

QNX 7.0 x86 32bit ランタイム環境構築方法

QNX ソフトウェアシステムズ

エンジニアリングサービス

2019 年 8 月 9 日

1. はじめに

本文書では Windows ホスト環境のみを用いて x86 CPU ターゲットハードウェアに QNX 7.0 の 32bit ランタイム環境をインストールする方法を説明します。

概要としては、Windows ホスト環境の作成、ターゲット用のファイルのアーカイブ、インストール用 USB メモリの作成となります。

2. 必要なハードウェアおよびソフトウェア

1) Windows PC

QNX SDP (Software Development Platform) 7.0 で正式にサポートされている Windows ホスト OS は以下の通りです。

- Windows 10 Professional 64-bit
- Windows 8.1 Professional 64-bit
- Windows 7 Professional 64-bit

2) USB メモリ

ランタイム環境をターゲットハードウェアにインストールするためのインストール USB メモリを作成します。容量は 2GB 以上であれば問題ありません。

3) ディスクイメージ書き込みソフトウェア

インストールディスクイメージを USB メモリに書き込むユーティリティーソフトウェアです。

一般的には下記のソフトウェアが知られていますが、同等の機能を持ったソフトウェアであればそれでも構いません。

- Win32 Disk Imager
- DD for Windows

4) サンプルファイル

この構築方法では本書と共にアーカイブされている以下のサンプルファイルを使用します。

- disk.cfg
- install-usb.build
- mkusbimage.sh
- root.build
- set-perm-link.sh
- targetfile.list
- x86_32.ahci.build, x86_32.eide.build
- etc folder

3. Windows ホスト環境の作成

Windows ホスト環境の作成には、まずは下記の QNX Download Center サイトから QNX Software Center - Windows Hosts をダウンロードする必要があります。

<http://www.qnx.com/download/group.html?programid=29178>

2019 年 8 月現在では Version 1.3.1 Build 201808201144 が最新ですが、更新があればその Version をダウンロードして下さい。

また、すでに Windows PC に QNX SDP 7.0 のホスト環境をインストール済みの場合は、「3.3 アップデートのインストール」のみ確認して下さい。

3.1 QNX Software Center のインストール

ダウンロードした下記の QNX Software Center インストーラーを、Installation Note に従ってインストールします。

- qnx-setup-201808201144-win.exe

3.2 QNX SDP 7.0 のインストール

QNX Software Center を起動すると、“Add Installation” から QNX SDP 7.0 がインストール可能になりますので、購入した QNX SDP 7.0 License Key を登録してインストールして下さい。

3.3 アップデートのインストール

QNX SDP 7.0 インストール後は QNX Software Center の “Check for Package Updates” から各種アップデートパッケージがインストール可能となりますので、必要に応じてインストールしてください。

4. BSP for Generic x86 のインストール

4.1 BSP for Generic x86 のダウンロード

QNX Software Center の “Install New Packages” をクリックし、QNX Software Development Platform > Board Support Packages から QNX SDP 7.0 BSP for Generic x86 (32-bit) を選択してインストールします。

BSP for Generic x86 には QNX SDP 7.0 リリース以後に発売された CPU チップセットや SoC のサポート等が含まれています。

2019 年 8 月現在では下記の BSP が最新となっていますが、今後も更新される可能性がありますので、できるだけ最新の BSP をインストールする事をお勧めします。

- BSP_x86_br-mainline_be-700_SVN848888_JBN5.zip

4.2 BSP for Generic x86 のビルド

BSP for Generic x86 のビルド方法には QNX Momentics IDE を使用した方法とコマンドプロンプトでコマンドラインによる方法があり、ここではコマンドプロンプトを使用した方法を説明します。

インストールされた BSP ファイルはデフォルトのインストールディレクトリの場合は C:¥Users¥(user name)¥qnx700¥bsp に保存されています。

このファイルを適当なフォルダ(ここでは user name が qnxusr であるとして、C:¥Users¥qnxusr¥x86bsp と仮定します)にコピーします。

次にコマンドプロンプトを開いて、次のコマンドを実行します。

(注: QNX SDP 7.0 がインストールされているディレクトリはデフォルトの C:¥Users¥(user name)¥qnx700 とします)

```
c:¥Users¥qnxusr> c:¥Users¥qnxusr¥qnx700¥qnxsdp-env. bat
```

```
c:¥Users¥qnxusr> cd x86bsp
```

```
c:¥Users¥qnxusr¥x86bsp> sh
```

```
sh-3.1$ unzip BSP_x86_br-mainline_be-700_SVN848888_JBN5.zip
```

```
sh-3.1$ make
```

これで c:¥Users¥qnxusr¥x86bsp¥install フォルダに BSP for Generic x86 にあらかじめ含まれていたバイナリーやソースコードからビルドされたバイナリーが集約されます。

4.3 BSP for Generic x86 の適用

BSP for Generic x86 をビルドしただけでは、生成されたバイナリーは開発環境のターゲット用のファイルにはコピーされていませんので、これらのバイナリーをホスト環境にコピーします。

```
sh-3.1$ cp -pR install/* $QNX_TARGET
```

これにより開発環境は BSP for Generic x86 が完全に適用された状態になります。

5. 起動イメージの作成

BSP for Generic x86 が適用された環境で起動イメージを作成します。

ここではビルドファイルの例として本書と共にアーカイブされているビルドファイルを使用します。

「2. 必要なハードウェアおよびソフトウェア」の「4) サンプルファイル」に示す、すべてのファイルを適当なフォルダ(ここでは C:\Users\%qnxusr%\runtime とします)にコピーして起動イメージを作成し、ターゲットハードウェア用の起動イメージとします。

```
c:\Users\%qnxusr> c:\Users\%qnxusr%\qnx700\qnxsd-p-env.bat
```

(注:まだ qnxsd-p-env.bat を実行していない場合)

```
c:\Users\%qnxusr> cd runtime
```

```
c:\Users\%qnxusr%\runtime> sh
```

```
sh-3.1$ mkifs x86_32.ahci.build x86_32.ahci.ifs
```

```
sh-3.1$ mkifs x86_32.eide.build x86_32.eide.ifs
```

サンプルとして用意した 2 つのビルドファイルによって作成される起動イメージは、それぞれ次のような違いがあります。

x86_32.ahci.ifs : SATA disk を AHCI モードで起動します。

x86_32.eide.ifs : SATA disk を EIDE モードで起動します。

6. ランタイム環境に必要なファイルのアーカイブ

開発環境に含まれる、ターゲット用のファイルからターゲットハードウェアにインストールするファイルを収集してアーカイブします。

アーカイブするファイルは目的によって異なりますが、ここではサンプルとしてほぼすべてのファイルをアーカイブするための targetfile.list を用意しましたので、このファイルを使用して次の手順でアーカイブを作成します。

```
c:\Users\%qnxusr> c:\Users\%qnxusr%\qnx700\qnxsd-p-env.bat
```

(注:まだ qnxsd-p-env.bat を実行していない場合)

```
c:\Users\%qnxusr> cd runtime
```

```
c:\Users\%qnxusr%\runtime> sh
```

```
sh-3.1$ export RUNTIME=$PWD
sh-3.1$ cd $QNX_TARGET
sh-3.1$ tar -T $RUNTIME/targetfile.list -czf $RUNTIME/targetfile.tgz
sh-3.1$ cd $RUNTIME
```

7. インストール用 USB メモリの作成

7.1 起動イメージの作成

インストール用 USB メモリ用の起動イメージを作成するためのビルドファイル `install-usb.build` をサンプルとして用意しましたので、このファイルを使用して次の手順で起動イメージを作成します。

```
c:\¥Users¥qnxusr> c:\¥Users¥qnxusr¥qnx700¥qnxsdpc-env.bat
```

(注:まだ `qnxsdpc-env.bat` を実行していない場合)

```
c:\¥Users¥qnxusr> cd runtime
```

```
c:\¥Users¥qnxusr¥runtime> sh
```

```
sh-3.1$ mkifs install-usb.build install-usb.ifs
```

7.2 ディスクイメージの作成

`mkxfs` および `diskimage` ユーティリティを使用して、これまでに作成した起動イメージやアーカイブをディスクイメージに書き込みます。

サンプルとして用意した `disk.cfg`, `root.build`, `mkusbimage.sh` ファイルを使用して次の手順でディスクイメージを作成します。

```
c:\¥Users¥qnxusr> c:\¥Users¥qnxusr¥qnx700¥qnxsdpc-env.bat
```

(注:まだ `qnxsdpc-env.bat` を実行していない場合)

```
c:\¥Users¥qnxusr> cd runtime
```

```
c:\¥Users¥qnxusr¥runtime> sh
```

```
sh-3.1$ sh mkusbimage.sh
```

これによりディスクイメージファイル usb.img が作成されます。

7.3 USB メモリへの書き込み

Windows PC に USB メモリを装着し、Win32 Disk Imager 等のディスクイメージ書き込みソフトウェアを起動します。

イメージファイルに usb.img を指定し、書き込み先のディスクに装着した USB メモリを指定し、書き込みを実行します。

8. ターゲットハードウェアの準備

8.1 ターゲットハードウェアの BIOS 設定

1) ブートデバイスの選択

BIOS のブートデバイスの優先順位の設定で、USB デバイスを HDD よりも高く設定しておきます。

起動時に USB デバイスからブートするように選択できれば、それでも構いません。

2) SATA モードの選択

デフォルトでは AHCI になっている事が多く、そのまま構いませんが、AHCI で何か問題があれば、IDE 互換モードに設定してください。

3) USB Legacy Support 設定

BIOS によって名称が異なったり、そのメニューがない場合もありますが、この設定は USB Keyboard や Mouse を PS/2 マウスとしてエミュレートするかどうかを Enable / Disable します。

この機能を Enable にしていると起動出来ない、特定のドライバーが動作しない、等の問題が発生する事があるため、それが疑われる場合はこの機能を Disable にしてみてください。

注:この機能を Disable にすると、QNX 7.0 の USB ドライバーが動作するまで、USB Keyboard は使用できません。

8.2 ターゲットハードウェアの起動

ターゲットハードウェアにインストール用 USB メモリを装着して電源を入れ、USB メモリから QNX 7.0 がブートしてシェルが起動する事を確認します。

次にランタイム環境の入った USB メモリを以下のコマンドで /fs/usb0 にマウントして USB メモリの内容が表示される事を確認します。

```
# sh /scripts/mount-usb.sh  
  
# ls -lR /fs/usb0
```

8.3 ブロックドライバの起動

SATA モードを AHCI に設定している場合は、次のコマンドを実行します。

```
# sh /scripts/start-ahci.sh
```

SATA モードを IDE 互換モードに設定している場合は、次のコマンドを実行します。

```
# sh /scripts/start-eide.sh
```

上記のスクリプトは接続されている HD が1台の場合を想定しており、複数の HD が接続されている場合は、スクリプトを変更するか手動でマウントして下さい。

ブロックドライバ起動後に /dev の下に hd0 デバイスが作成されている事を確認します。

```
# ls /dev/hd0
```

8.4 パーティションの作成

QNX 7.0 をインストールするためのパーティションを作成します。

以下のコマンドで /dev/hd0 に存在するすべてのパーティションを消去し、最大サイズで type 179 (QNX6 filesystem) のパーティションを作成、ブータブルに設定して QNX ローダーを書き込んでいます。

```
# sh /scripts/format-hd0.sh
```

次に QNX6 ファイルシステムを /fs/hd0 にマウントし、/fs/hd0 に .boot ディレクトリが作成されている事を確認します。

```
# sh /scripts/mount-hd0.sh  
# ls -al /fs/hd0
```

これでターゲットハードウェアに QNX 7.0 をインストールするための準備ができました。

なお、これらのスクリプトでは QNX6 ファイルシステムを作成と /fs/hd0 へのマウントを行っていますが、パーティションの作成方法やフォーマットを変更したい場合はこのスクリプトを変更するか手動で行ってください。

9. ランタイム環境の作成

ターゲットハードウェアの準備が完了したら、次のスクリプトで USB メモリに格納されているランタイム環境に必要なファイルのアーカイブを一旦ターゲットハードウェアのハードディスクに展開し、ディレクトリ構成を調整してファイルパーミッションの変更やシンボリックリンクの設定を行います。

```
# sh /scripts/mk-runtime-img.sh
```

これにより直接ターゲットハードウェアのハードディスクにインストール可能なランタイム環境 runtime.tgz が作成され USB メモリにアーカイブされます。

このスクリプトは毎回実行する必要はなく、インストールするファイルが変更された場合等に行ってください。

また、引き続き「10. ランタイム環境のインストールと初期設定」を行う場合は、次のスクリプトでハードディスクを再フォーマットしておきます。

```
# sh /scripts/reformat-hd0.sh
```

10. ランタイム環境のインストールと初期設定

10.1 ランタイム環境のインストール

ランタイム環境のインストールを実行するスクリプトを実行します。

```
# sh /scripts/inst-runtime.sh
```

これによりターゲットハードウェアのハードディスクに起動可能な QNX 7.0 のランタイム環境のインストールされましたので、シャットダウンして電源を切り、USB メモリを取り外します。

```
# shutdown -b
```

ターゲットハードウェアの電源を入れ、QNX 7.0 が起動する際、2 つの起動イメージのどの起動イメージから起動するか、キーボードの上下矢印キーで選択することができます。

そして login: プロンプトが出る事を確認したら、root アカウント、パスワードも root でログインします。

10.2 初期設定

インストール直後の状態では、ネットワーク等のドライバーやネットワークサービスは動作していませんので、次の要領で初期設定を行って下さい。

1) デフォルトの起動イメージ

/.boot ディレクトリに複数の起動イメージを置いて、起動時に選択することができますが、何も操作を行わない場合は最も日付の新しい起動イメージから起動しますので、touch コマンドを使ってデフォルトの起動イメージの日付を更新しておきます。

例:

```
# touch /.boot/x86_32.ahci.ifs
```

2) root アカウントのパスワード

root アカウントのデフォルトのパスワードは“root”になっていますので、任意のパスワードに変更しておきます。

例:

```
# passwd root
```

(新パスワードの入力)

3) ユーザーアカウントの作成

スーパーユーザー以外の一般ユーザーアカウントが必要であれば、作成しておきます。

例: qnxusr アカウントの作成

```
# passwd qnxusr
```

(アカウント情報の入力)

4) ドライバー起動

/etc/rc.d/rc.drivers にサンプルのドライバー起動スクリプトを用意しましたので、必要なドライバーやサービスのコメントを外して、起動時に自動的に実行されるようにします。

記載されているドライバーの他に起動したいドライバーがあれば、新規に追加して下さい。

5) アプリケーション、サービスの起動

OS 起動時に自動的に実行したいアプリケーションやサービスがあれば、/etc/rc.d/rc.local ファイルに記述して下さい。

一般的にはネットワークスーパーサーバーの inetd やホスト PC の QNX Momentics IDE との接続に必要な qconn を記述しておきます。

例:

inetd

qconn

6) コンフィグレーションファイルの編集

/etc ディレクトリにある様々なコンフィグレーションファイルを編集します。

inetd によって ftp や telnet を起動したい場合は、/etc/inetd.conf ファイルの ftp, telnet 行のコメントを外しておきます。

11. Screen 機能の動作確認

QNX 7.0 のランタイム環境には Screen と呼ばれるグラフィック機能が含まれており、対応しているハードウェアであれば、その機能を試してみることができます。

ここでは、下記のハードウェアおよび Screen Board Support がインストールされている環境を例に説明します。

ハードウェア: Intel NUC 6i5SYH

QNX SDP 7.0 Screen Board Support Intel

STABLE(7.0 Build ID 6425) November 07,2018

1) 準備

Intel NUC 6i5SYH に QNX 7.0 ランタイム環境をインストールしておきます。

ランタイム環境の /etc/rc.d/rc.drivers ファイルを編集し、devc-serusb 行のコメントを外しておきます。

次のコマンドを実行し、デフォルトの graphics.conf ファイルを作成します。

```
# cp /usr/lib/graphics/intel-drm/graphics-nuc-6i5SYH.conf /usr/lib/graphics/intel-drm/graphics.conf
```

USB-Serial アダプタを用意し、NUC の USB ポートと PC のシリアルポートを接続します。

PC 側では Tera Term 等のターミナルソフトウェアを起動し、ボーレートを 115200 bps に設定します。

NUC の HDMI ポートに 1920x1080 解像度のディスプレイを接続しておきます。

2) DRM および Screen の起動

NUC で QNX 7.0 起動後、PC の Tera Term に “login: “ プロンプトが現れたら、root でログインし、下記のコマンドを実行します。

```
# drm-intel
```

```
# screen -c /usr/lib/graphics/intel-drm/graphics.conf
```

3) サンプルアプリケーションの実行

静止画の描画:

```
# display_image /etc/pictures/1920x1080-gray.png &
```

ディスプレイ全面にグレーの画像が表示されます。

ソフトウェア・ラスターサイズ:

```
# sw-vsync
```

黄色バックの画面に青色の縦線が左から右に移動する画像が表示されます。

GL ES2 動画:

```
# gles2-gears
```

青色バックに3つのギアが回転する動画が表示されます。

/usr/lib/graphics/intel-drm ディレクトリにはいくつかサンプルの graphics.conf ファイルがありますので、もし同じハードウェアがあれば NUC 6i5SYH のように graphics.conf ファイルを作成して Screen を試してみる事ができます。

また、drm-intel 起動後に drm-probe-displays コマンドを実行すると、DRM ドライバーが認識しているディスプレイの番号と対応解像度、パイプラインの数が表示されますので、この情報を元にサンプルの graphics.conf ファイルを変更し、手持ちのハードウェアに合わせ込むことで Screen を動作させる事もできます。

12. 変更履歴

2019 年 8 月 9 日 初版