 3 UDP 通信エコーバックサーバ動作 .....	23
4. 3 メモリマップ (UDP 通信サンプルプログラム共通) .....	25

5. WM-RP-0XS 制御方法 .....	26
5. 1 概要 .....	26
5. 2 SPI .....	26
5. 2. 1 SPI 仕様 .....	26
5. 2. 2 SPI 通信の基本的な流れ .....	27
5. 2. 3 INTR 割り込み信号 .....	27
5. 2. 4 無線データ送信処理 .....	28
5. 3 Redpine Signals 社提供のライブラリ .....	29
5. 3. 1 各種変数等 .....	29
5. 3. 2 無線通信用の主なコマンドファイル .....	31
6. WM-RP-0XS 上の LED に関して .....	37
7. サンプルプログラムに関して .....	38
7. 1 ネットワークの設定 .....	38
7. 2 環境依存部（ハードウェア等） .....	41

## 1. 概要

### 1. 1 概要

本アプリケーションノートでは、弊社製 CPU ボード AP-RX111-0A 上で動作する、WM-RP-0xS 用サンプルプログラムについて解説します。

本サンプルプログラムの概要を以下に示します。

サンプルプログラム	ターゲットボード	動作内容
TCP/IP 通信サンプルプログラム (アドホックモード クリエイター)	・ AP-RX111-0A	・ TCP/IP アドホックモード (クリエイター) エコーバックサーバ ・ シリアル通信 ・ タイマ割り込み
TCP/IP 通信サンプルプログラム (アドホックモード ジョイナー)	・ AP-RX111-0A	・ TCP/IP アドホックモード (ジョイナー) エコーバックサーバ ・ シリアル通信 ・ タイマ割り込み
TCP/IP 通信サンプルプログラム (インフラストラクチャモード)	・ AP-RX111-0A	・ TCP/IP インフラストラクチャモード エコーバックサーバ ・ シリアル通信 ・ タイマ割り込み
UDP 通信サンプルプログラム (アドホックモード クリエイター)	・ AP-RX111-0A	・ UDP アドホックモード (クリエイター) エコーバックサーバ ・ シリアル通信 ・ タイマ割り込み
UDP 通信サンプルプログラム (アドホックモード ジョイナー)	・ AP-RX111-0A	・ UDP アドホックモード (ジョイナー) エコーバックサーバ ・ シリアル通信 ・ タイマ割り込み
UDP 通信サンプルプログラム (インフラストラクチャモード)	・ AP-RX111-0A	・ UDP インフラストラクチャモード エコーバックサーバ ・ シリアル通信 ・ タイマ割り込み

## 1. 2 動作環境

各サンプルプログラムの動作確認に必要な機器を以下に記します。

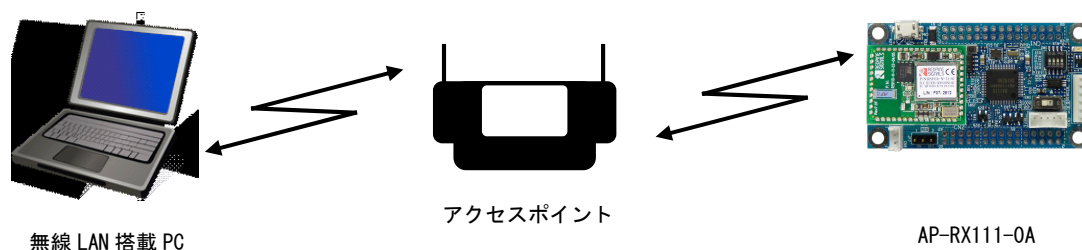
サンプルプログラム	動作確認に必要な機器
TCP/IP 通信サンプルプログラム (アドホックモード クリエイター)	・アドホック通信可能なホスト PC
TCP/IP 通信サンプルプログラム (アドホックモード ジョイナー)	・アドホック通信可能なホスト PC
TCP/IP 通信サンプルプログラム (インフラストラクチャモード)	・ネットワーク通信可能なホスト PC ・アクセスポイント
UDP 通信サンプルプログラム (アドホックモード クリエイター)	・アドホック通信可能なホスト PC
UDP 通信サンプルプログラム (アドホックモード ジョイナー)	・アドホック通信可能なホスト PC
UDP 通信サンプルプログラム (インフラストラクチャモード)	・ネットワーク通信可能なホスト PC ・アクセスポイント

### 1. 3 ネットワーク構成イメージ図

以下に、「インフラストラクチャ」と「アドホック」時のネットワーク構成イメージ図を示します。

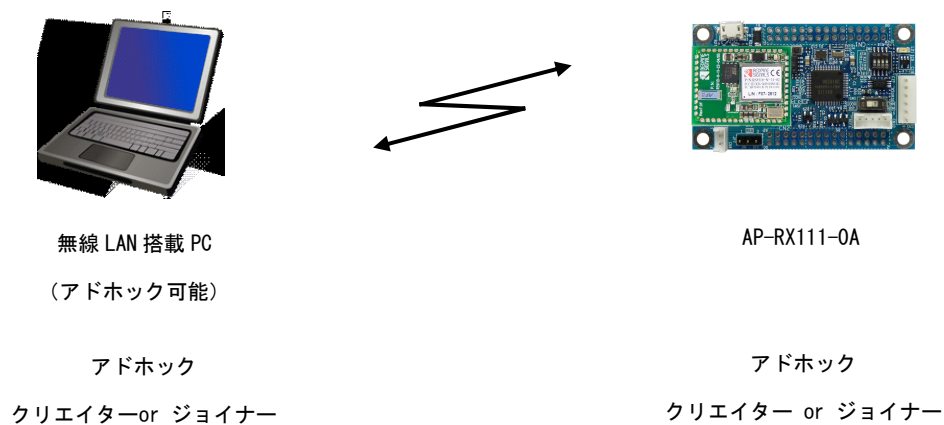
#### ・インフラストラクチャ

アクセスポイント経由で、無線通信を行います。



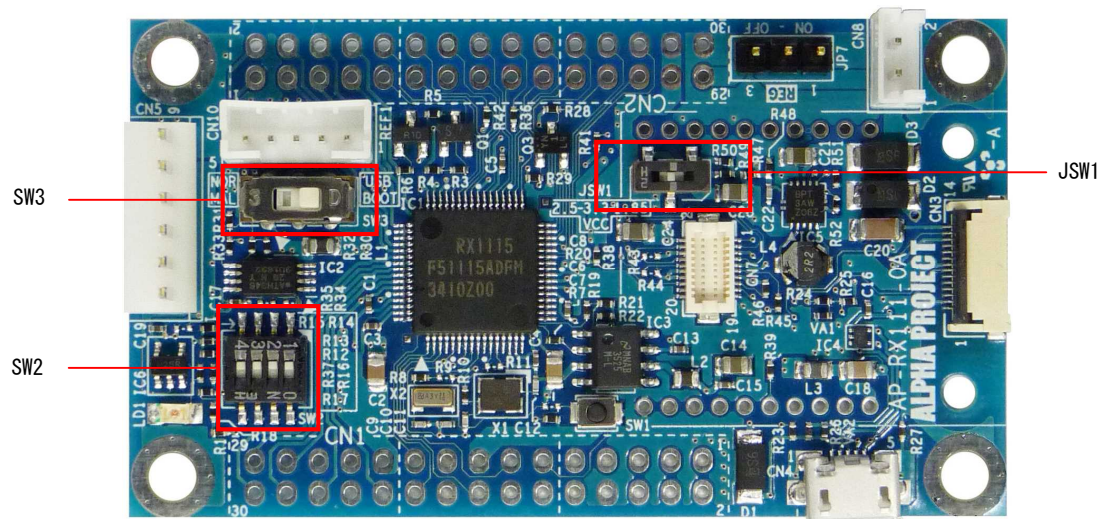
#### ・アドホック

アドホック通信では、クリエイター（親）となった機器に、ジョイナー（子）となった機器が接続して通信を行います。



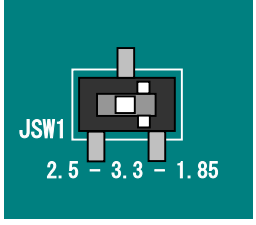
## 1. 4 動作モード

本サンプルプログラムは、AP-RX111-0Aで動作します。CPU動作モード、各メモリ設定は下記のようになっています。モードの設定方法等につきましては、「AP-RX111-0Aハードウェアマニュアル」をご覧ください。



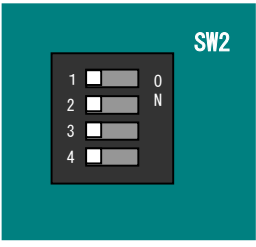
CPU ボードの設定を製品出荷時の状態とし、使用方法に合わせて以下の各スイッチの設定を行って下さい。

**・ JSW1**



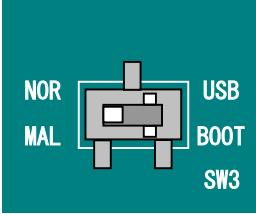
<JSW1 設定>  
VCC 出力値設定

**・ SW2**



<SW2 設定>  
通常時 1-4:OFF  
USB BOOT 時 1-2:ON  
3-4:OFF  
※左図は通常時のものです。

**・ SW3**



<SW3 設定>  
NORMAL 通常起動モード  
USB BOOT USB ブートモード

Fig 1.4-1 動作モード設定

## 1. 5 開発環境について

本サンプルプログラムは総合開発環境 CubeSuite+を用いて開発されております。

サンプルプログラムに対応する開発環境、コンパイラのバージョンは次のようになります。

開発環境	バージョン	コンパイラ名	バージョン
CubeSuite+	2.00.01	CC-RX	2.00.00.16

## 1. 6 ワークスペースについて

本サンプルプログラムの総合開発環境 CubeSuite+ワークスペースは次のフォルダに格納されています。

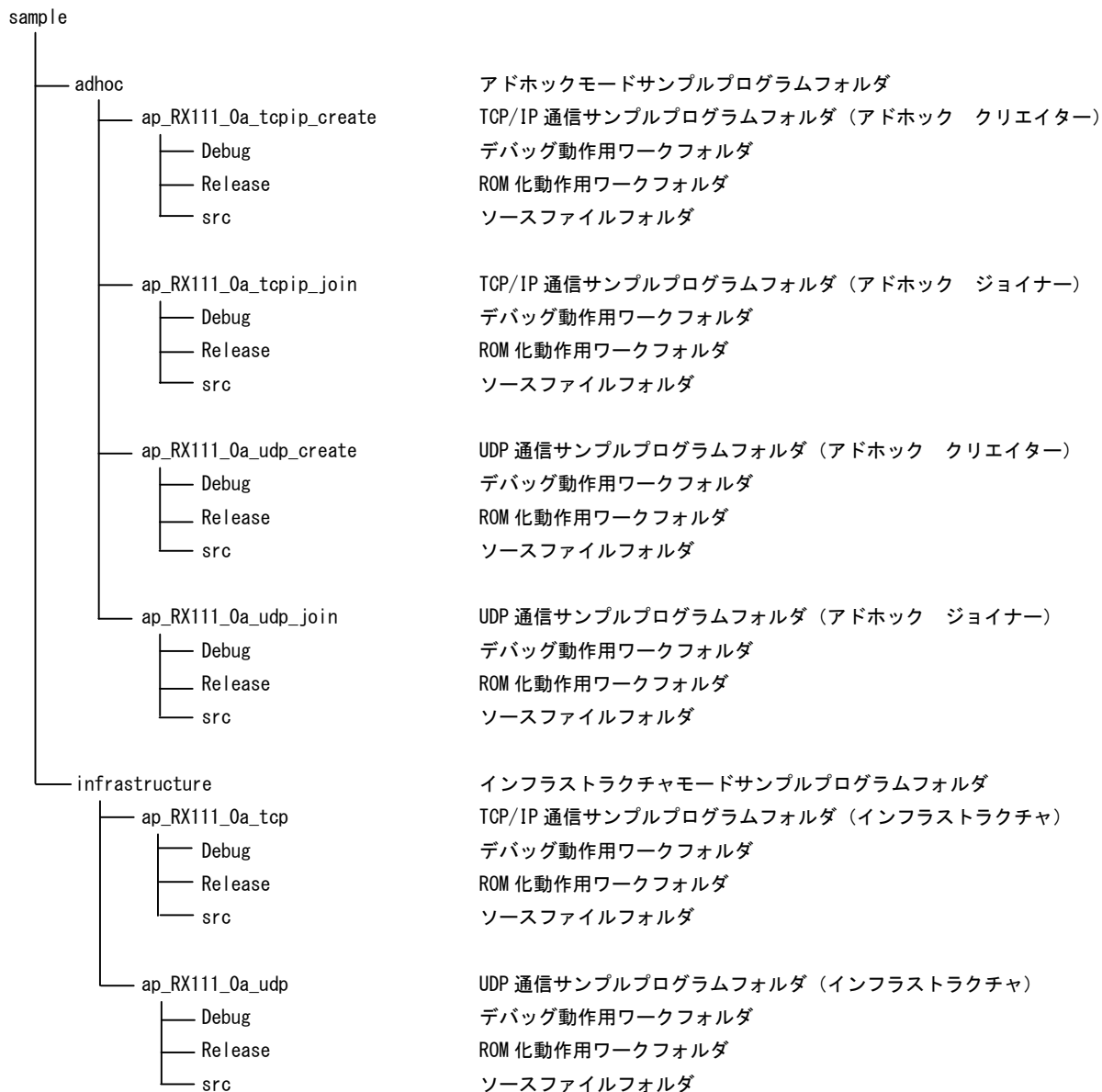
サンプルプログラム	フォルダ
TCP/IP サンプルプログラム (アドホックモード クリエイター)	¥sample¥adhoc¥ap_RX111_0a_tcpip_create¥ ap_RX111_0a_tcpip_create.mtpj
TCP/IP サンプルプログラム (アドホックモード ジョイナー)	¥sample¥adhoc¥ap_RX111_0a_tcpip_join¥ ap_RX111_0a_tcpip_join.mtpj
UDP サンプルプログラム (アドホックモード クリエイター)	¥sample¥adhoc¥ap_RX111_0a_udp_create¥ ap_RX111_0a_udp_create.mtpj
UDP サンプルプログラム (アドホックモード ジョイナー)	¥sample¥adhoc¥ap_RX111_0a_udp_join¥ ap_RX111_0a_udp_join.mtpj
TCP/IP サンプルプログラム (インフラストラクチャモード)	¥sample¥infrastructure¥ap_RX111_0a_tcp¥ap_RX111_0a_tcp.mtpj
UDP サンプルプログラム (インフラストラクチャモード)	¥sample¥infrastructure¥ap_RX111_0a_udp¥ap_RX111_0a_udp.mtpj



## 2. サンプルプログラムの構成

### 2. 1 フォルダ構成

サンプルプログラムは下記のようなフォルダ構成になっています。



## 2. 2 ファイル構成

サンプルプログラムは以下のファイルで構成されています。

<¥sample フォルダ内>

adhoc	...	アドホックモードサンプルプログラム
infrastructure	...	インフラストラクチャモードサンプルプログラム

### [アドホック]

<¥sample¥adhoc フォルダ内>

ap_rx111_0a_tcpip_create	...	TCP/IP 通信サンプルプログラムフォルダ (アドホック クリエイター)
ap_rx111_0a_tcpip_join	...	TCP/IP 通信サンプルプログラムフォルダ (アドホック ジョイナー)
ap_rx111_0a_udp_create	...	UDP 通信サンプルプログラムフォルダ (アドホック クリエイター)
ap_rx111_0a_udp_join	...	UDP 通信サンプルプログラムフォルダ (アドホック ジョイナー)

### [インフラストラクチャ]

<¥sample¥infrastructure フォルダ内>

ap_rx111_0a_tcp	...	TCP/IP 通信サンプルプログラムフォルダ
ap_rx111_0a_udp	...	UDP 通信サンプルプログラムフォルダ

※以下、「TCP/IP アドホック クリエイター」のフォルダ 及びファイル構成に関して記述致しますが、それ以外の  
「TCP/IP アドホック ジョイナー」  
「UDP アドホック クリエイター」  
「UDP アドホック ジョイナー」  
「TCP/IP インフラストラクチャ」  
「UDP インフラストラクチャ」

は、フォルダ名、ファイル名が違うだけとなります。

<%sample%adhoc%ap\_rx111\_0a\_tcpip\_create フォルダ内>

ap\_rx111\_0a\_tcpip\_create.mtpj … TCP/IP 通信サンプルプログラム CubeSuite+用ワークスペース  
ファイル (アドホック クリエイター)

<%sample%adhoc%ap\_rx111\_0a\_tcpip\_create%ap\_rx111\_0a\_tcpip\_create%Debug フォルダ内>

ap\_rx111\_0a\_tcpip\_create.abs … TCP/IP 通信サンプルプログラムデバッグ動作オブジェクト  
ファイル (アドホック クリエイター) (elf 形式)  
ap\_rx111\_0a\_tcpip\_create.mot … TCP/IP 通信サンプルプログラムデバッグ動作  
モトローラ S フォーマット形式ファイル  
(アドホック クリエイター)  
ap\_rx111\_0a\_tcpip\_create.map … TCP/IP 通信サンプルプログラムデバッグ動作マップファイ  
ル  
(アドホック クリエイター)  
コンパイル後は、.obj, .lib 等のファイルが生成されます

<%sample%adhoc%ap\_rx111\_0a\_tcpip\_create%ap\_rx111\_0a\_tcpip\_create%Release フォルダ内>

ap\_rx111\_0a\_tcpip\_create.abs … TCP/IP 通信サンプルプログラム ROM 化動作オブジェクト  
ファイル (アドホック クリエイター) (elf 形式)  
ap\_rx111\_0a\_tcpip\_create.mot … TCP/IP 通信サンプルプログラム ROM 化動作  
モトローラ S フォーマット形式ファイル  
(アドホック クリエイター)  
ap\_rx111\_0a\_tcpip\_create.map … TCP/IP 通信サンプルプログラム ROM 化動作マップファイ  
ル  
(アドホック クリエイター)  
コンパイル後は、.obj, .lib 等のファイルが生成されます

<%sample%adhoc%ap\_rx111\_0a\_tcpip\_create%ap\_rx111\_0a\_tcpip\_create%src フォルダ内>

ap\_rx111\_0a\_tcp.c … main 関数処理  
dbsct.c … CPU 初期化処理ファイル  
intprg.c … 割り込みベクタ処理関数ファイル  
resetprg.c … PowerOn リセット時の動作処理ファイル  
sample.c … サンプルプログラムメイン処理ファイル  
sbrk.c … ヒープメモリ処理ファイル  
sci.c … シリアル処理ファイル  
spi.c … SPI 処理ファイル  
tmr.c … タイマ処理ファイル  
vecttbl.c … 割込ベクタテーブルファイル  
wm\_rp\_04s.c … WiFi モジュールサンプルドライバファイル  
BoardDepend.h … ボード依存ファイル  
common.h … 共通ヘッダファイル  
iodefine.h … RX111 内部レジスタ定義ヘッダファイル  
sbrk.h … ヒープメモリサイズ定義ファイル  
stacksct.h … スタックサイズ定義ファイル  
typedefine.h … typedef 定義用ヘッダファイル  
vect.h … 割り込みベクタ処理関数定義ヘッダファイル  
wm\_rp\_04s.h … WiFi モジュールサンプルドライバヘッダファイル  
lowlvl.src … 入出力関連低レベル処理ファイル

※以下のフォルダ内のファイルは、「Redpine Signals 社」の SPI 用ライブラリファイルとなります。

<¥sample¥ad hoc¥ap\_rx111\_0a\_tcpip\_create¥ap\_rx111\_0a\_tcpip\_create¥src¥API\_Lib フォルダ内>

rsi_api_sysinit.c	...	WM-RP 初期化処理
rsi_hal_mcu_interrupt.c	...	MCU 依存割り込み処理
rsi_hal_mcu_ioports.c	...	MCU 依存 I/O 処理
rsi_hal_mcu_spi.c	...	MCU 依存 SPI 処理
rsi_hal_mcu_timers.c	...	MCU 依存タイマ処理
rsi_interrupt.c	...	WM-RP 各割り込み確認処理
rsi_lib_util.c	...	データ変換用ユーティリティ
rsi_spi_api.c	...	SPI による送信時の各処理定義データファイル
rsi_spi_band.c	...	band 処理
rsi_spi_bootloader.c	...	bootloader 処理
rsi_spi_disconnect.c	...	disconnect 処理
rsi_spi_execute_cmd.c	...	SPI による各処理コマンド送信処理
rsi_spi_feat_select.c	...	Feature Select 処理
rsi_spi_framerdwr.c	...	SPI Frame description 処理
rsi_spi_funcs.c	...	SPI による送信コマンド C1, C2, C3, C4 処理
rsi_spi_fwupgrade.c	...	FW upgrade 処理
rsi_spi_http_get.c	...	HTTP Get Request 処理
rsi_spi_http_post.c	...	HTTP Post Request 処理
rsi_spi_iface_init.c	...	WM-RP 初期化コマンド送信処理
rsi_spi_init.c	...	init 処理
rsi_spi_interrupt_handler.c	...	WM-RP からの割り込みステータス取得処理
rsi_spi_ipparam.c	...	Set IP Parameters 処理
rsi_spi_join.c	...	join 処理
rsi_spi_memrdwr.c	...	SPI Memory read/write 処理
rsi_spi_mode_select.c	...	TCP/IP Bypass 処理
rsi_spi_power_mode.c	...	Power Mode 処理
rsi_spi_query_bssid_nwtype.c	...	Query MAC address and Network Type of Scanned Networks 処理
rsi_spi_query_conn_status.c	...	Query Connection Status 処理
rsi_spi_query_dhcp_parms.c	...	Query DHCP Information 処理
rsi_spi_query_dns.c	...	DNS Request 処理
rsi_spi_query_fwversion.c	...	Query Firmware Version 処理
rsi_spi_query_macaddress.c	...	Query MAC Address 処理
rsi_spi_query_net_parms.c	...	Query Network Parameters 処理
rsi_spi_query_rssi.c	...	Query RSSI Value 処理
rsi_spi_read_packet.c	...	Receive data 処理
rsi_spi_regrdwr.c	...	WM-RP 内部レジスタアクセス処理
rsi_spi_scan.c	...	Scan 処理
rsi_spi_send_data.c	...	Send Data 処理
rsi_spi_send_ludp_data.c	...	Listening UDP Send 処理
rsi_spi_send_wps_data.c	...	WPS Send 処理
rsi_spi_set_listen_interval.c	...	Set a Listen Interval 処理
rsi_spi_set_mac_addr.c	...	Set MAC Address 処理
rsi_spi_sleep_timer.c	...	Sleep timer 設定処理
rsi_spi_socket.c	...	Open a Socket 処理
rsi_spi_socket_close.c	...	Close a Socket 処理

rsi_spi_store_config.c	...	Connecting to a Preconfigured Access Point 処理
rsi_spi_wepkeys.c	...	WEP Key 設定処理
rsi_spi_wl_bootloader.c	...	Wireless bootloader 処理
rsi_spi_wlfw_upgrade.c	...	Wireless FWUpgrade 処理
rsi_hal.h	...	ハードウェア依存処理
rsi_lib_util.h	...	データ変換用ユーティリティヘッダファイル
rsi_spi_api.h	...	SPI による送信時の各処理定義データヘッダファイル
Makefile	...	メイクファイル

<¥sample¥adhoc¥ap\_rx111\_0a\_tcpip\_create¥ap\_rx111\_0a\_tcpip\_create¥src¥API\_Lib¥Firmware フォルダ内>

ffdata	...	FW ファイル ※このファイルは、弊社からは提供していません。
ffinst1	...	FW ファイル ※このファイルは、弊社からは提供していません。
ffinst2	...	FW ファイル ※このファイルは、弊社からは提供していません。
iudata	...	FW ファイル ※このファイルは、弊社からは提供していません。
iuinst1	...	FW ファイル ※このファイルは、弊社からは提供していません。
iuinst2	...	FW ファイル ※このファイルは、弊社からは提供していません。
sbdata1	...	bootloader ファイル
sbdata2	...	bootloader ファイル
sbinst1	...	bootloader ファイル
sbinst2	...	bootloader ファイル

<¥sample¥adhoc¥ap\_rx111\_0a\_tcpip\_create¥ap\_rx111\_0a\_tcpip\_create¥src¥API\_Lib¥Wireless\_Upgrade フォルダ内>

※機能として存在しておりますが、基本的な無線通信を行う場合においては特に使用致しません。

cbdata	...	Wireless bootloader ファイル
cbinst1	...	Wireless bootloader ファイル
cbinst2	...	Wireless bootloader ファイル
cfdata	...	Wireless upgrade FW ファイル
cfinst1	...	Wireless upgrade FW ファイル
cfinst2	...	Wireless upgrade FW ファイル
cudata	...	Wireless upgrade FW ファイル
cuint1	...	Wireless upgrade FW ファイル
cuint2	...	Wireless upgrade FW ファイル
DeviceConfigGUI-v3.3.jar	...	Wireless upgrade PC ソフト
wlan_file.rps	...	Wireless upgrade PC ソフト用ファイル

<¥sample¥adhoc¥ap\_rx111\_0a\_tcpip\_create¥ap\_rx111\_0a\_tcpip\_create¥src¥API\_Lib¥wps フォルダ内>

このフォルダ内のファイルは、WPS パケット送信処理を行う為のファイル群です。

※機能として存在しておりますが、基本的な無線通信を行う場合においては特に使用致しません。

<¥sample¥adhoc¥ap\_rx111\_0a\_tcpip\_create¥ap\_rx111\_0a\_tcpip\_create¥src¥Applications¥MCU フォルダ内>

rsi_app_util.c	...	アプリ作成用ユーティリティ
rsi_config_init.c	...	各種ネットワーク設定
rsi_app_util.h	...	アプリ作成用ユーティリティヘッダファイル
rsi_config.h	...	各種ネットワーク設定定義ヘッダファイル
rsi_global.h	...	各種処理上の構造体などの定義ヘッダファイル
Makefile	...	メイクファイル
main.c	...	参考用の main プログラム

### 3. TCP/IP 通信サンプルプログラム

#### 3. 1 ビルド・デバッグ方法

サンプルプログラムを CPU ボード上で実行するためには、ビルドしたサンプルプログラムの実行ファイルを CPU ボードにダウンロードする必要があります。

サンプルプログラムのビルド方法および CPU ボードにサンプルプログラムをダウンロードする方法については、アプリケーションノート「AN1513 RX 開発環境の使用方法 (CubeSuite+)」に詳細な手順が記されていますので、参照してください。

ビルド・動作確認方法		
項目名	項目番号	設定値
アドホック：クリエイター		
出力フォルダ	2-2	¥sample¥adhoc¥ap_rx111_0a_tcpip_create¥Debug
モトローラファイル名	2-3	¥sample¥adhoc¥ap_rx111_0a_tcpip_create¥Debug¥ap_rx111_0a_tcpip_create.mot
アブソリュートファイル名	2-4	¥sample¥adhoc¥ap_rx111_0a_tcpip_create¥Debug¥ap_rx111_0a_tcpip_create.abs
マップファイル	2-5	¥sample¥adhoc¥ap_rx111_0a_tcpip_create¥Debug¥ap_rx111_0a_tcpip_create.map
アドホック：ジョイナー		
出力フォルダ	2-2	¥sample¥adhoc¥ap_rx111_0a_tcpip_join¥Debug
モトローラファイル名	2-3	¥sample¥adhoc¥ap_rx111_0a_tcpip_join¥Debug¥ap_rx111_0a_tcpip_join.mot
アブソリュートファイル名	2-4	¥sample¥adhoc¥ap_rx111_0a_tcpip_join¥Debug¥ap_rx111_0a_tcpip_join.abs
マップファイル	2-5	¥sample¥adhoc¥ap_rx111_0a_tcpip_join¥Debug¥ap_rx111_0a_tcpip_join.map
インフラストラクチャー		
出力フォルダ	2-2	¥sample¥infrastructure¥ap_rx111_0a_tcp¥Debug
モトローラファイル名	2-3	¥sample¥infrastructure¥ap_rx111_0a_tcp¥Debug¥ap_rx111_0a_tcp.mot
アブソリュートファイル名	2-4	¥sample¥infrastructure¥ap_rx111_0a_tcp¥Debug¥ap_rx111_0a_tcp.abs
マップファイル	2-5	¥sample¥infrastructure¥ap_rx111_0a_tcp¥Debug¥ap_rx111_0a_tcp.map

E1 エミュレータを使用したデバッグ方法		
項目名	項目番号	設定値
ボード設定	4-1	Fig 3.1-1 を参照
JTAG クロック	4-10	入力不要
EXTAL クロック	4-11	16 (MHz)

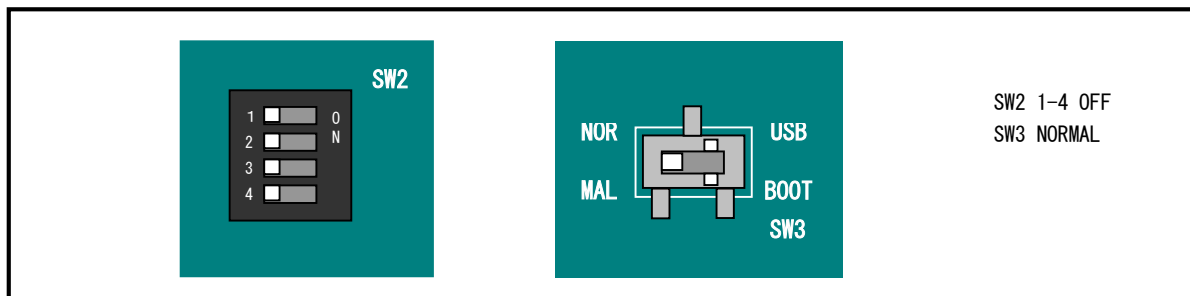


Fig 3.1-1 E1 エミュレータデバッグ時のボード設定

### 3. 2 動作説明 (TCP/IP 通信)

#### 3. 2. 1 サンプルプログラム概要 (TCP/IP 通信 アドホックモード)

TCP/IP 通信サンプルプログラム (アドホックモード) は、下記の動作を行います。

##### 1) クリエイター

- SPI 接続された WM-RP-0xS に対してコマンドを送信し、TCP/IP エコーバックサーバを構築します。その後、受信したデータをそのまま送信元に送信します。  
動作確認は、ホスト PC 上のターミナルソフト (ハイパーターミナルなど) を使用して行ってください。  
※ TCP/IP エコーバックサーバ動作の詳細は、「3.2.3 TCP/IP 通信エコーバックサーバ動作」を参照してください。

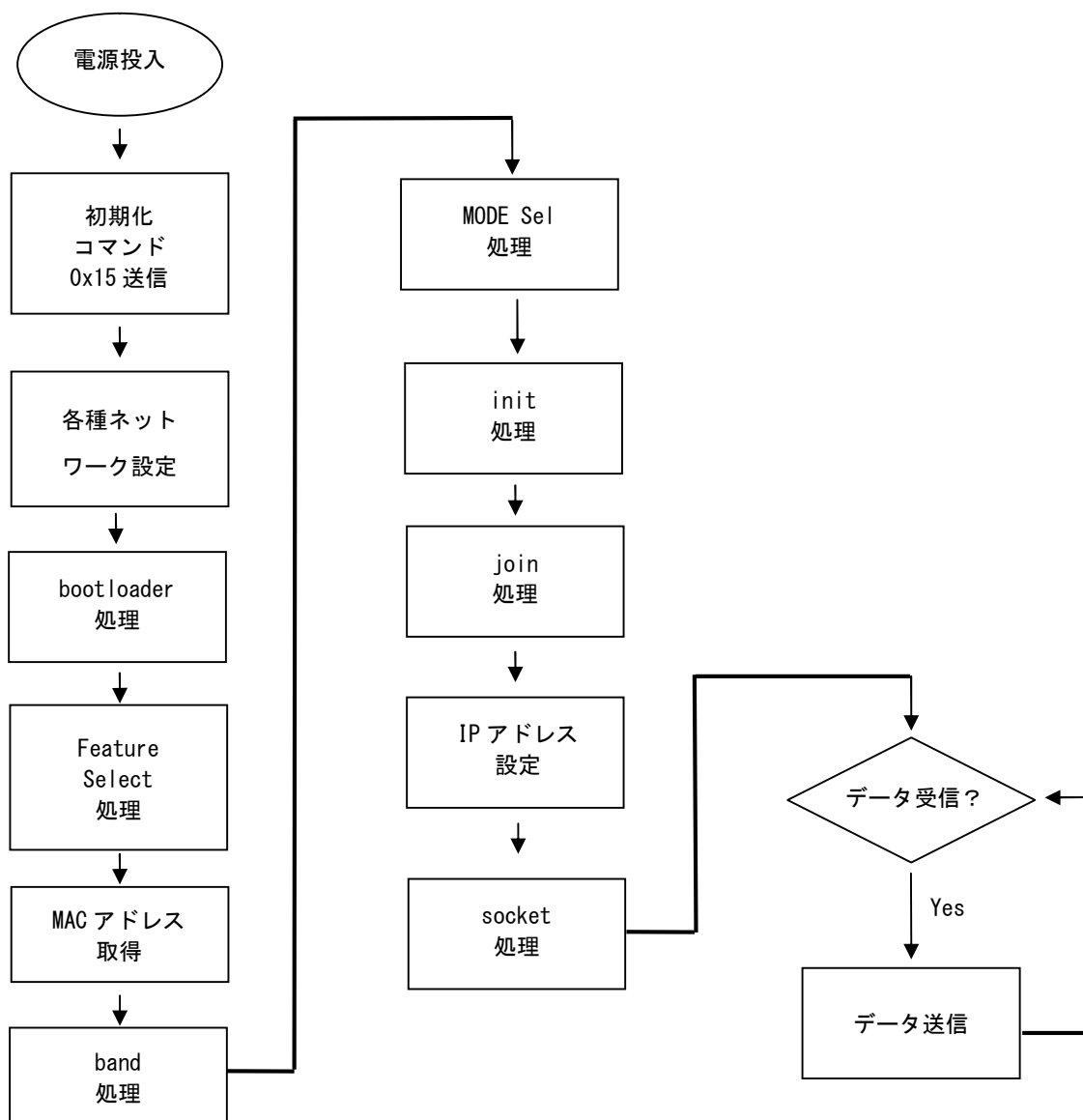


Fig 3.2-1 TCP/IP 通信サンプルプログラム アドホッククリエイター WM-RP-0xS 制御フロー

2) ジョイナー ※ジョイナー処理は、「scan」処理が加わります。

- SPI 接続された WM-RP-0xS に対してコマンドを送信し、TCP/IP エコーバックサーバを構築します。その後、受信したデータをそのまま送信元に送信します。動作確認は、ホスト PC 上のターミナルソフト（ハイパーターミナルなど）を使用して行ってください。  
※ TCP/IP エコーバックサーバ動作の詳細は、「3.2.3 TCP/IP 通信エコーバックサーバ動作」を参照してください。

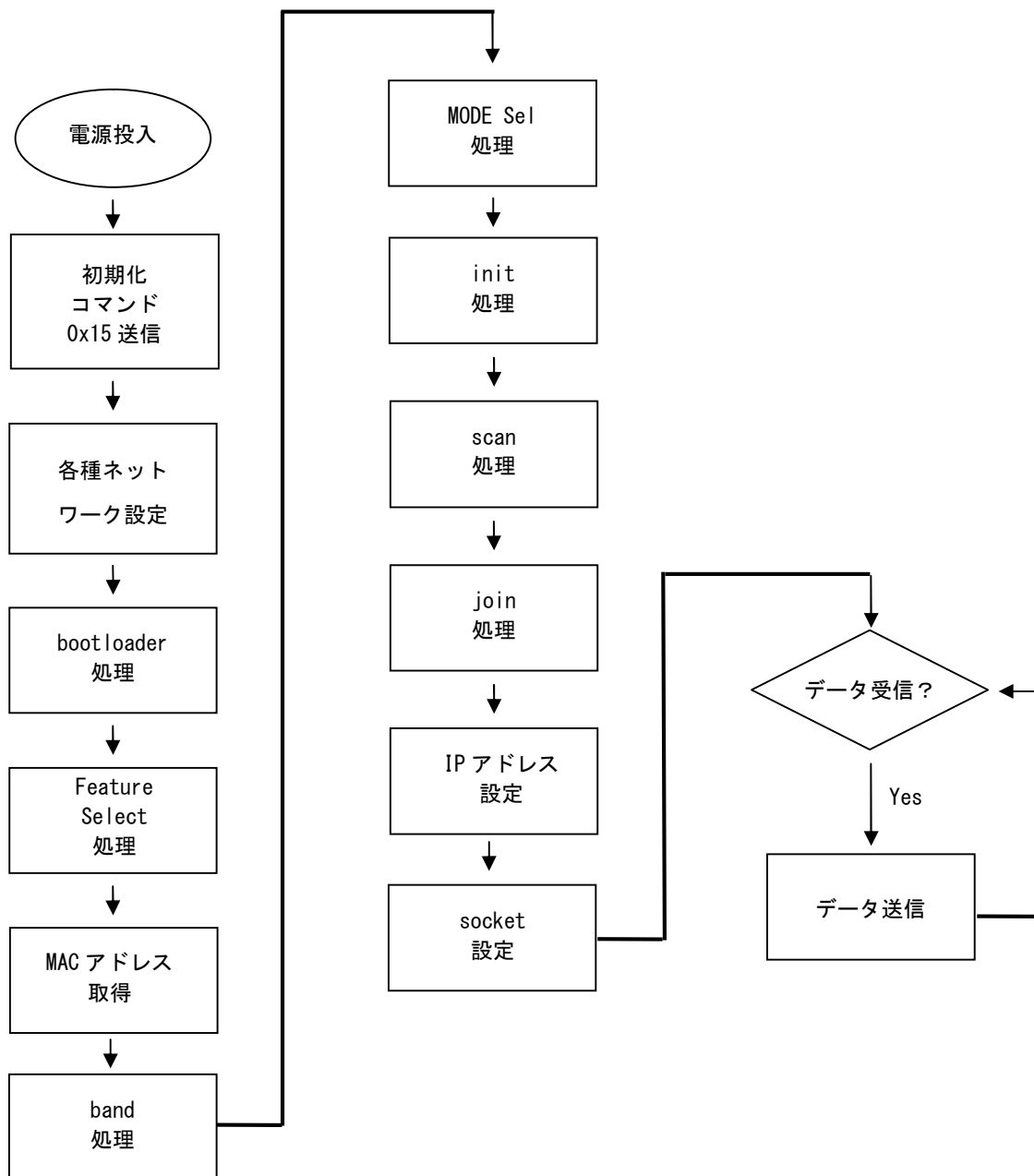


Fig 3.2-2 TCP/IP 通信サンプルプログラム アドホックジョイナー WM-RP-0xS 制御フロー



3. 2. 2 サンプルプログラム概要 (TCP/IP 通信 インフラストラクチャモード)

TCP/IP 通信サンプルプログラム (インフラストラクチャモード) は、下記の動作を行います。

- SPI 接続された WM-RP-0xS に対してコマンドを送信し、インフラストラクチャモードでアクセスポイントに接続した後、TCP/IP エコーバックサーバを構築します。その後、受信したデータをそのまま送信元に送信します。動作確認は、ホスト PC 上のターミナルソフト (ハイパーターミナルなど) を使用して行ってください。  
※ TCP/IP エコーバックサーバ動作の詳細は、「3.2.3 TCP/IP 通信エコーバックサーバ動作」を参照してください。

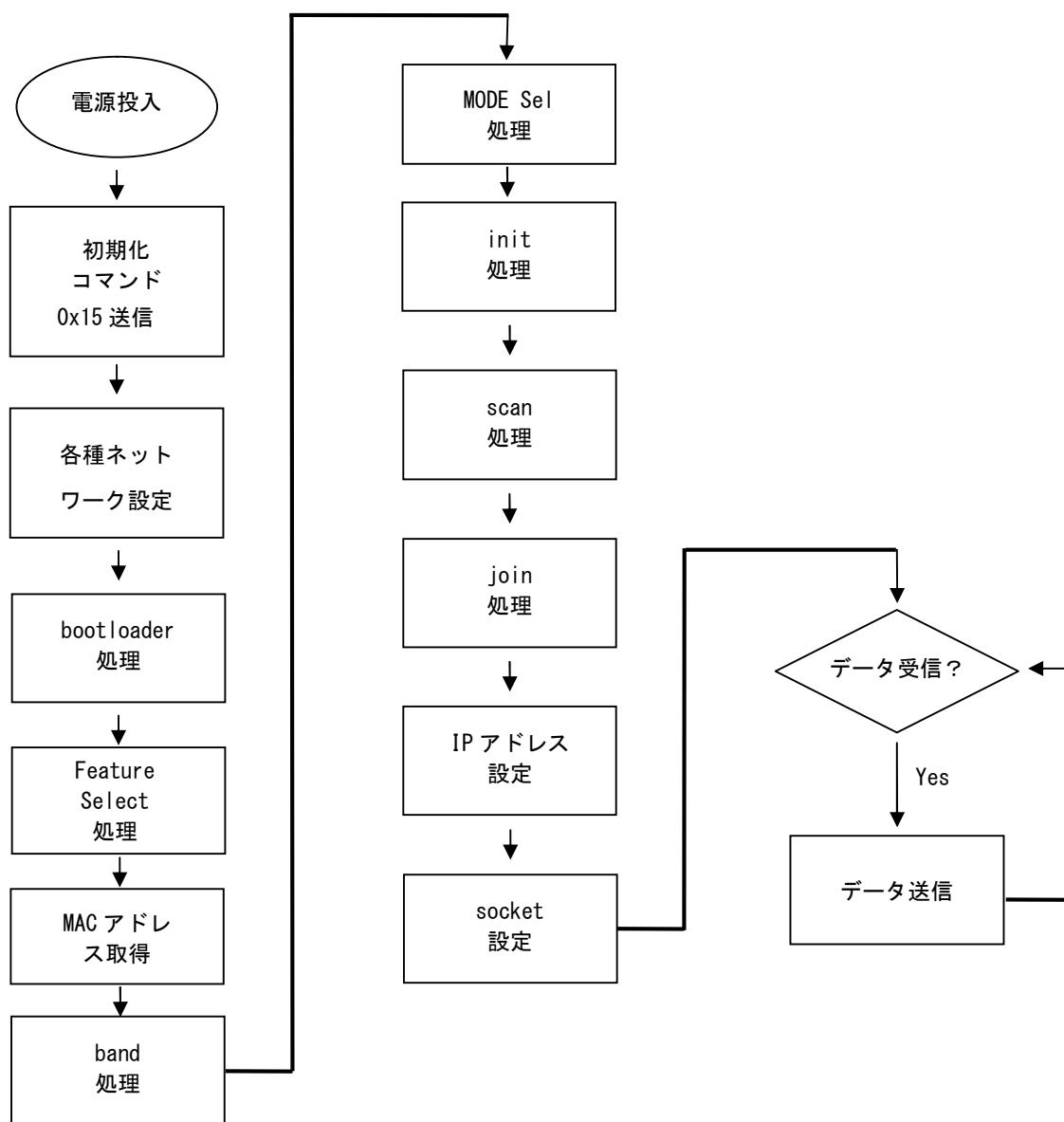


Fig 3.2-3 TCP/IP 通信サンプルプログラム インフラストラクチャ WM-RP-0xS 制御フロー

## 3. 2. 3 TCP/IP 通信エコーバックサーバ動作

## (1) TCP/IP ネットワーク設定

以下にサンプルプログラムのネットワーク設定を記します。

TCP/IP ネットワーク設定 (アドホックモード)	
使用帯域	2.4GHz
使用チャンネル	3ch
ネットワーク接続	アドホックモード
送信レート	自動設定
送信レベル	ハイレベル
PSK	-
アクセスポイント SSID	WM-RP_RX111Sample
IP アドレス	192.168.1.200
サブネットマスク	255.255.255.0
ゲートウェイ	192.168.1.253
使用するポート	8001

Table3.2-1 TCP/IP ネットワーク設定 (アドホックモード)

TCP/IP ネットワーク設定 (インフラストラクチャモード)	
使用帯域	2.4GHz
使用チャンネル	3ch
ネットワーク接続	インフラストラクチャモード
送信レート	自動設定
送信レベル	ハイレベル
PSK	WM-RP_RX111SamplePSK
アクセスポイント	WM-RP_RX111Sample
IP アドレス	192.168.1.200
サブネットマスク	255.255.255.0
ゲートウェイ	192.168.1.253
使用するポート	8001

Table3.2-2 TCP/IP ネットワーク設定 (インフラストラクチャモード)

※ これらの設定は弊社の環境において設定した値となっています。ご利用時は、お使いの環境のネットワーク管理者に問い合わせ、ご利用になられる環境に沿ったそれぞれ適切な値を設定しビルドしてください。

## (2) TCP/IP 通信エコーバックサーバ動作 (アドホックモード クリエイター)

以下の手順に従い、TCP/IP 通信エコーバックサーバの動作を確認してください。

- ① CPU ボードに電源を投入し、サンプルプログラムを動作させます。AP-RX111-0A 上の LD1 が点滅します。
- ② アドホック通信機器の設定を行います。  
その際、使用する設定は「Table3.2-1 TCP/IP ネットワーク設定 (アドホックモード)」で設定した値となります。
- ③ アドホック機器を無線 LAN ネットワークに接続し、エコーバックが行われることを確認してください。
- ④ 以上で TCP/IP 通信エコーバックサーバ動作 (アドホックモード クリエイター) の動作確認は終了です。

## (3) TCP/IP 通信エコーバックサーバ動作 (アドホックモード ジョイナー)

以下の手順に従い、TCP/IP 通信エコーバックサーバの動作を確認してください。

- ① アドホック通信機器の設定を行い、無線 LAN ネットワークに接続します。  
その際、使用する設定は「Table3.2-1 TCP/IP ネットワーク設定 (アドホックモード)」で設定した値となります。
- ② CPU ボードに電源を投入し、サンプルプログラムを動作させます。AP-RX111-0A 上の LD1 が点滅します。
- ③ アドホック通信機器の側からエコーバックが行われることを確認してください。
- ④ 以上で TCP/IP 通信エコーバックサーバ動作 (アドホックモード ジョイナー) の動作は終了です。

## (4) TCP/IP 通信エコーバックサーバ動作 (インフラストラクチャモード)

以下の手順に従い、TCP/IP 通信エコーバックサーバの動作を確認してください。

- ① CPU ボードに電源を投入し、サンプルプログラムを動作させます。AP-RX111-0A 上の LD1 が点滅します。
- ② ホスト PC 上でターミナルソフト (ハイパーターミナルなど) を起動し、Ethernet 接続の設定を行います。  
その際、使用する設定は「Table3.2-2 TCP/IP ネットワーク設定 (インフラストラクチャモード)」で設定した値となります。
- ③ ターミナルソフトを使用し、エコーバックが行われることを確認してください。
- ④ 以上で TCP/IP 通信エコーバックサーバ動作 (インフラストラクチャモード) の動作確認は終了です。

3. 3 メモリマップ (TCP/IP 通信サンプルプログラム共通)

メモリマップを以下に示します。

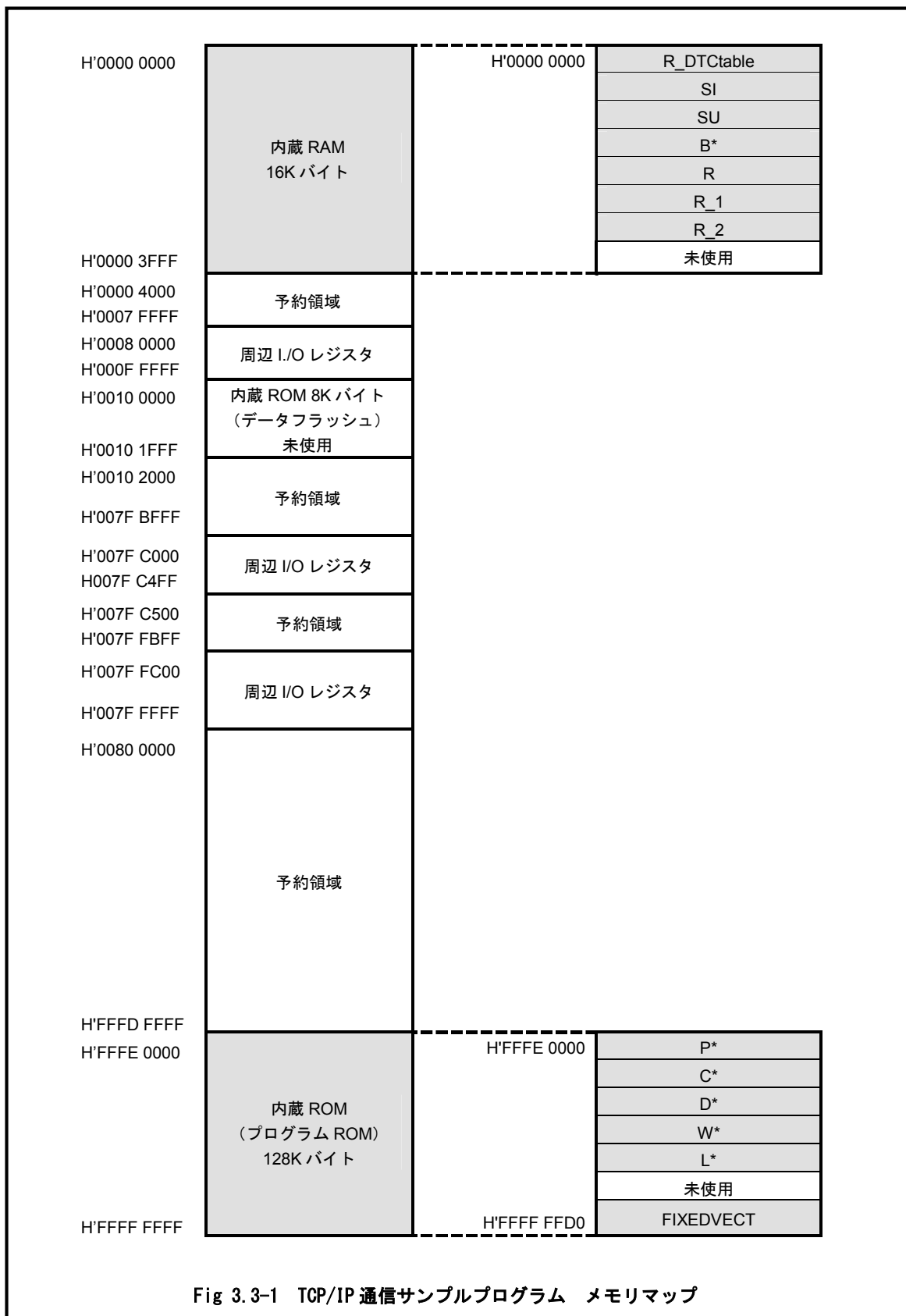


Fig 3.3-1 TCP/IP 通信サンプルプログラム メモリマップ

## 4. UDP 通信サンプルプログラム

### 4. 1 ビルド・デバッグ方法 (UDP 通信サンプルプログラム)

サンプルプログラムを CPU ボード上で実行するためには、ビルドしたサンプルプログラムの実行ファイルを CPU ボードにダウンロードする必要があります。

サンプルプログラムのビルド方法および CPU ボードにサンプルプログラムをダウンロードする方法については、アプリケーションノート「AN1513 RX 開発環境の使用方法 (CubeSuite+)」に詳細な手順が記されていますので、参照してください。

ビルド・動作確認方法		
項目名	項目番号	設定値
アドホック：クリエイター		
出力フォルダ	2-2	¥sample¥adhoc¥ap_rx111_0a_udp_create¥Debug
モトローラファイル名	2-3	¥sample¥adhoc¥ap_rx111_0a_udp_create¥Debug¥ap_rx111_0a_udp_create.mot
アプソリュートファイル名	2-4	¥sample¥adhoc¥ap_rx111_0a_udp_create¥Debug¥ap_rx111_0a_udp_create.abs
マップファイル	2-5	¥sample¥adhoc¥ap_rx111_0a_udp_create¥Debug¥ap_rx111_0a_udp_create.map
アドホック：ジョイナー		
出力フォルダ	2-2	¥sample¥adhoc¥ap_rx111_0a_udp_join¥Debug
モトローラファイル名	2-3	¥sample¥adhoc¥ap_rx111_0a_udp_join¥Debug¥ap_rx111_0a_udp_join.mot
アプソリュートファイル名	2-4	¥sample¥adhoc¥ap_rx111_0a_udp_join¥Debug¥ap_rx111_0a_udp_join.abs
マップファイル	2-5	¥sample¥adhoc¥ap_rx111_0a_udp_join¥Debug¥ap_rx111_0a_udp_join.map
インフラストラクチャー		
出力フォルダ	2-2	¥sample¥infrastructure¥ap_rx111_0a_udp¥Debug
モトローラファイル名	2-3	¥sample¥infrastructure¥ap_rx111_0a_udp¥Debug¥ap_rx111_0a_udp.mot
アプソリュートファイル名	2-4	¥sample¥infrastructure¥ap_rx111_0a_udp¥Debug¥ap_rx111_0a_udp.abs
マップファイル	2-5	¥sample¥infrastructure¥ap_rx111_0a_udp¥Debug¥ap_rx111_0a_udp.map

E1 エミュレータを使用したデバッグ方法		
項目名	項目番号	設定値
ボード設定	4-1	Fig 4.1-1 を参照
JTAG クロック	4-10	入力不要
EXTAL クロック	4-11	16 (MHz)

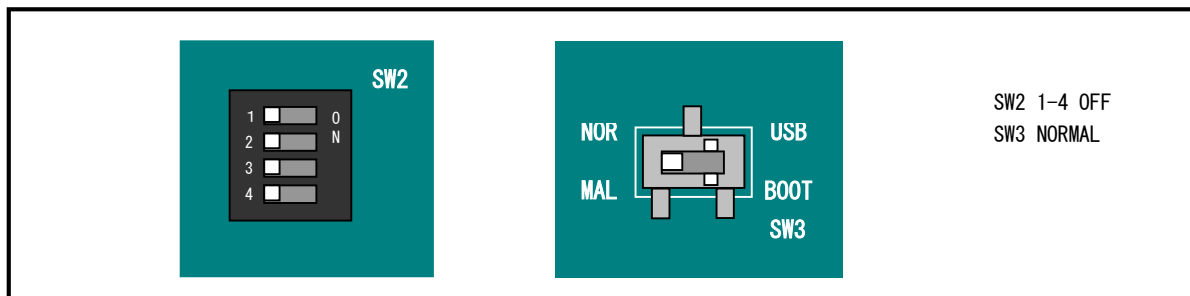


Fig 4.1-1 E1 エミュレータデバッグ時のボード設定

4. 2 動作説明 (UDP 通信)

4. 2. 1 サンプルプログラム概要 (UDP 通信 アドホックモード)

UDP 通信サンプルプログラム (アドホックモード) は、下記の動作を行います。

1) クリエイター

- SPI 接続された WM-RP-0xS に対してコマンドを送信し、UDP ポートを開放します。その後、UDP 通信で受信したデータをそのまま UDP 通信で送信元に送信します。  
動作確認は、ホスト PC 上のターミナルソフト (ハイパーターミナルなど) を使用して行ってください。  
※ UDP エコーバックサーバ動作の詳細は、「4. 2. 3 UDP 通信エコーバックサーバ動作」を参照してください。

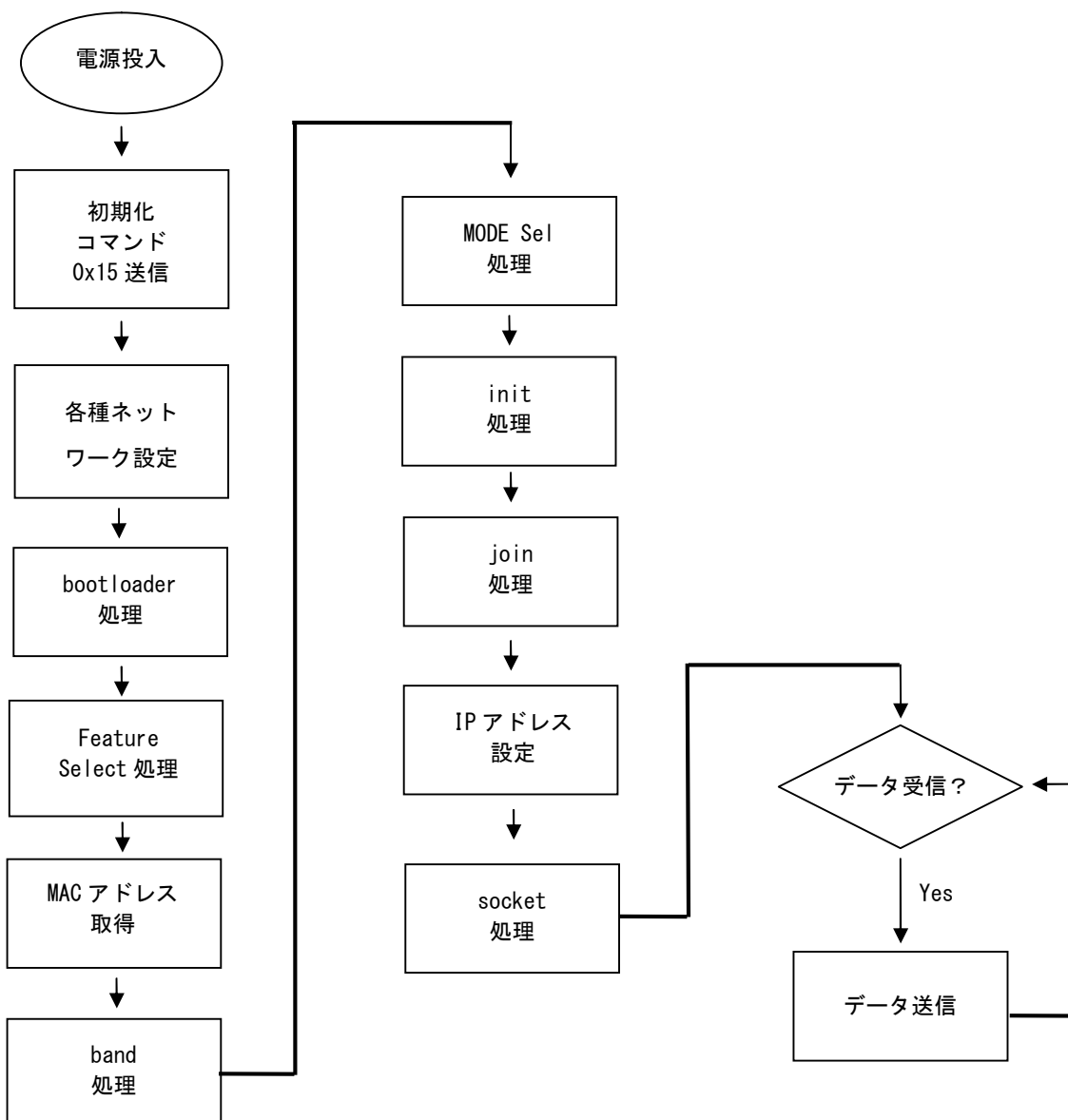


Fig 4. 2-1 UDP 通信サンプルプログラム アドホッククリエイター WM-RP-0xS 制御フロー

2) ジョイナー ※ジョイナー処理は、「scan」処理が加わります。

- SPI 接続された WM-RP-0xS に対してコマンドを送信し、アドホックモードでアクセスポイントに接続した後、UDP 通信で受信したデータをそのまま UDP 通信で送信元に送信します。  
動作確認は、ホスト PC 上のターミナルソフト（ハイパーターミナルなど）を使用して行ってください。  
※ UDP エコーバックサーバ動作の詳細は、「4.2.3 UDP 通信エコーバックサーバ動作」を参照してください。

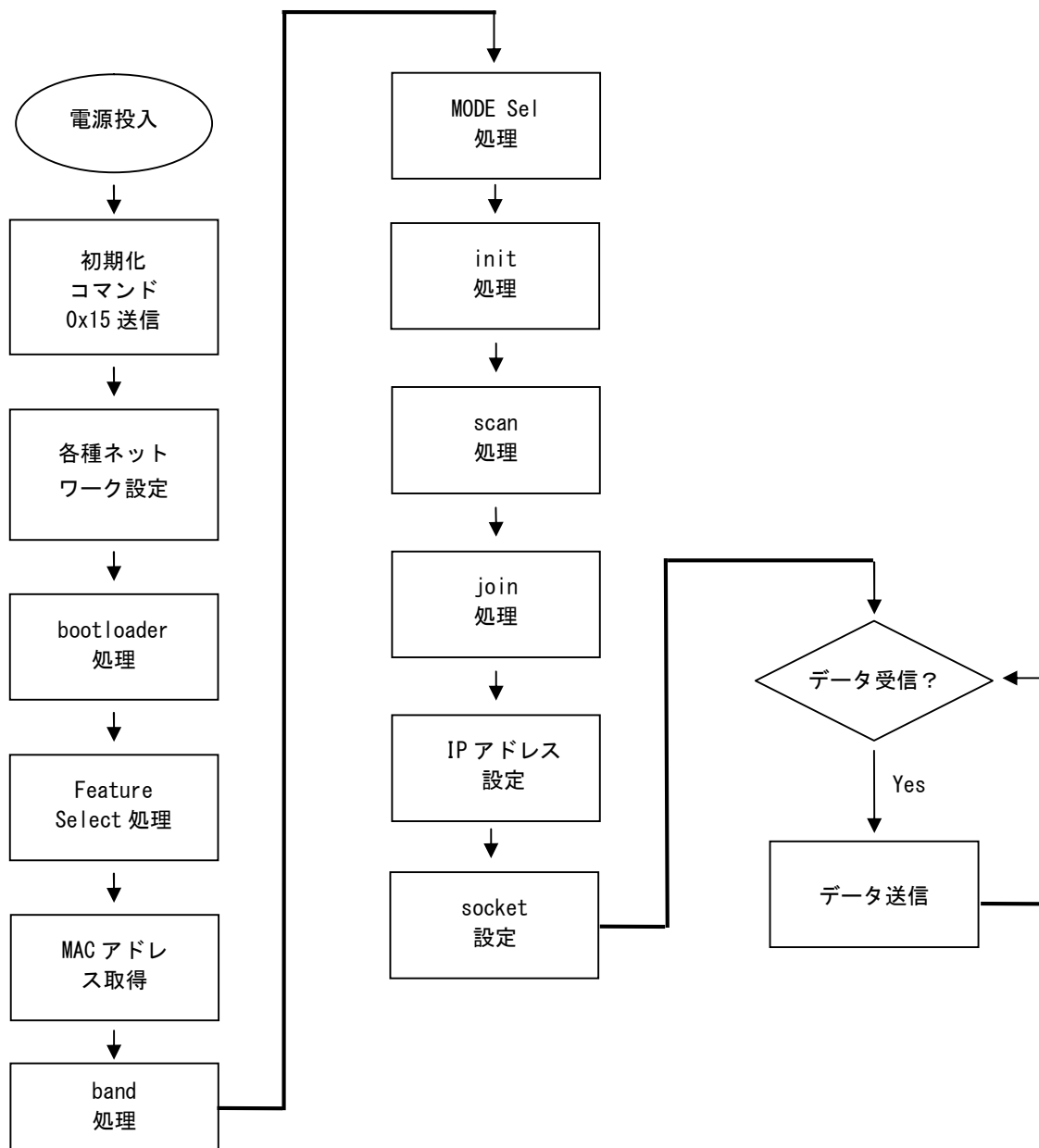


Fig 4.2-2 UDP 通信サンプルプログラム アドホックジョイナー WM-RP-0xS 制御フロー

4. 2. 2 サンプルプログラム概要 (UDP 通信 インフラストラクチャモード)

TCP/IP 通信サンプルプログラム (インフラストラクチャモード) は、下記の動作を行います。

- SPI 接続された WM-RP-0xS に対してコマンドを送信し、UDP エコーバックサーバを構築します。UDP 通信で受信したデータをそのまま UDP 通信で送信元へ送信します。動作確認は、ホスト PC 上のターミナルソフト (ハイパーターミナルなど) を使用して行ってください。  
※ UDP エコーバックサーバ動作の詳細は、「4. 2. 3 UDP 通信エコーバックサーバ動作」を参照してください。

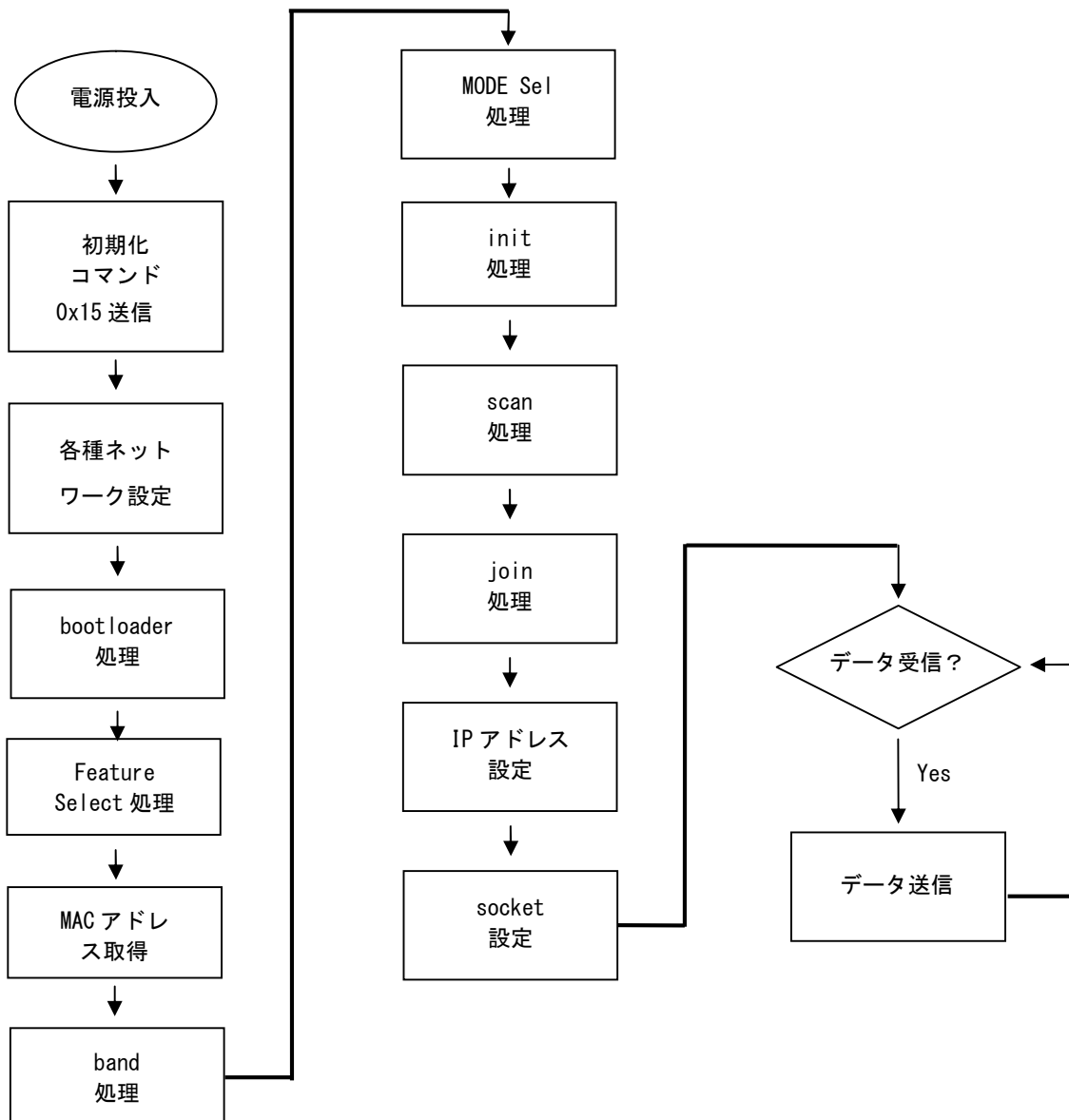


Fig 4. 2-3 UDP 通信サンプルプログラム インフラストラクチャ WM-RP-0xS 制御フロー



## 4. 2. 3 UDP 通信エコーバックサーバ動作

## (1) ネットワーク設定

以下にUDP 通信エコーバックサーバのネットワーク設定を記します。

UDP 通信サンプルプログラムネットワーク設定 (アドホックモード)	
使用帯域	2.4GHz
使用チャンネル	3ch
ネットワーク接続	アドホックモード
送信レート	自動設定
送信レベル	ハイレベル
PSK	-
アクセスポイント	WM-RP_RX111Sample
IP アドレス	192.168.1.200
サブネットマスク	255.255.255.0
ゲートウェイ	192.168.1.253
使用ポート	8001

Table4.2-1 UDP ネットワーク設定 (アドホックモード)

UDP 通信サンプルプログラムネットワーク設定 (インフラストラクチャモード)	
使用帯域	2.4GHz
使用チャンネル	3ch
ネットワーク接続	インフラストラクチャモード
送信レート	自動設定
送信レベル	ハイレベル
PSK	WM-RP_RX111SamplePSK
アクセスポイント	WM-RP_RX111Sample
IP アドレス	192.168.1.200
サブネットマスク	255.255.255.0
ゲートウェイ	192.168.1.253
使用ポート	8000

Table4.2-2 UDP ネットワーク設定 (インフラストラクチャモード)

※ これらの設定は弊社環境において設定した値となっています。ご利用時は、お使いの環境のネットワーク管理者に問い合わせ、ご利用になられる環境に沿ったそれぞれ適切な値を設定してください。

## (2) UDP 通信エコーバックサーバ動作 (アドホックモード クリエイター)

以下の手順に従い、UDP 通信エコーバックサーバの動作を確認してください。

- ① CPU ボードに電源を投入し、サンプルプログラムを動作させます。AP-RX111-0A 上の LD1 が点滅します。
- ② アドホック通信機器の設定を行います。  
その際、使用する設定は「Table4.2-1 UDP 通信サンプルプログラムネットワーク設定 (アドホックモード)」で設定した値となります。
- ③ アドホック機器を無線 LAN ネットワークに接続し、エコーバックが行われることを確認してください。
- ④ 以上で UDP 通信エコーバックサーバ動作 (アドホックモード クリエイター) の動作確認は終了です。

## (3) UDP 通信エコーバックサーバ動作 (アドホックモード ジョイナー)

以下の手順に従い、UDP 通信エコーバックサーバの動作を確認してください。

- ① アドホック通信機器の設定を行い、無線 LAN ネットワークに接続します。  
その際、使用する設定は「Table4.2-1 UDP 通信サンプルプログラムネットワーク設定 (アドホックモード)」で設定した値となります。
- ② CPU ボードに電源を投入し、サンプルプログラムを動作させます。AP-RX111-0A 上の LD1 が点滅します。
- ③ アドホック通信機器の側からエコーバックが行われることを確認してください。
- ④ 以上で UDP 通信エコーバックサーバ動作 (アドホックモード ジョイナー) の動作は終了です。

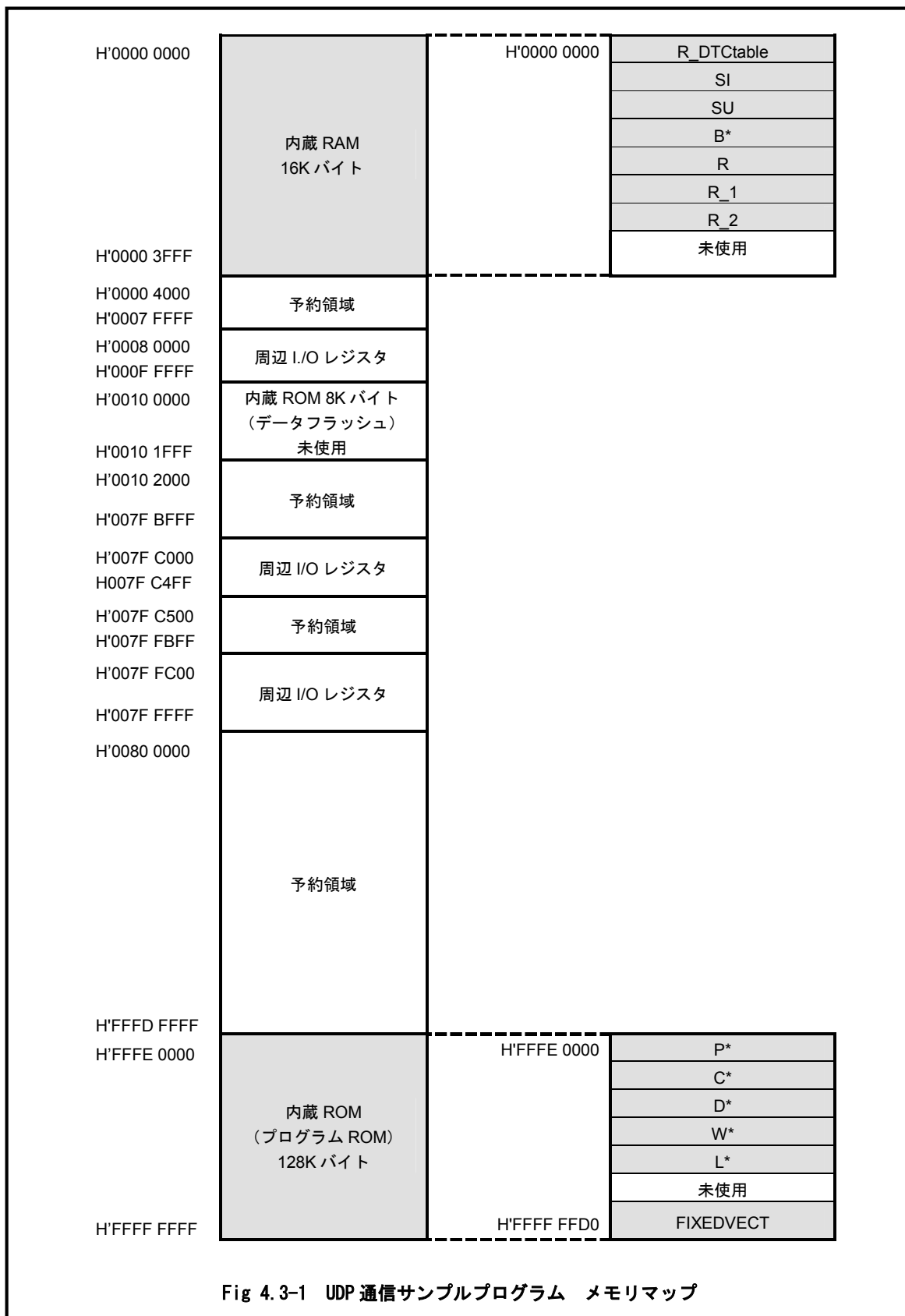
## (4) UDP 通信エコーバックサーバ動作 (インフラストラクチャモード)

以下の手順に従い、UDP 通信エコーバックサーバの動作を確認してください。

- ① CPU ボードに電源を投入し、サンプルプログラムを動作させます。AP-RX111-0A 上の LD1 が点滅します。
- ② ホスト PC 上でターミナルソフト (ハイパーターミナルなど) を起動し、Ethernet 接続の設定を行います。  
その際、使用する設定は「Table4.2-2 UDP 通信サンプルプログラムネットワーク設定 (インフラストラクチャモード)」で設定した値となります。
- ③ ターミナルソフトを使用し、エコーバックが行われることを確認してください。
- ④ 以上で UDP 通信エコーバックサーバ動作 (インフラストラクチャモード) の動作確認は終了です。

4. 3 メモリマップ (UDP 通信サンプルプログラム共通)

メモリマップを以下に示します。



## 5. WM-RP-0xS 制御方法

### 5. 1 概要

WM-RP-0xS はホスト CPU とのインタフェースに SPI を採用しています。  
 ホスト CPU は SPI から各種バイナリコマンドを送信することで WM-RP-0xS の操作を行い、初期化、ネットワークの設定、データの送受信などを行います。

### 5. 2 SPI

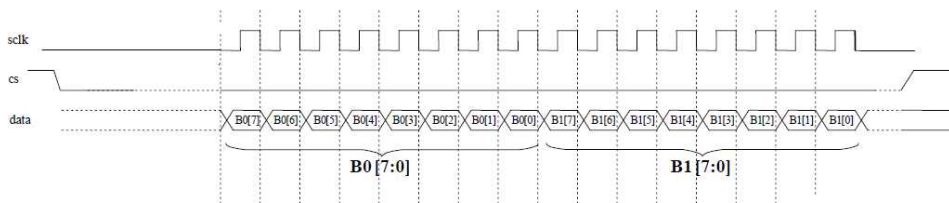
#### 5. 2. 1 SPI 仕様

WM-RP-0xS の SPI 仕様を以下に記します。

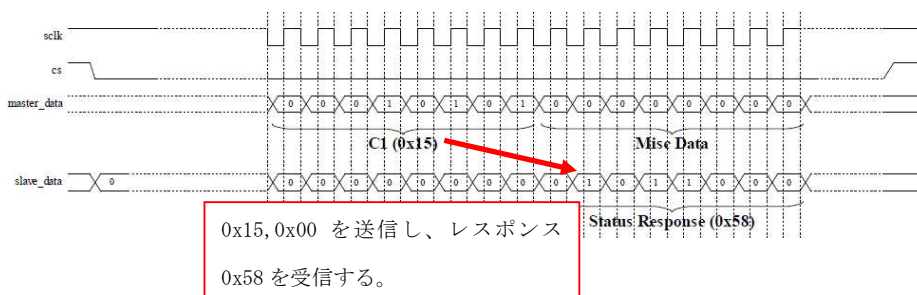
機能	仕様
通信方式	SPI 4 線式 SCK, SCS, MOSI, MISO
SPI クロック	25MHz (MAX)
データ	ホスト CPU が SPI マスター、MSB ファースト
割り込み	INTR 信号 ※ハイレベル割り込みです。 エッジ割り込みでは、ありません。

Table 5.2-1 SPI 仕様

下記画像は、SPI 動作時のタイミングです。ホスト CPU が SPI マスターとなり MSB ファーストにてデータを送受信します。データは、CLK の立ち上がりで有効です。



下記画像は、WM-RP-0xS の初期化処理時の動作です。ホスト CPU から「0x15」を送信し、そのレスポンスである「0x58」を WM-RP-0xS から受信する為に「0x00」（画像では、MiscData）を送信しています。



0x15, 0x00 を送信し、レスポンス 0x58 を受信する。

### 5. 2. 2 SPI 通信の基本的な流れ

WM-RP-0xS とホスト CPU との SPI 通信は以下の流れで行われます。

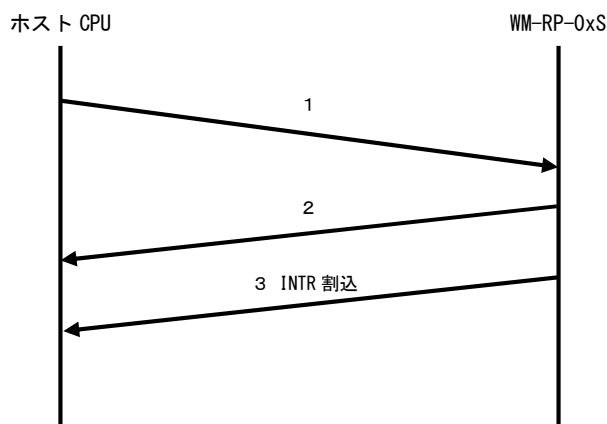


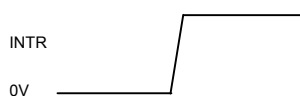
Fig 5.2-1 SPI インタフェース制御フロー

1. WM-RP-0xS へのコマンド入力です。バイナリコマンドを送信することで、ホスト CPU から WM-RP-0xS を制御することができます。
2. WM-RP-0xS からのレスポンス (0x58、0x55 など) です。バイナリデータにてホスト CPU に応答が返されます。
3. その後、コマンドによって INTR 割り込みが入りレスポンス受信を行います。

※ 送信するコマンドおよびレスポンスの詳細に関しましては、「RS9110-N-11-22\_24\_28-Software\_PRM.pdf」を参照してください。

### 5. 2. 3 INTR 割り込み信号

割り込みは、WM-RP-0xS からのハイレベル割り込みとなります。※エッジ割り込みではありません。



WM-RP-0xS が以下の状態の時、INTR 割り込みが発生します。

- 無線通信の相手より、データを受信した。
- WM-RP-0xS へのコマンド処理による、レスポンスデータがある場合。
- WM-RP-0xS 内部 SPI 受信メモリバッファがフルとなった。  
※ SPI 受信用メモリは、1460Byte で 7 個のメモリが用意されています。
- PowerMode1 の起床時

**割り込み発生時は、必ず適切な処理を他の処理よりも先に処理して下さい。**

例えば、無線データ送信処理中に割り込みが発生した場合は送信処理完了後に割り込みの処理を行って下さい。

割り込み処理が適切に処理されなかった場合、WM-RP-0xS からの応答が 0x54 ビジーレスポンスとなります。

この 0x54 ビジーレスポンスとなった場合、最悪の場合電源を OFF → ON する必要があります。

#### 5. 2. 4 無線データ送信処理

無線データ送信時は、送信処理を行う場合は、必ず `rsi_intHandler()` 関数を呼び出し `rsi_strIntStatus.bufferFull` を確認して下さい。

これは、WM-RP モジュール内部バッファが FULL かどうかを確認しております。

もし、FULL となった場合は FULL が解除されてから送信を行う様にプログラムして下さい。

以下は、`rsi_spi_send_data.c` ファイルの `rsi_send_data()` 関数（無線データ送信）内 92 行目～97 行目の処理です。

```
rsi_intHandler();
if(rsi_strIntStatus.bufferFull == RSI_TRUE)
// if(rsi_checkBufferFullIrq() == RSI_TRUE)
{
    return RSI_BUFFER_FULL;
}
```

### 5. 3 Redpine Signals 社提供のライブラリ

「¥Driver¥Driver\_TCP¥API\_Lib」内のファイルは WM-RP-0xS を使用するに当たって Redpine Signals 社が用意したライブラリとなります。

サンプルプログラムは、このライブラリを使用しています。詳細は、「¥Driver¥Driver\_TCP¥Documentation」内の「RS9110-N-11-22\_24\_28\_SPI\_API\_Library\_Manual.pdf」を参照してください。

また、合わせて「RS9110-N-11-22\_24\_28-Software\_PRM.pdf」も参照してください。

※各 pdf 上で「5GHz」の設定、「14CH」の設定に関して記述がありますが、  
WM-RP-0xS は、「2.4GHz、1-13CH」にて技術基準適合証明を取得しております。  
よって「2.4GHz、1-13CH」以外の設定を行って動作させた場合 弊社は一切責任を負いませんのでご了承下さい。

#### 5. 3. 1 各種変数等

##### < 1 > rsi\_api 構造体

「¥src¥Applications¥MCU」フォルダ内「rsi\_global.h」ファイルの 1011 行目～1030 行目で定義されています。  
この構造体は、主に WM-RP-0xS を初期化する時のパラメータを格納する為の構造体となっています。

この構造体へのパラメータ設定は、

「¥src¥Applications¥MCU」フォルダ内「rsi\_config\_init.c」ファイルの「rsi\_init\_struct()」関数にて行われます。  
サンプルプログラムでは、「wm\_rp\_04s.c」ファイルの 81 行目にて、グローバル変数として「rsi\_api rsi\_strApi」の様に定義して使用しています。

##### < 2 > rsi\_uCmdRsp 構造体

「¥src¥Applications¥MCU」フォルダ内「rsi\_global.h」ファイルの 952 行目～981 行目で定義されています。  
この構造体は、主に WM-RP-0xS からのレスポンスを格納する為の構造体となっています。

サンプルプログラムでは、「wm\_rp\_04s.c」ファイルの 57 行目にて、static なグローバル変数として「static volatile rsi\_uCmdRsp uCmdRspFrame」の様に定義して使用しています。

この構造体ですが、WM-RP-0xS からの何らかのレスポンスを状況に応じて適切な変数に格納する様に作られています。

例) 「band」コマンドを実行する場合 以下の様な処理となります。

```
retval = rsi_band( rsi_strApi.band ); /* band コマンド送信 */
if( retval == RSI_SUCCESS ){
    RSI_RESPONSE_TIMEOUT( RSI_BANDTIMEOUT ); /* レスポンス待ち */
    rsi_read_packet( &uCmdRspFrame ); /* レスポンス取得 */
    rsi_clearPktIrq();
    if( uCmdRspFrame.mgmtResponse.rspCode[0] == RSI_RSP_BAND ){ /* band のレスポンスコード=0x97 */
        if( uCmdRspFrame.mgmtResponse.status == 0x00 ){
            } else {
            }
        }
    }
}
```

※詳細は、「RS9110-N-11-22\_24\_28\_SPI\_API\_Library\_Manual.pdf」ファイルの「2.1.8 Read Packet data structure (From module):」を参照して下さい。

### < 3 > タイマ用変数

「wm\_rp\_04s.c」ファイルの80行目～98行目で以下の3つのグローバル変数を定義しています。

- uint32 rsi\_spiTimer1
- uint32 rsi\_spiTimer2
- uint32 rsi\_spiTimer3

この3つの変数は、「¥src¥Applications¥MCU」フォルダ内「rsi\_global.h」ファイルの80行目～98行目で定義されている以下のマクロに影響します。

- #define RSI\_INC\_TIMER\_2 rsi\_spiTimer2++
- #define RSI\_INC\_TIMER\_1 rsi\_spiTimer1++
- #define RSI\_INC\_TIMER\_3 rsi\_spiTimer3++
- #define RSI\_RESET\_TIMER1 rsi\_spiTimer1=0
- #define RSI\_RESET\_TIMER2 rsi\_spiTimer2=0
- #define RSI\_RESET\_TIMER3 rsi\_spiTimer3=0

これら6個のマクロを使用する場合は、「uint32 rsi\_spiTimer1」「uint32 rsi\_spiTimer2」「uint32 rsi\_spiTimer3」変数をサンプルプログラムの様にグローバル変数として定義して下さい。

実際に、「tmr.c」ファイルの「cmt\_init()」関数内で、

- RSI\_RESET\_TIMER1
- RSI\_RESET\_TIMER2
- RSI\_RESET\_TIMER3

を使用しており、また「tmr.c」ファイルの「Excep\_CMTU0\_CMT0 ()」割り込み関数内で

- RSI\_INC\_TIMER\_2
- RSI\_INC\_TIMER\_1
- RSI\_INC\_TIMER\_3

を使用しております。

この変数が使用されるのは、タイムアウト処理が殆どです。

例えば、「¥src¥Applications¥MCU」フォルダ内「rsi\_global.h」ファイルの151行目の

「#define RSI\_RESPONSE\_TIMEOUT(A)」マクロでは「RSI\_RESET\_TIMER3」を使ってタイムアウト処理を行っています。

### < 4 > INTR 割り込み状態格納変数

「wm\_rp\_04s.c」ファイルの79行目にて、グローバル変数として「volatile rsi\_intStatus rsi\_strIntStatus」を定義しています。

ライブラリでは、INTR 割り込み時のステータスを「rsi\_strIntStatus」変数に格納するようになっています。



## 5. 3. 2 無線通信用の主なコマンドファイル

※ 「¥src¥API\_Lib」フォルダ内の主なコマンドファイルに関して記述します。

## &lt; 1 &gt; Band コマンド

コマンド説明	
説明	使用周波数帯域の設定を行います。
コマンドファイル名	rsi_spi_band.c
使用方法	引数に設定値を指定する。
パラメータ説明	0:2.4GHz
レスポンスコード	0x97
レスポンス詳細	-

## &lt; 2 &gt; Init コマンド

コマンド説明	
説明	Band コマンド送信後に要求されるコマンドです。
コマンドファイル名	rsi_spi_init.c
使用方法	rsi_band()関数処理後、rsi_init()関数を呼ぶ
パラメータ説明	-
レスポンスコード	0x94
レスポンス詳細	-

## &lt; 3 &gt; Scan コマンド

コマンド説明	
説明	指定したチャンネルを走査します。
コマンドファイル名	rsi_spi_scan.c
使用方法	予め、rsi_api 構造体で定義した変数にスキャンするチャンネルと SSID を設定しておきます。
パラメータ説明	rsi_api 構造体で定義した変数ポインタ指定
レスポンスコード	0x95
レスポンス詳細	指定した SSID からのパラメータが、rsi_uCmdRsp 構造体で定義した変数の rsi_scanResponse 構造体内に格納されます。

## &lt; 4 &gt; Join コマンド

コマンド説明	
説明	SSID への接続を行います。
コマンドファイル名	rsi_spi_join.c
使用方法	<p>予め、rsi_api 構造体で定義した変数に</p> <ul style="list-style-type: none"> <li>・ ネットワークタイプ</li> <li>・ セキュリティタイプ</li> <li>・ 送信データレート</li> <li>・ 送信出力パワー</li> <li>・ セキュリティキー</li> <li>・ 接続する SSID</li> <li>・ IBSS モード</li> <li>・ IBSS チャンネル</li> </ul> <p>を設定しておきます</p>
パラメータ説明	rsi_api 構造体で定義した変数ポインタ指定
レスポンスコード	0x96
レスポンス詳細	-

## &lt; 5 &gt; IP Config コマンド

コマンド説明	
説明	IP アドレス等を設定します。
コマンドファイル名	rsi_spi_ipparam.c
使用方法	<p>予め、rsi_api 構造体で定義した変数に</p> <ul style="list-style-type: none"> <li>・ WM-RP-0xS の IP アドレス</li> <li>・ ネットマスク</li> <li>・ ゲートウェイ</li> <li>・ DNS サーバ</li> <li>・ DHCP の使用、未使用</li> </ul> <p>を設定しておきます。</p>
パラメータ説明	rsi_api 構造体で定義した変数ポインタ指定
レスポンスコード	-
レスポンス詳細	-

## &lt; 6 &gt; Socket タイプコマンド

コマンド説明	
説明	Socket タイプを設定します。
コマンドファイル名	rsi_spi_socket.c
使用方法	<p>予め、rsi_api 構造体で定義した変数に</p> <ul style="list-style-type: none"> <li>・ WM-RP-0xS の PORT 番号</li> <li>・ 接続先の PORT 番号</li> <li>・ TCP or UDP, サーバ or クライアント</li> <li>・ 接続先の IP アドレス</li> </ul> <p>を設定しておきます。</p>
パラメータ説明	rsi_api 構造体で定義した変数ポインタ指定
レスポンスコード	0x02
レスポンス詳細	<p>この処理が成功すると、 rsi_uCmdRsp 構造体で定義した変数の rsi_socketFrameRcv 構造体内にソケットディスクリプタが格納されます。</p> <p>ソケットディスクリプタは、無線データ送受信等において必要になるパラメータですのでグローバル変数等に保持して使用して下さい。</p>

## &lt; 7 &gt; Send data コマンド

コマンド説明	
説明	オープンしたソケットからデータを送信します。
コマンドファイル名	rsi_spi_send_data.c
使用方法	Socket タイプコマンドで取得したソケットディスクリプタを指定して関数を呼び出します。
パラメータ説明	<p>以下のパラメータを指定します。</p> <ul style="list-style-type: none"> <li>・ ソケットディスクリプタ</li> <li>・ 送信データバッファ</li> <li>・ 送信データサイズ</li> <li>・ TCP or UDP の指定</li> </ul>
レスポンスコード	-
レスポンス詳細	-

## &lt; 8 &gt; Receive data コマンド

コマンド説明	
説明	WM-RP-0xS からの受信を行います。
コマンドファイル名	rsi_spi_read_packet.c
使用方法	状況に応じて WM-RP-0xS からの受信を行います。 1、INTR 割り込みが入った時 WM-RP-0xS 内部レジスタからどういった割り込みが入ったかステータスを取得します。 これは、「rsi_spi_interrupt_handler.c」ファイルの「rsi_intHandler()」関数内にて「rsi_irqstatus()」関数を呼ぶ事で行っています。 2、接続先からの無線データか？ WM-RP-0xS 内部レジスタの内容に応じて無線データか？ 或いは WM-RP-0xS のレスポンスデータか？ に応じて WM-RP-0xS からの受信処理を行います。
パラメータ説明	rsi_uCmdRsp 構造体で定義した変数
レスポンスコード	0x07
レスポンス詳細	この処理が成功すると、rsi_uCmdRsp 構造体で定義した変数にレスポンスデータ、もしくは無線データが格納されます。

## &lt; 9 &gt; Close socket コマンド

コマンド説明	
説明	オープンしたソケットのクローズを行います。
コマンドファイル名	rsi_spi_socket_close.c
使用方法	rsi_socket_close() 関数にクローズするソケットディスクリプタを指定して呼び出します。
パラメータ説明	クローズするソケットディスクリプタ
レスポンスコード	0x06
レスポンス詳細	-

## &lt; 10 &gt; Disassociate コマンド

コマンド説明	
説明	接続しているアクセスポイントからの切断を行います。
コマンドファイル名	rsi_spi_disconnect.c
使用方法	rsi_disconnect () 関数を呼び出します。
パラメータ説明	-
レスポンスコード	0x0C
レスポンス詳細	-

## &lt; 1 1 &gt; Remote close

コマンド説明	
説明	接続先が切断した時の処理です。
コマンドファイル名	-
使用方法	rsi_uCmdRsp 構造体で定義した変数の rsi_recvRemTerm 構造体のレスポンスデータを確認します。
パラメータ説明	-
レスポンスコード	0x05
レスポンス詳細	接続先が切断（ソケットクローズなど）した場合、INTR 割り込みによって、rsi_recvRemTerm 構造体に 0x05 が格納されます。 接続先が切断した事を知る為に、常にこの変数を監視するようにプログラムして下さい。

## &lt; 1 2 &gt; WM-RP-0xS 初期化用コマンド

コマンド説明	
説明	WM-RP-0xS 初期化コマンドです。 電源 ON 後、必ず最初に 1 度実行して下さい。
コマンドファイル名	rsi_spi_iface_init.c
使用方法	rsi_spi_iface_init() 関数を呼び出します。
パラメータ説明	-
レスポンスコード	0x58
レスポンス詳細	

## &lt; 1 3 &gt; WM-RP-0xS 各種設定用ファイル

コマンド説明	
説明	WM-RP-0xS に対して実行する各種設定、コマンド等の値を変数に設定します。
コマンドファイル名	rsi_config_init.c (「¥src¥Applications¥MCU」フォルダ内)
使用方法	rsi_init_struct() 関数を呼び出します。
パラメータ説明	rsi_api 構造体で定義した変数ポインタを指定します。
レスポンスコード	-
レスポンス詳細	-

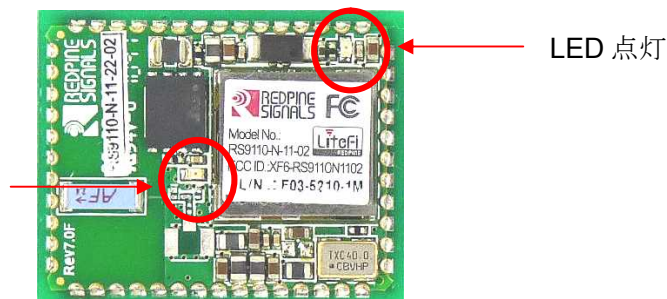
## &lt; 1 4 &gt; bootloader 処理

コマンド説明	
説明	WM-RP-0xS に対して bootloader 処理を行います。
コマンドファイル名	rsi_bootloader.c
使用方法	<p>&lt; 1 2 &gt; WM-RP-0xS 初期化用コマンドの次に必ず rsi_bootloader() 関数を呼び出します。</p> <p>また、bootloader() 関数内では以下のファイルを include して使用しています。</p> <p>「¥src¥API_Lib¥Firmware」フォルダ内</p> <ul style="list-style-type: none"> <li>● sbinst1</li> <li>● sbinst2</li> <li>● sbdata1</li> <li>● sbdata2</li> </ul> <p>※rsi_global.h ファイル内の 「#define RSI_LOAD_SBDATA2_FROM_HOST」定義の値を必ず「1」 にしておいて下さい。</p>
パラメータ説明	-
レスポンスコード	-
レスポンス詳細	-

## 6. WM-RP-0xS 上の LED に関して

bootloader 終了後、WM-RP-0xS 上の以下の部分の LED が点灯します。

こちらの LED は未使用の為  
点灯する事はありません。



## 7. サンプルプログラムに関して

### 7. 1 ネットワークの設定

#### <1> ポート番号

「¥src¥Applications¥MCU」フォルダ内「rsi\_config\_init.c」ファイル  
37行目～39行目にて、ポート番号の設定をしています。  
必要に応じて以下の定義の変更をして下さい。

- #define RSI\_MODULE\_SOCKET\_ONE WM-RP-0xS のポート番号
- #define RSI\_TARGET\_SOCKET\_ONE 接続先相手のポート番号

#### <2> TCP or UDP , Server or Client

「¥src¥Applications¥MCU」フォルダ内「rsi\_config\_init.c」ファイル  
41行目～42行目にて、TCP or UDP , Server or Client の設定をしています。  
必要に応じて以下の定義の変更をして下さい。  
※この定義に指定可能な値は、「¥src¥API\_Lib」フォルダ内「rsi\_spi\_api.h」ファイルの  
390行目～394行目の「SOCKET Defines」定義を使用して下さい。

- #define RSI\_SOCKET\_TCP\_CLIENT\_TYPE WM-RP-0xS の TCP or UDP , Server or Client の設定

#### <3> インフラストラクチャ or アドホック

「¥src¥Applications¥MCU」フォルダ内「rsi\_config.h」ファイル  
42行目～44行目にて、インフラストラクチャ or アドホックの設定をしています。  
必要に応じて以下の定義の変更をして下さい。  
※この定義に指定可能な値は、「¥src¥API\_Lib」フォルダ内「rsi\_spi\_api.h」ファイルの  
420行目～424行目の「NETWORK Type」定義を使用して下さい。

- #define RSI\_NETWORK\_TYPE WM-RP-0xS インフラストラクチャ or アドホック

#### <4> WM-RP-0xS の IP アドレス

「¥src¥Applications¥MCU」フォルダ内「rsi\_config.h」ファイル  
54行目～55行目にて、WM-RP-0xS の IP アドレスを設定をしています。  
必要に応じて以下の定義の変更をして下さい。

- #define RSI\_MODULE\_IP\_ADDRESS WM-RP-0xS の IP アドレス



## < 5 > ゲートウェイ

「¥src¥Applications¥MCU」フォルダ内「rsi\_config.h」ファイル  
57 行目～58 行目にて、ゲートウェイのアドレスを設定をしています。  
必要に応じて以下の定義の変更をして下さい。

- #define RSI\_GATEWAY ネットワーク接続 ゲートウェイ

## < 6 > 接続先相手の IP アドレス

「¥src¥Applications¥MCU」フォルダ内「rsi\_config.h」ファイル  
60 行目～61 行目にて、接続先相手の IP アドレスを設定をしています。  
必要に応じて以下の定義の変更をして下さい。

- #define RSI\_TARGET\_IP\_ADDRESS 接続先相手の IP アドレス

## < 7 > ネットマスク

「¥src¥Applications¥MCU」フォルダ内「rsi\_config.h」ファイル  
63 行目～64 行目にて、ネットマスクのアドレスを設定をしています。  
必要に応じて以下の定義の変更をして下さい。

- #define RSI\_NETMASK ネットワーク接続 ネットマスク

## < 8 > 無線アクセスポイントセキュリティタイプ

「¥src¥Applications¥MCU」フォルダ内「rsi\_config.h」ファイル  
66 行目～67 行目にて、無線アクセスポイントのセキュリティタイプを設定をしています。  
必要に応じて以下の定義の変更をして下さい。  
※この定義に指定可能な値は、「¥src¥API\_Lib」フォルダ内「rsi\_spi\_api.h」ファイルの  
412 行目～417 行目の「SECURITY Type Defines」定義を使用して下さい。

- #define RSI\_SECURITY\_TYPE 無線アクセスポイントセキュリティタイプ

## < 9 > 無線アクセスポイントセキュリティキー

「¥src¥Applications¥MCU」フォルダ内「rsi\_config.h」ファイル  
71 行目～76 行目にて、無線アクセスポイントのセキュリティキーを設定をしています。  
必要に応じて以下の定義の変更をして下さい。

- #define RSI\_PSK 無線アクセスポイントセキュリティキー

### < 1 0 > 無線アクセスポイントスキャン SSID

「¥src¥Applications¥MCU」フォルダ内「rsi\_config.h」ファイル  
86 行目～87 行目にて、無線アクセスポイントのスキャン SSID を設定をしています。  
必要に応じて以下の定義の変更をして下さい。

- #define RSI\_SCAN\_SSID 無線アクセスポイントスキャン SSID

### < 1 1 > 無線アクセスポイントスキャン CH

「¥src¥Applications¥MCU」フォルダ内「rsi\_config.h」ファイル  
91 行目にて、無線アクセスポイントのスキャン CH を設定をしています。  
必要に応じて以下の定義の変更をして下さい。

- #define RSI\_SCAN\_CHANNEL 無線アクセスポイントスキャン CH

### < 1 2 > 無線アクセスポイント参加 (Join) SSID

「¥src¥Applications¥MCU」フォルダ内「rsi\_config.h」ファイル  
93 行目～94 行目にて、無線アクセスポイントの参加 (Join) SSID を設定をしています。  
必要に応じて以下の定義の変更をして下さい。

- #define RSI\_JOIN\_SSID 無線アクセスポイント参加 (Join) SSID

### < 1 3 > DHCP 使用 or 未使用

「¥src¥Applications¥MCU」フォルダ内「rsi\_config.h」ファイル  
135 行目～136 行目にて、DHCP 使用 or 未使用を設定をしています。  
必要に応じて以下の定義の変更をして下さい。  
※この定義に指定可能な値は、「¥src¥API\_Lib」フォルダ内「rsi\_spi\_api.h」ファイルの  
386 行目～387 行目の「TCP/IP Defines」定義を使用して下さい。

- #define RSI\_IP\_CFG\_MODE DHCP 使用 or 未使用

## 7. 2 環境依存部（ハードウェア等）

サンプルプログラムは、WM-RP-0xS の制御 CPU として RX111 を使用しました。  
WM-RP-0xS をお使いになる環境に合わせて、以下の処理を変更して下さい。

### < 1 > エンディアン

「¥src¥Applications¥MCU」フォルダ内「rsi\_global.h」ファイル  
53 行目～54 行目にて、エンディアンを設定をしています。  
必要に応じて以下の定義の変更をして下さい。

- #define RSI\_LITTLE\_ENDIAN 0:Big endian , 1:Little endian

### < 2 > WM-RP-0xS レスポンス待ちタイムアウト処理定義

「¥src¥Applications¥MCU」フォルダ内「rsi\_global.h」ファイル  
76 行目～169 行目にて、WM-RP-0xS レスポンス待ちのタイムアウト処理定義をしています。  
必要に応じて各定義の値を変更して下さい。  
※タイムアウト処理は、WM-RP-0xS からのレスポンス待ちで使用します。  
通常であれば、レスポンスが返ってくるまでの待ちとなりますので、環境に合わせて値を設定して下さい。

- #define RSI\_TICKS\_PER\_SECOND この値をベースにタイムアウト処理が行われます。  
使用される制御 CPU に合わせて適宜変更して下さい。

### < 3 > SPI

「¥src¥API\_Lib」フォルダ内「rsi\_hal\_mcu\_spi.c」ファイル  
にて、SPI の処理を行っております。  
WM-RP-0xS を使用する環境に合わせて変更して下さい。  
また、「¥src¥API\_Lib」フォルダ内「rsi\_hal.h」ファイル  
41 行目～44 行目にて、SPI を使用する為の定義をしています。  
必要に応じて各定義の値を変更して下さい。

- #define RSI\_SPI\_SEND\_BYTE(A) SPI による 8bit ライト
- #define RSI\_SPI\_SEND\_4BYTE(A) SPI による 32bit ライト
- #define RSI\_SPI\_READ\_BYTE(B) SPI による 8bit リード
- #define RSI\_SPI\_READ\_4BYTE(B) SPI による 32bit リード

#### < 4 > 割り込み

「¥src¥API\_Lib」フォルダ内「rsi\_hal\_mcu\_interrupt.c」ファイルにて、割り込みの処理を行っております。  
WM-RP-0xSを使用する環境に合わせて変更して下さい。

※本サンプルプログラムでは、ポーリング処理（ポート入力）を行う事で WM-RP-0xS からの割り込み処理を行っております。  
サンプルプログラム「¥src」フォルダ内「wm\_rp\_04s.c」の「wm\_rp\_04s\_check\_intr\_level()」関数をメインから常に呼び出す事で、INTR 信号がローレベルかを確認しております。  
合わせて、「¥src¥API\_Lib」フォルダ内「rsi\_interrupt.c」ファイルも参考して下さい。具体的な割り込みの処理を記述しております。

#### < 5 > ハードウェアタイマ

「¥src¥API\_Lib」フォルダ内「rsi\_hal\_mcu\_timers.c」ファイルにて、ハードウェアタイマの処理をしています。  
必要に応じて処理を変更して下さい。

#### < 6 > I/O ポート

「¥src¥API\_Lib」フォルダ内「rsi\_hal\_mcu\_ioports.c」ファイルにて、I/O ポートの処理をしています。  
必要に応じて処理を変更して下さい。

※サンプルプログラムでは、使用しておりません。

## ご注意

- ・本文書の著作権は株式会社アルファプロジェクトが保有します。
- ・本文書の内容を無断で転載することは一切禁止します。
- ・本文書に記載されているサンプルプログラムの著作権は株式会社アルファプロジェクトが保有します。
- ・本文書に記載されている内容およびサンプルプログラムについての技術サポートは一切受け付けておりません。
- ・本サンプルプログラムに関して、ルネサスエレクトロニクスへのお問い合わせはご遠慮ください。
- ・本文書の内容およびサンプルプログラムに基づき、アプリケーションを運用した結果、万一損害が発生しても、弊社およびルネサスエレクトロニクスでは一切責任を負いませんのでご了承下さい。
- ・本文書の内容については、万全を期して作成いたしました。万一ご不審な点、誤りなどお気づきの点がありましたら弊社までご連絡下さい。
- ・本文書の内容は、将来予告なしに変更されることがあります。

## 商標について

- ・RX111 は、株式会社ルネサスエレクトロニクスの登録商標、商標または商品名称です。
  - ・RX は、株式会社ルネサスエレクトロニクスの登録商標、商標または商品名称です。
  - ・Cubesuite+は、株式会社ルネサスエレクトロニクスの登録商標、商標または商品名称です。
- 
- ・その他の会社名、製品名は、各社の登録商標または商標です。



株式会社アルファプロジェクト  
〒431-3114  
静岡県浜松市東区積志町834  
<http://www.apnet.co.jp>  
E-MAIL : [query@apnet.co.jp](mailto:query@apnet.co.jp)