目次

SMS を使う (MA-E350/F□MA-E350/N)

u-blox 製モジュールを搭載した MA-E350 と v2.1.6β2 以降のファームウェアで、SMS の送受信[1]が可能となります。

SMS Server Tools 3 を使用しています。

設定

daemon の起動はデフォルトでは無効としてあります。有効にするため□/etc/default/smstools を編集します。

```
user1@plum:~$ sudo nano -w /etc/default/smstools
```

smstools

```
# smsd default file
ENABLE=no

# enable wakeup by SMS
WAKEUP=no

# Defines under which user smsd is running. This may need to be changed
# if using devices other then /dev/ttyS0
USER="smsd"
GROUP="dialout"

# Specifies pathes.
PIDFILE="/var/run/smstools/smsd.pid"
INFOFILE="/var/run/smstools/smsd.working"
```

“ENABLE=no” を “ENABLE=yes” に変更します。
SMS 着信による Suspend 状態からの Wakeup を有効にする[2]場合□“WAKEUP=no” を “WAKEUP=yes” に変更します。

daemon の起動

設定ファイルを更新したら、daemon を起動します[3]。

```
user1@plum:~$ sudo service smstools start
Starting SMS Daemon: smsd.
user1@plum:~$
```

起動しているか確認をしておきます。

```
user1@plum:~$ ps ax|grep sms
 1574 ?        Ss     0:00 /usr/sbin/smsd -p/var/run/smstools/smsd.pid -
i/var/run/smstools/smsd.working -usmsd -gdialout
 1575 ?        S      0:00 /usr/sbin/smsd -p/var/run/smstools/smsd.pid -
i/var/run/smstools/smsd.working -usmsd -gdialout
 1581 pts/1    S+     0:00 grep --color=auto sms
user1@plum:~$
```

SMS の送受信

SMS の送信

SMS の送信は、

- 送信先電話番号[4]
- メッセージ

を設定したメッセージファイルを、SMS Server Tools 3 の送信ディレクトリ (/var/spool/sms/outgoing/) にコピーすることで実行されます。
もっとも簡単なメッセージファイルは、下記の書式となります。

message.txt

```
To: 080xxxxxxxx

Hello! from MA-E3xx.
```

※ より細かいオプションは、SMS Server Tools 3 - SMS file format を参照してください。

メッセージファイルを作成したら、実際に送信してみます。

```
user1@plum:~$ sudo cp msg.txt /var/spool/sms/outgoing/
user1@plum:~$
```

ログファイル (/var/log/smstools/smsd.log) には次のように記録され、送信が成功しました。

```
2014-07-30 13:56:19,5, smsd: Moved file /var/spool/sms/outgoing/msg.txt to
/var/spool/sms/checked
2014-07-30 13:56:23,5, GSM1: sent, Message: Hello! from MA-E3xx.=>Hello!
from MA-E3xx., l=20.
2014-07-30 13:56:26,5, GSM1: SMS sent, Message_id: 43, To: 080xxxxxxxx,
sending time 5 sec.
```

送信されたファイルは、送信時の情報が付加され、送信済みディレクトリ (/var/spool/sms/sent/) へ移動されます。

```
user1@plum:~$ sudo cat /var/spool/sms/sent/msg.txt
To: 080xxxxxxxx
Modem: GSM1
Sent: 14-07-30 13:56:26
IMSI: 44xxxxxxxxxxxxx

Hello! from MA-E3xx.
user1@plum:~$
```

SMS の受信

SMS の受信には、2つの動作があります。

- 受信したメッセージが、受信ディレクトリ (/var/spool/sms/incoming/) へ保存される
- メッセージ受信後、指定したハンドラ (スクリプト/実行ファイル) を実行する

SMSの受信をきっかけとして、何らかの動作をさせたい場合、ハンドラを設定するのが良いと思います。まずは、単純にファイルとして保存される動作を確認してみます。

受信内容の保存

SMS を受信すると、ログファイルには下のような記録がされます。

```
2014-07-30 14:17:08,5, GSM1: SMS received, From: 080xxxxxxxx
```

受信ディレクトリには、ファイルが作成されています。

```
user1@plum:~$ ls -l /var/spool/sms/incoming/
total 4
-rw-r--r-- 1 smsd smsd 232 Jul 30 14:17 GSM1.cEnCZH
```

```
user1@plum:~$
```

```
user1@plum:~$ cat /var/spool/sms/incoming/GSM1.cEnCZH
From: 080xxxxxxxx
From_TOA: 80 unknown, unknown
From_SMSC: 8190xxxxxxxx
Sent: 14-07-30 14:17:03
Received: 14-07-30 14:17:08
Subject: GSM1
Modem: GSM1
IMSI: 44xxxxxxxxxxxxx
Report: no
Alphabet: ISO
Length: 23

Test Reply from iPhone.user1@plum:~$
```

※ 最後に改行コードがないので、プロンプト (user1@plum:~$) が続いてしまっています。メッセージは “Test Reply from iPhone.” までです。

メッセージ受信ハンドラの設定

設定ファイル (/etc/smsd.conf) の “eventhandler” に設定します。

```
～～～ 略 ～～～
[GSM1]
device = /dev/ttyLISA1
incoming = yes
#pin =
baudrate = 19200
#incoming_utf8 = yes
decode_unicode_text = yes
#eventhandler=   <--- ここ
verify_pdu = no
sms_mode = 1
```

詳細な説明は、SMS Server Tools 3 - Event handler, Alarm handler にあります。
指定したスクリプト/プログラムを、イベントにより 2 ～ 3 個の引数付きで起動します。

- イベント (SENT, RECEIVED, FAILED, REPORT or CALL)
- SMS ファイル名
- 送信済みメッセージの message id (イベントが “SENT” の場合のみ)

メッセージハンドラの例

使い方としては、つぎのような用例が考えられると思います。

- SMS を受信したら PPP の発信を行う (IP 着信ができないMVNO SIM での、閉域網エミュレーション)。
- ネットワークのトラフィックを監視し、ターゲットとなるアドレスへの SYN を検知したら□SMS を送信、相手側のデバイスとトンネルを張る。

日本語メッセージの処理

v2.6.9rc2 から、python-gsmmodem パッケージを追加しました。
それを用いて□PDUから日本語メッセージをデコードすることができます。

まず□/etc/smsd.conf の下記項目を設定し、smstoolsを再起動しておきます。

```
store_received_pdu = 3
```

SMSを受信したときに作成されるファイルを、下記スクリプトで処理します。

受信したファイル

GSM1.3Eqc6l

```
From: 080xxxxxxxx
From_TOA: 80 unknown, unknown
From_SMSC: 8190xxxxxxxxx
Sent: 16-05-25 11:39:42
Received: 16-05-25 11:40:14
Subject: GSM1
Modem: GSM1
IMSI: 4401xxxxxxxxxx
Report: no
Alphabet: ISO
Length: 15
PDU: .....

□,□k{□Thogehoge
```

スクリプト

<sxh python toolbar:false; title:decodeSMS.py> #! /usr/bin/env python3

import sys import gsmmodem

def main(filename):

```
  f = open(filename, encoding='latin1')
```

```
  for line in f.readlines():
      if line.startswith("PDU:"):
          pdu = line.strip().split(" ")[1]
```

```
          info = gsmmodem.pdu.decodeSmsPdu(pdu)
          print(info['text'])
```

```
          break
```

if name == “main”:

```
  filename = sys.argv[1]
  main(filename)
```

</sxh>

実行例

```
root@plum:~# python3 decodeSMS.py GSM1.3Eqc6l
日本語にほんごhogehoge
```

1)
SMS 契約ありの SIM が必要です
2)
MA-E350/N で利用可MA-E350/F では機能しません。
3)
設定ファイル更新後の次回以降は自動で起動します
4)
マニュアルでは“international format without the leading +” となっていますが“8180xxxxxxxx” とすると送信できません。