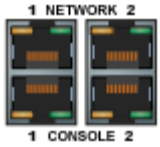
ThinkLogical SCS-R

Console Port 2 にモデムを接続する

株式会社 昌新 技術部

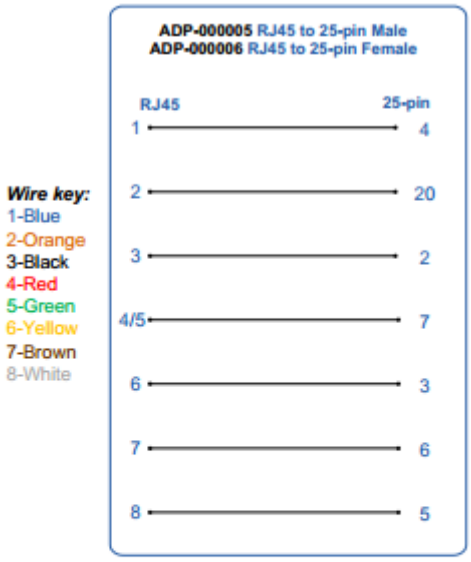
モデムの接続：

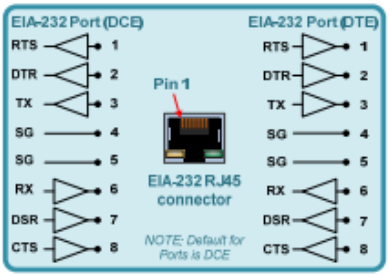
SCS-R には 2 つのネットワークポートと、2 つのシリアルコンソールが用意されています。公衆電話回線からアクセスする外付けモデムはシリアルコンソールポート 2 に接続します。

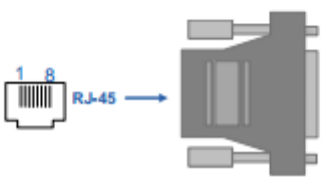


本体背面の CONSOLE ポート(下の 2 つ)

Console Port 1 は DCE、そして Console Port 2 は DTE 接続になっています。これはそれぞれ、シリアル端末とモデムの接続想定しているためであり、ここでは Console Port 2 に CAT5e ケーブルと、シリアルアダプターを使いモデム（DCE）を接続します。多くのモデムは 25 ピンメスコネクタを持ちますので、付属品の ADP-000005 アダプタと CAT-5 ケーブルを使って接続します。ADP-00005 の接続は次の通りです。







注意すべきは SCS のシリアル出力は TXD, DTR, RTS であり、入力は RXD, DSR, CTS の 6 本だけであることです。SCS のコンソールポートには DCD ピンがありません。モデムを制御する mgetty は DCD に依存せずに着信発信できますが、不意の回線断ないしログアウトを DCD 信号の変化から知ることができません。これについてはシェルなどのタイムアウトによる自動リセットで対応します。データの送受は TXD と RXD、フロー制御は RTS と CTS で行います。また DTR でモデムのリセットを、DSR でモデムが ON であることの確認を行います。 モデムと通信するデーモン mgetty は RING や CONNECT 文字列を認識して通信を制御します。よってモデムは Result Code を Verbose（V1）にセットしておき、自動着信しないようにします。MULTITECH モデムに次の設定をしておきます。
&D3E0&K3Q0S0=0V1X0&S0&W0

参考：SCS は minicom 通信ソフトを備えており、モデムの設定にも使えます。使い方は man minicom を参照ください。

inittab の設定と init による mgetty の起動：
工場出荷時に SCS の Console Port 2 は無効（モニタしていない）になっています。有効にするには /etc/inittab を変更して、mgetty がこのポートをモニタするようにします。次の行がありますので、コメントを外してください。

```
#M1:2345:respawn:/sbin/mgetty  -s 57600 -n 1 -D ttyS1
```
を
```
M1:2345:respawn:/sbin/mgetty  -s 57600 -n 1 -D ttyS1
```
にします。

この行では mgetty を 57,600 baud で起動し、RING を 1 回受け取ると ATA を発行して着信します。-x 0 をつけるとログを出さなくなります。-x 9 を付けると最も詳しいログを書き出します。mgetty にはいろいろなオプションが設定できますが、必ずしもすべて動作しません。上記設定から変更する際は、注意願います。また、オプションは `/etc/mgetty+sendfax/mgetty.config` に書くこともできます。

/etc/inittab を変更したら、次のコマンドを使い変更を保存し、init に inittab を再読込させます。
```
# save
# telinit q
```

mgetty の詳細はマニュアルページ man mgetty と mgetty のウェブサイトを参照くださ

い。
http://mgetty.greenie.net/

mgetty のランタイム設定ファイルは
/etc/mgetty+sendfax/mgetty.config
この中でいくつか設定しておいた方がよい項目があります。初期値では、DTR を落とした直後に初期化文字列を送ってしまいますので、遅延を入れます。それには末尾に次の行を加えます。¥d¥d¥d が遅延です。

```
port ttyS1
init-chat  "" ¥d¥d¥dATQ0V1E0H0 OK ATS0=0Q0&D3&C1 OK
```

また必要に応じて issue ファイルをここに指定しておきます。Issue は接続時に端末に表示されます。

```
issue-file /etc/mgetty+sendfax/mgissue
```

mgissue ファイルの中身は次のようなものです

```
|-----------------------------------------------------------------|
| This system is for the use of authorized users only.           |
| Individuals using this computer system without authority, or in |
| excess of their authority, are subject to having all of their   |
| activities on this system monitored and recorded by system      |
| personnel.                                                       |
|                                                                 |
| In the course of monitoring individuals improperly using this   |
| system, or in the course of system maintenance, the activities  |
| of authorized users may also be monitored.                      |
|                                                                 |
| Anyone using this system expressly consents to such monitoring  |
| and is advised that if such monitoring reveals possible         |
| evidence of criminal activity, system personnel may provide the |
| evidence of such monitoring to law enforcement officials.       |
|-----------------------------------------------------------------|
```

着信後に login 以外のプログラム（たとえば callback）を実行するには以下のファイルに書きます。

```
/etc/mgetty+sendfax/login.config
```

ログは以下のファイルに書き出されます。
/var/log/mgetty.log.ttyS1

上記の設定では、モデムに着信すると mgetty によりログインプロンプトが表示され、ユーザーID がユーザーにより入力されると、mgetty は login を起動して後を委ねます。ユーザーはパスワードを入力して、login が認証を確認し、ユーザーのシェルを起動して後を委ねます。

以上の設定が終わると、モデムを経由して SCS にユーザーID とパスワードを使ってログインすることができます。

Callback の設定：
セキュリティーの向上のために、あるいは発信元を SCS とするために SCS から指定する電話番号に電話させることができます。これを callback と呼びます。

Callback では、ユーザーはコールバックユーザーID と呼ばれる、通常のユーザーID ではない、特別なユーザーID でログインして、通常の login プロセスの代わりに callback プロセスを起動します。コールバックユーザーID は /etc/mgetty+sendfax/login.config に作ります。その後、callback がユーザーに電話をかけます。ユーザーは自身のモデムに対して ata を入力して SCS からのコールに着信します。

login.config の初期値の動作は、次の指定と同じものです。
```
*         - - /bin/login @
```
ここで、*は任意のユーザーID であり、@は mgetty に入力されたユーザーID を意味します。すなわち login 起動して、ユーザーID を渡します。この行がなくても、初期値としてこの行と同じ動作をします。コールバックの行を login.config に追加する際は、この行の上に指定することに注意してください。以下に callback を起動する行の例を示します。

コールバックと通常のモデムからのログインを併用する場合
```
# callback user ID
!version 2
calltokyo  N - - /usr/sbin/callback -S 0312341234
callnagoya N - - /usr/sbin/callback -S 0521231234
*          - - - /bin/login @
```

コールバック以外のモデムからのログインを禁止する場合

```
# callback user ID
!version 2
calltokyo  N - - /usr/sbin/callback -S 0312341234
callnagoya N - - /usr/sbin/callback -S 0521231234
*          Y - - /bin/login @
*          - - - /bin/false
```

ここで、

!version 2 Version 2 の書式を使うことの宣言です。

Y は callback 中にのみこの行が有効になることを意味します。

-S オプションは、着信したライン（モデム）からコールバックすることを意味します。また、コールバックする電話番号を書かないと、任意の番号を指定できるようになりますが、セキュリティー上好ましくありません。

Callback のログは、

/var/log/mgetty.log.callback

に書き出されます。ログの例を以下に示します。

シェルのタイムアウトの設定：

先にも書いたとおり、SCS は DCD を監視できないため、不意の回線断やユーザーがログアウトせずに放置した場合に接続が続いているとして切断せずにいる可能性があります。これを防ぐには、シェルにタイムアウトを設定します。

Bash の例としては；

```
export TMOUT=300
```

とすることで、分間（300 秒）何も入力しないとログアウトして回線を切ります。これを、モデムログインするユーザーの.bashrc に書いておきます。全ユーザーに設定する場合は /etc/bashrc に書きます。ユーザーの.bashrc ないし、/etc/bashrc に書いた場合は、必ず save コマンドを実行して変更を保存します。

参考情報：

mgetty と callback の動作：

コールバックを使うにあたっては、init, mgetty, callback の動作の概要を理解しておくべきです。

Unix/Linux が起動する過程で、init プロセスが i/o デバイス（この場合シリアルポート）に getty が実行されていることを確認します。init は/etc/inittab を読んで、mgetty を起動すべきデバイスを知り、/etc/utmp にエントリを作ります（mgetty を手で起動しても起動できないのはこの過程が必要なためです)。そして、その行にあるオプションに従い mgetty プロセスを fork（swapwn）します。

mgetty が起動すると、まずそのポートにロックファイルが作られていないか確認します。もしロックファイルが見つかったら、それはそのポートが使用中であることを意味しますので、mgetty はロックファイルが消されるまで待ちます。プロセスを伴わないロックファイルは無視します。

ポートが空いていることが確認できると mgetty はロックファイルを作り、モデムを初期化して、ロックファイルを消去します。そして、何かが起こるのを待ちます。この時点では文字列を読み込むことはしません。単に poll()や select()で読み込むものがあるかどうかチェックするだけです。

ポートに対して文字が書き込まれるには 2 つの向きがあります。何らかの内部プロセスがモデムを使おうとしてそのポートに文字を書き込む場合と、モデムから文字を受け取る場合です。前者の場合は、mgetty はモデムを使おうとするプロセスが作ったロックファイルを見つけて待機状態になり、ロックファイルが消えると自身を終了して、また init が mgetty を起動します(respawn)。後者の場合、ロックファイルは無く、mgetty は電話に着信したと判断して、ロックファイルを作り、文字列を読み込みます。もし、その文字列に RING を見つけたら、ATA をモデムに送り電話に応答し、CONNECT を含むメッセージが来るのを待ちます。もし、相手が FAX だったら FAX_SPOOL_IN ディレクトリにファックスを保存します。もし、通常のモデムだったら/etc/issue をプリントしてから、ログインプロンプトを表示します。ログインの文字列を受け取ったら、/etc/mgetty+sendfax/login.config の設定に従い、/bin/login なり、その他のプロセスを起動してその後をゆだねます。login を起動した場合、パスワードを読み込んでそのユーザーのログインシェルを起動します。ロックファイルはそのまま残り、そのユーザーがログインしている間に他のユーザーがそのポートを使うことを防ぎます。

login.config ファイルに指定することで、mgetty は login 以外のプログラムを起動できます。起動できるプログラムには、例えば callback があります。

もし、mgetty が何らかの理由で中断すると、init はそのポートを再度初期化し（これが mgetty がすぐに終了せずにロックファイルが消えるのを待つ理由です)、新しい mgetty プロセスを起動します。そのプロセスが最後のログインが残したロックファイルを消します。

ロックファイルの取り扱いは分かりづらいですが、とても重要です。すべてのモデムを使うプロセスが一つのロックプロトコルに従うことが基本です。さもないと、あるプログラムはほかのプログラムがポートを使っていることに気づかずに、そのポートか発信しようとしてしまいます。通常、ロックファイルは/etc/var/lock/LCK..ttyS1 であり、そのポートを使っているプロセスの ID (PID) を含んでいます。ほかのプロセスもそのファイルを読むことができ、そのポートを使っているプロセスが実存するか、あるいは存在しないからそのロックファイルを消していいかどうかを判断します。もし、プロセスが別の場所にロックファイルを探しに行ったり、またあるプロセスは PID を ASCII で書くがほかのプロセスはバイナリで書いてしまうと、この仕掛けは機能しません。mgetty は ASCII でもバイナリでも読むことができますが、必ずしもすべてのプロセスがそうではありません。

参考までですが、/etc/var/lock に作られるロックファイルの内容を見ていると、その時点でどのプロセスがモデムのポートを使っているのかが分かります。コールバックでは
mgetty
mgetty
callback
mgetty
とロックしているプロセスが変わります。

Callback の動作は、init と mgetty の両方に依存しています。init が utmp ファイルを管理しますので、mgetty は/etc/inittab の指定に従い init により起動されます。callback は mgetty がモニタしているポートに接続されたモデムを使いダイアルアウトします。接続が確立すると、callback は mgetty にシグナル SIGUSR1 を送り、mgetty は同じシグナルを送り返してきます。ここで callback は終了して、以降を mgetty に委ねます。mgetty はユーザーID を尋ね、login.config の指定(あるいは初期設定)に従い/bin/login を起動します。login はパスワードを尋ねて、認証されるとユーザーシェルを起動します。

Callback プロセスの詳細は、man callback を参照ください。

/etc/inittab の書式：

inittab の通常の書式は次の通りです

<tt>:rlevel:<respawn|off>:/usr/local/sbin/mgetty [options] <device>

tt は短縮デバイス名です。

rlevel は run レベルです。

respawn|off とその次のエントリが実行するプログラムです。

option は mgetty のオプションです。通常は/etc/mgetty+sendfax/mgetty.config に書きます。

callback コマンドのオプション：

callback [-x<debuglevel>] [-V] [-l<modemlines>] [-m<initstring>] [-s<speed>] [-d] [-S] [phone-number]

-x デバッグレベル

-V バージョン番号を表示して終了

-d デーモンにならない

-l モデムライン

-m モデム初期化文字列

-s ポートのスピード（callback.config の speed と同じ）

-S ダイアルインと同じモデムをダイアルアウトに使う

ログの例：ハイライトは callback のログ、その他は mgetty のログ

- **Init が mgetty を起動、モデムを初期化して電話を待ちます**

11/26 18:24:32 yS1 waiting...

11/26 18:24:42 yS1 wfr: waiting for ``RING'' 電話が鳴りました

11/26 18:24:42 yS1 send: ATA[0d] ATA をモデムに送り着信します

11/26 18:24:42 yS1 waiting for ``CONNECT'' ** found ** 接続しました

11/26 18:25:02 yS1 send:

11/26 18:25:02 yS1 waiting for ``_'' ** found **

11/26 18:25:20 yS1 WARNING: starting login while DCD is low! DCD が接続されていないことの警告です。SCS では DCD を監視する方法がありません。

11/26 18:25:20 ##### data dev=ttyS1, pid=2153, caller='none', conn='', name='', cmd='/usr/sbin/callback', user='callme' mgetty の login プロンプトに対してユーザーID callme が入力されました。Callme は login.config ファイルにあるコールバックユーザーID に一致しますので、callback を起動します。

11/26 18:25:20 detaching from ctty... callback のために mgetty を再起動します。

--

11/26 18:25:22 yS1 mgetty: experimental test release 1.1.30-Dec16 mgetty が respawn しました。

11/26 18:25:22 yS1 check for lockfiles

11/26 18:25:22 yS1 locking the line

11/26 18:25:22 yS1 lowering DTR to reset Modem DTR を落としてモデムをリセットします。

11/26 18:25:23 yS1 send: ¥d¥d¥dATQ0V1E0H0[0d] mgetty.config にセットした初期化文字列を送ります。

11/26 18:25:24 yS1 waiting for ``OK'' ** found **

11/26 18:25:24 yS1 send: ATS0=0Q0&D3&C1[0d]

11/26 18:25:24 yS1 waiting for ``OK'' ** found **

11/26 18:25:25 yS1 waiting...

11/26 18:25:47 PID for mgetty on line ttyS1: 2155 callback が mgetty のポートを使いダイアルアウトします。

11/26 18:25:47 yS1 initializing modem...

11/26 18:25:47 yS1 send: ATQ0V1H0[0d]

11/26 18:25:47 yS1 waiting for ``OK'' ** found **

11/26 18:25:47 yS1 send: AT+FCLASS=0[0d]

11/26 18:25:47 yS1 waiting for ``OK'' ** found **

11/26 18:25:47 yS1 dialing *2...

11/26 18:25:47 yS1 mdm_send: 'ATD*2'

11/26 18:25:47 yS1 lock not made: lock file exists (pid=2154) callback プロセスが ttyS1 をロックしています

11/26 18:26:23 yS1 dialup: got 'CONNECT'

11/26 18:26:23 yS1 got CONNECT, success! callback で接続しました。

11/26 18:26:24 yS1 Got callback signal from pid=2154! callback からシグナルを受け取りました

11/26 18:26:24 yS1 stealing lock file from pid=2154 callback のロックを自分のものにします

11/26 18:26:28 yS1 WARNING: starting login while DCD is low!

11/26 18:26:28 ##### data dev=ttyS1, pid=2155, caller='none', conn='', name='', cmd='/bin/login', user='user1' ユーザーとしてログインします。 mgetty は login を起動します。login がパスワードを求めてきます。認証されるとユーザーシェルを起動します。

以上