

ISaGRAF PRO

C コンパイル型実行モードの実行方法

~ Cソースコードの生成と Target への組み込み ~

1. はじめに

ISaGRAF PRO では作成したプロジェクトを C ソースコードに変換できます。それをコンパイルし、カーネルに直接組み込むことによって、Target で高速に実行することができます。これを C コンパイル型実行モードと呼んでいます。

本書では例として実際に NT-Target でサンプルプロジェクト Project1 を C コンパイル型実行モードで実行するための方法を説明します。

2. 準備

ISaGRAF PRO には C コンパイルの機能がないため、コンパイラは別途用意する必要があります。NT - Target では Microsoft VC++ (Ver.5.0 以上) を使用します。

またリンクのためのライブラリは IO 開発キット (KDS-PIO) でサポートされていますので、これをインストールする必要があります。(開発ツールキット (KDS-PRDK) は IO 開発キットを含んでいます。)

◇ 注意：ワークベンチソフト (KDS-PRL, KDS-PRD, KDS-PRU) にはライブラリはサポートされていません。

3. 作業の流れ

実際の作業の流れは以下のようになります。

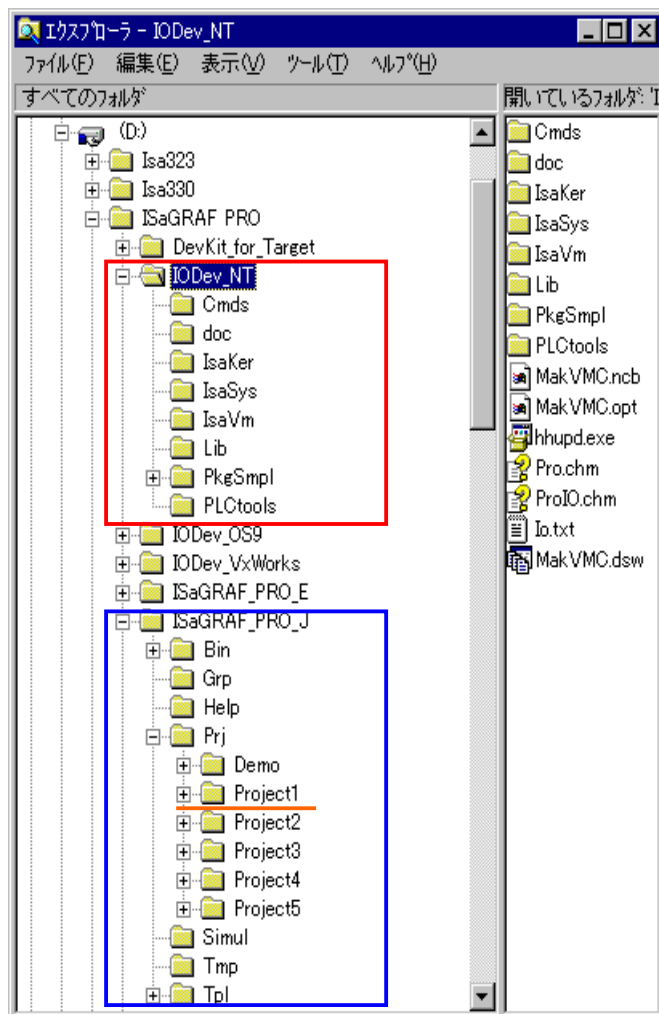
1 連の作業はすべてリソースごとに行います。

1. ターゲット / コード設定を C コンパイル型に変更します。
2. C ソースコードを生成します。
3. C ソースコードをコンパイルし、実行ファイルを生成します。
4. 実行ファイルおよびリソースファイルの名前を変更し、IVCxxxx.exe、ISPxxx01、ISPxxx03、ISPffe0d を準備します。
5. 上記ファイルをターゲットのフォルダに格納し、ターゲットを実行します。

4. 例) Project1 を C コンパイル実行モードで実行する

4.1. ディレクトリ構造

この例では以下のようなディレクトリ構造を取っています。ディレクトリ構造はインストール時の設定などによって異なることがありますので、実際には開発者の作業がしやすい構造をとってください。



〔ディレクトリ構造〕

ISaGRAF PRO ワークベンチは

D:¥ISaGRAF PRO¥ISaGRAF_PRO_J

にインストールされています。

ワークベンチで生成したファイルは

- ¥Prj¥Project1

の下に格納されます。

IO 開発キットは

D:¥ISaGRAF PRO¥IODev_NT

にインストールされています。

MakVMC.dsw は Make ファイルです。

Io.txt は IO 開発キットの ReadMe ファイルです。

- ¥IsaKer、IsaSys、Lib はライブラリです。

- ¥IsaVm は Make の雛形です。

- ¥Cmnds はターゲットの実行環境が格納されており、ここに実行ファイルを組み込みます。

4.2. Cソースコードの生成

4.2.1. ターゲット/コード設定

各リソースで、メニューから [プロジェクト] - [ターゲット/コード設定] を選びます。

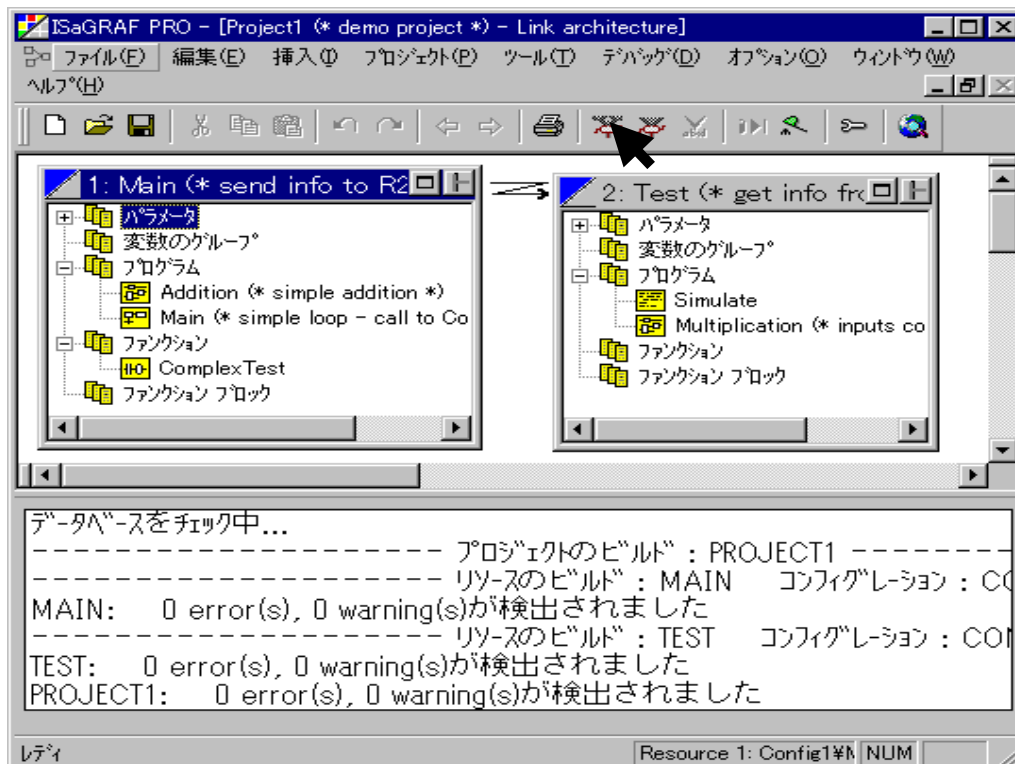
リソースウィンドウが開くので、[ターゲット/コード] で「Structured C source code」を選びます。このとき、「シミュレーション用コード」のチェックははずします。

また、[実行] で [オフライン] を選択します。



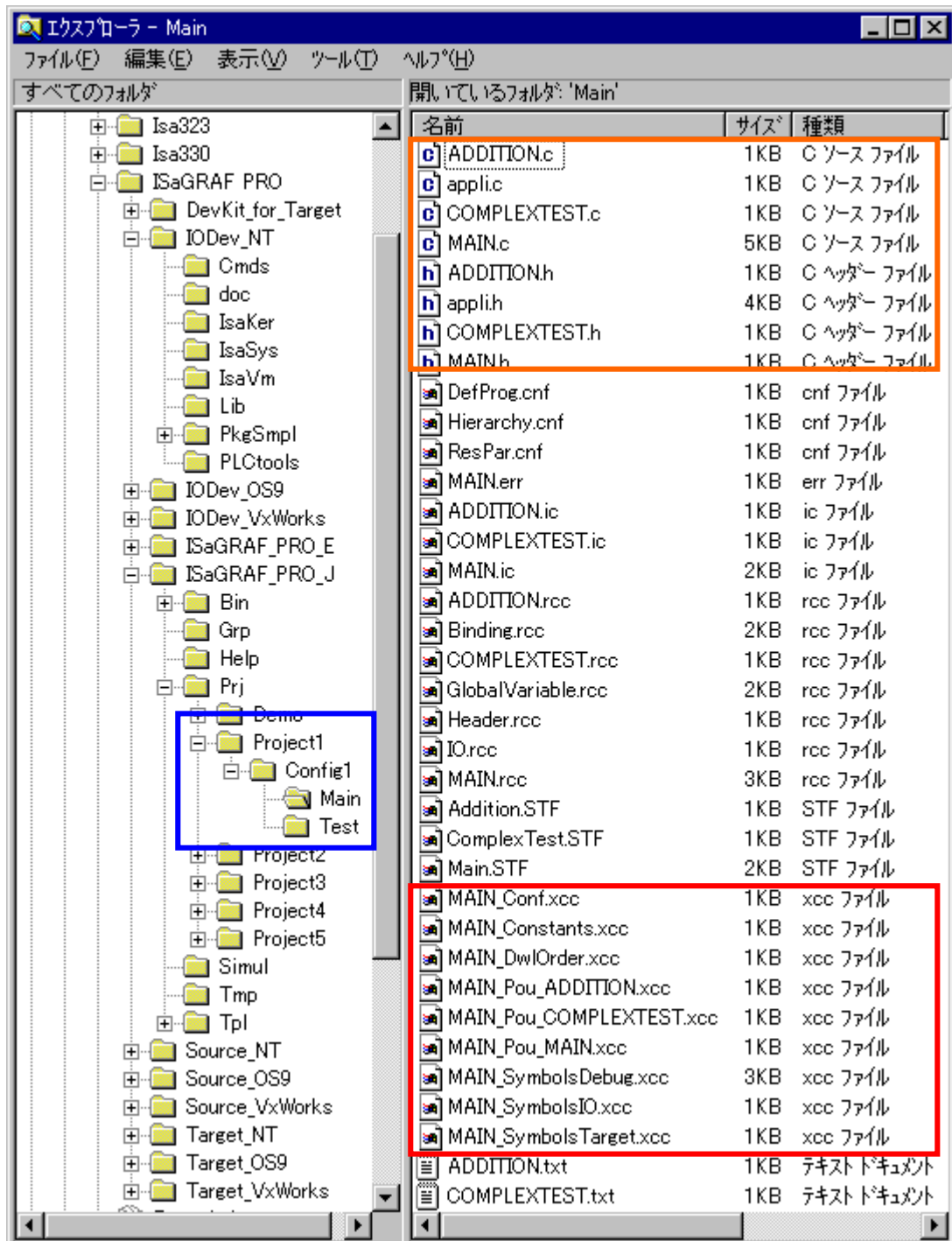
4.2.2. アプリケーションコードの生成

アプリケーションコードを生成します。



すると、- ¥Prj¥ Project1 の下に Config とリソース名のフォルダができ、生成されたコードが収納されます。

この中で使用するのは C ソースファイル、C ヘッダーファイル、xcc ファイルです。



4.3. Target への組み込み

4.3.1. ワークフォルダの準備

本例では作業をわかりやすくするために ...¥IODevKit_NT の下にワークフォルダを作り、実行ファイルのビルドなどの作業を行います。実際は、直接 -¥IsaVM 上で作業するなど、自由な方法で作業ができます。

...¥IODevKit_NT の下に プロジェクト名のワークフォルダ(WProject1)を作成します。

WProject1 フォルダに各リソース名のフォルダ (Main、TEST) を作成します。

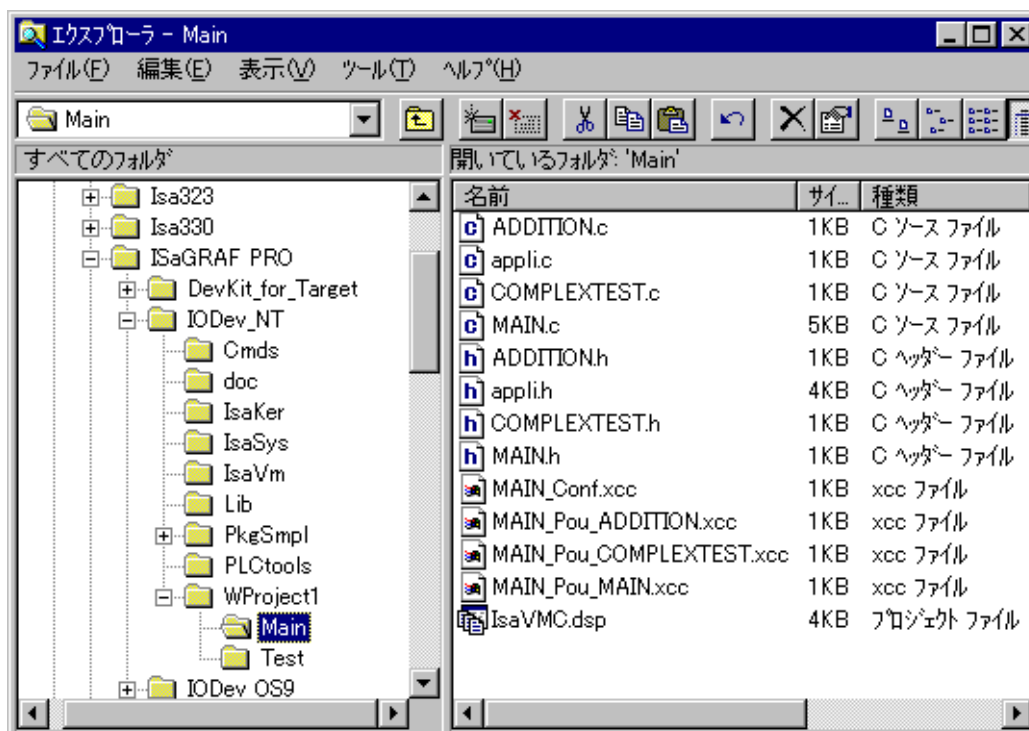
各リソースフォルダに...¥IsaVM¥IsaVMC.dsp をコピーします。最初は読取専用ファイルとなっているので、これを解除しておきます。

各リソースフォルダにワークベンチで生成した C ソースファイル、C ヘッダファイルコピーします。

各リソースフォルダにワークベンチで生成した xcc ファイルの内、以下のファイルをコピーします。

- ツ-ス名_Conf.xcc,
- ツ-ス名_Pou_プログラム名.xcc

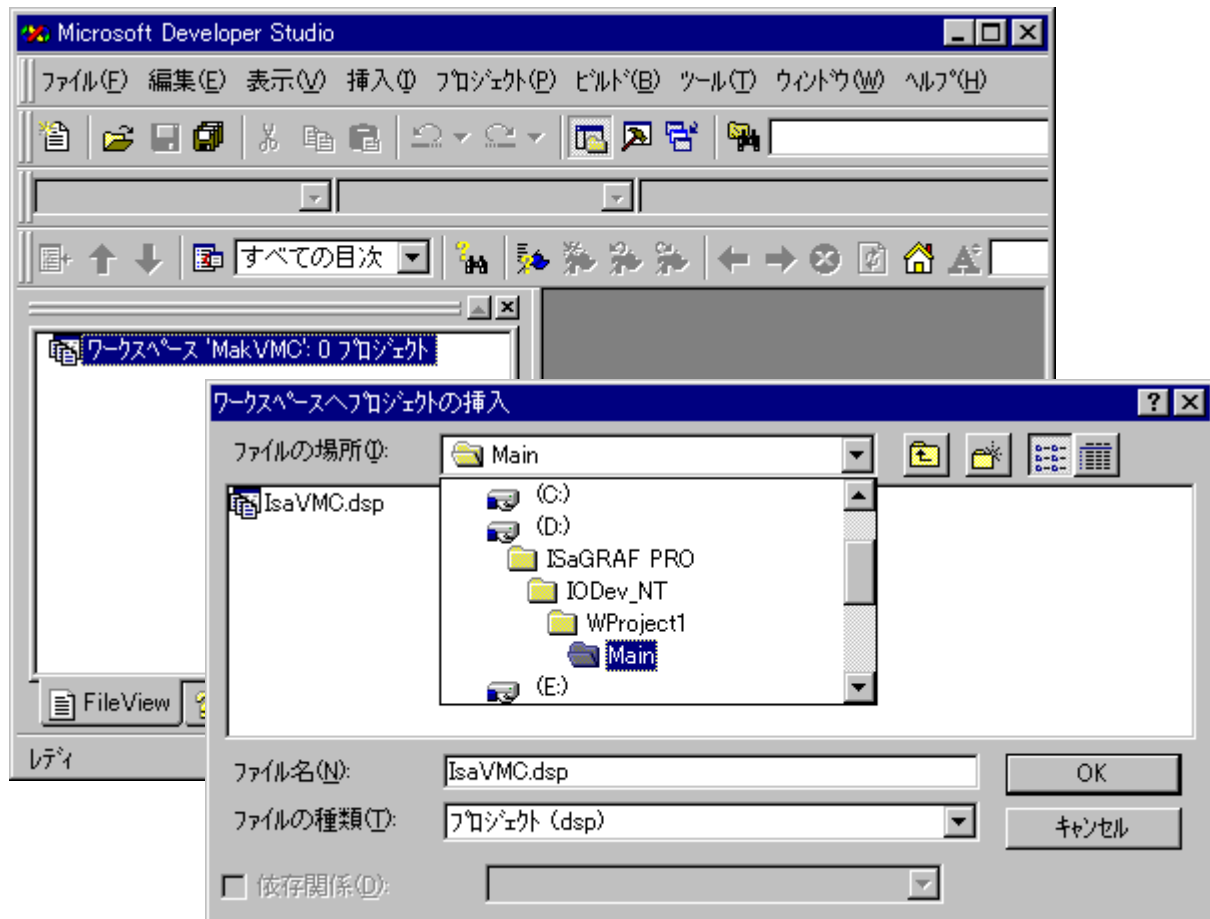
...Prj¥Project1¥Config¥コンフィグ名_NetworkConf.xcc を WProject1 にコピーします。



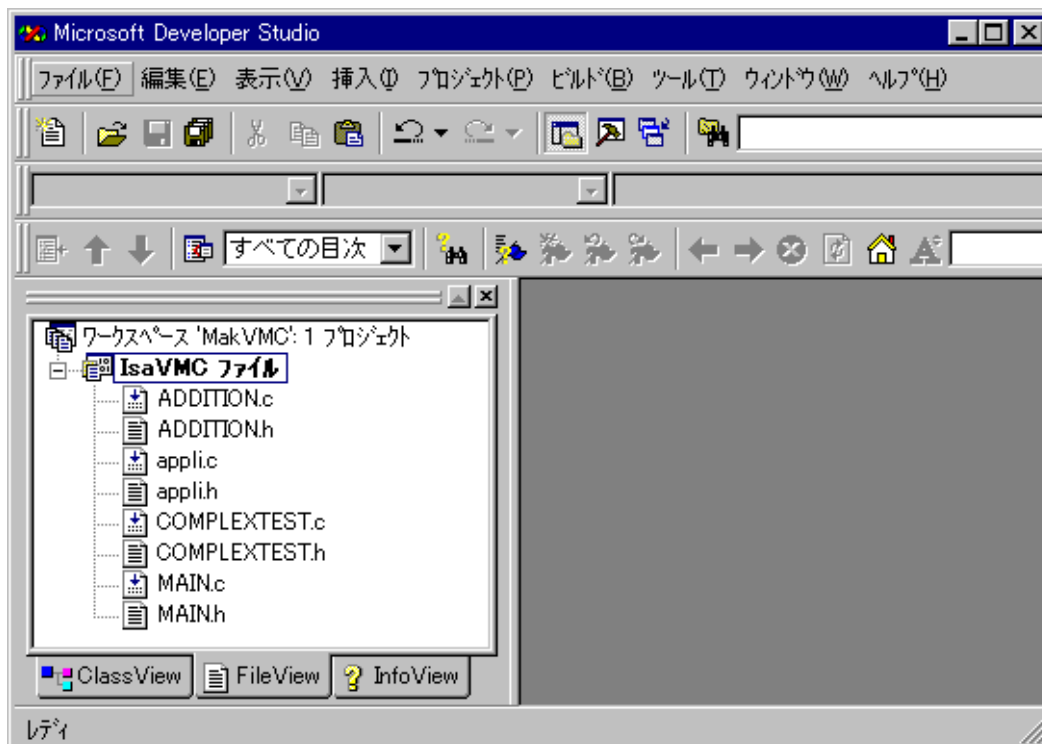
4.3.2. 実行ファイルのビルド

各リソース毎に実行ファイル IsaVMC.exe を作ります。

...¥IODevKit_NT¥MakVMC.dsw を立ちあげて、目的のリソースフォルダの IsaVMC.dsp を挿入します。

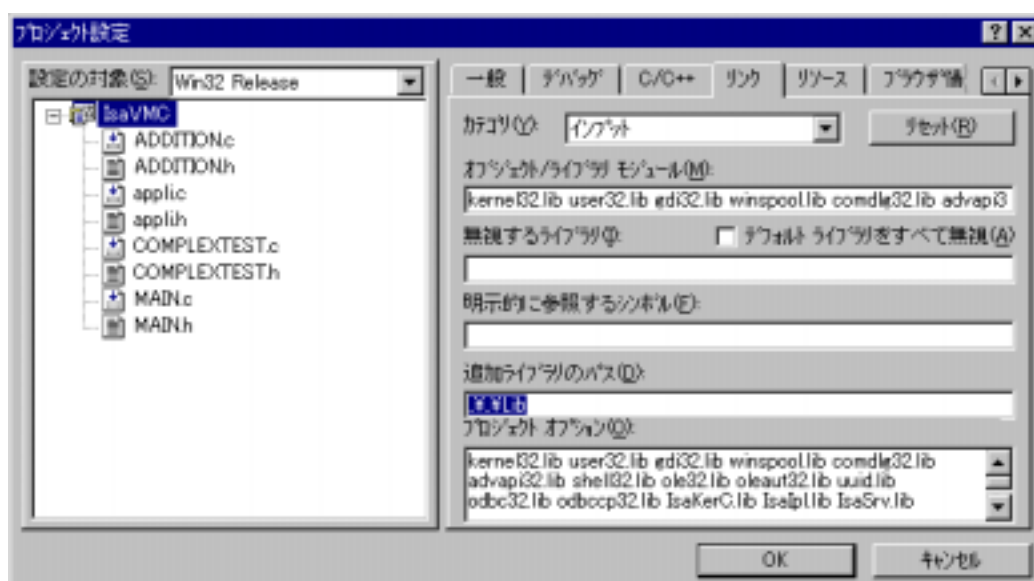


CソースファイルとCヘッダーファイルを挿入します。

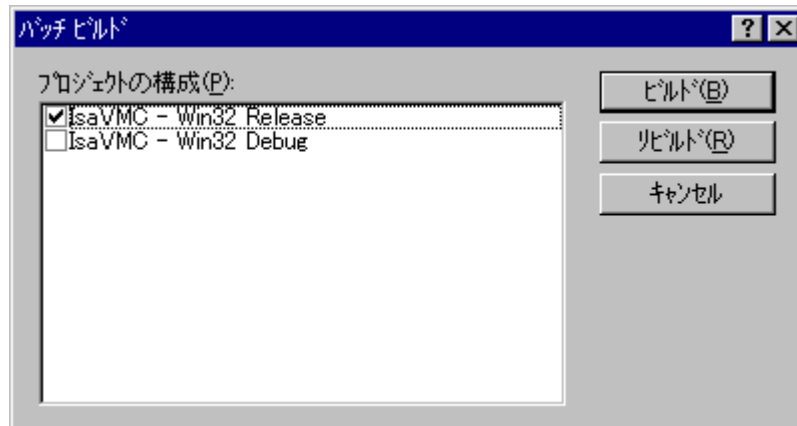


ビルドの環境をディレクトリ構造に合わせて変更します。ここでは Release モードを使用します。

変更するのは C/C++ / プリプロセッサ / インクルード ファイルのパス、リンク / インプット / 追加ライブラリのパス が考えられます。また、実行ファイル名は後で変更しますが、ここで直接設定してもかまいません。



リリースモードでビルドします。



するとワークホルダに IsaVMC.exe が生成されます。

これをリソース毎に行います。

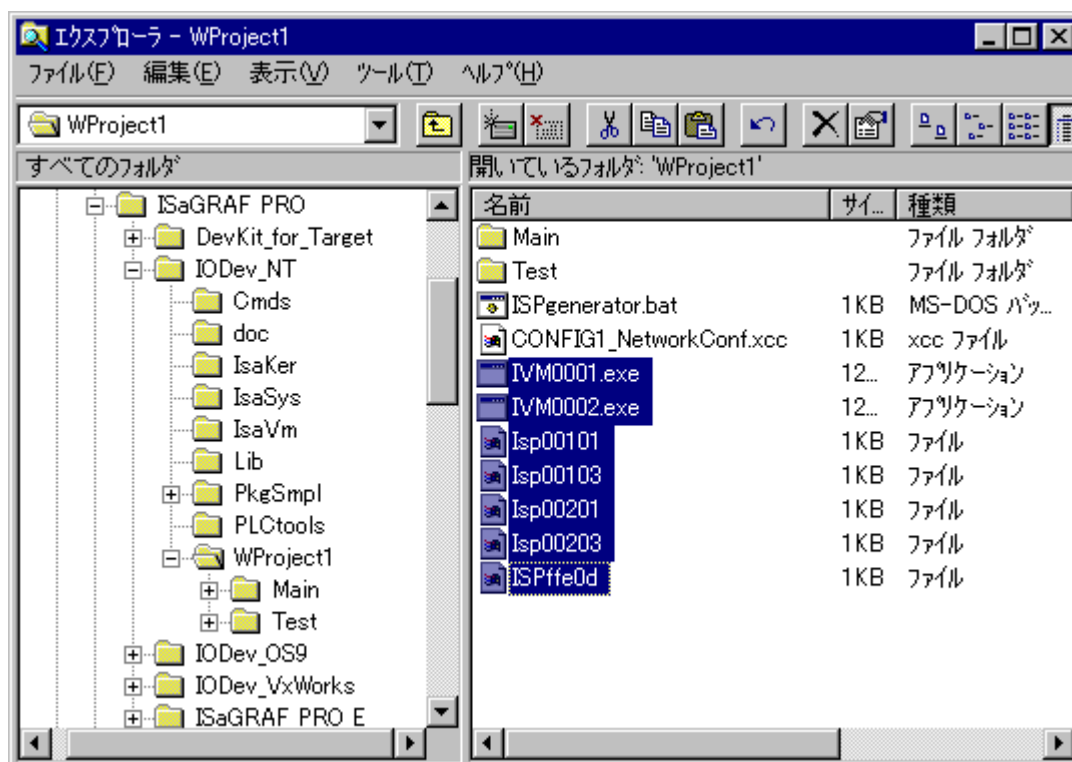
4.3.3. 実行ファイルとリソースファイルの準備

いままで生成されたファイルから、ISaGRAF PRO ターゲットに組み込むためのファイルを作ります。これらはリソース毎に必要です。

IVMxxxx.exe	各リソースの IsaVMC.exe を名前を変えて IVMxxxx.exe とします。 “xxxx”の部分はリソース番号が 16 進数で入ります。
ISPxxx01	リソースフォルダにコピーした、“リソース名_Pou_プログラム名.xcc” を連結して ISPxxx01 にします。 連結には DOS の copy コマンドの /B オプションが使えます。連結は appli.h に書いてある順序で行ってください。 “xxx”の部分はリソース番号が 16 進数で入ります。
ISPxxx03	リソースフォルダの“リソース名_Conf.xcc” を ISPxxx03 としてコピーします。 “xxx”の部分はリソース番号が 16 進数で入ります。

また、プロジェクトにつき 1 つ、次のファイルを用意します。

ISPffe0d	ワークフォルダの “コンフィグ名_NetworkConf.xcc” を ISPffe0d としてコピーします。
----------	---



これらの作業はバッチファイルを作って行うと便利です。

バッチファイルの例

```
@rem リソース 1 : MAIN の処理
copy %MAIN%\IsaVMC.exe %IVM0001.exe
copy /B %MAIN%\MAIN_Pou_ADDITION.xcc+%MAIN%\MAIN_Pou_MAIN.xcc+%MAIN%\MAIN_Pou_CO
MPLEXTTEST.xcc %ISP00101
copy %MAIN%\MAIN_Conf.xcc %ISP00103
```

```
@rem リソース 2 : TEST の処理
copy %TEST%\IsaVMC.exe %IVM0002.exe
copy /B %TEST%\TEST_Pou_SIMULATE.xcc+%TEST%\TEST_Pou_MULTIPLICATION.xcc %ISP00201
copy %TEST%\TEST_Conf.xcc %ISP00203
```

```
@rem コンフィグ : コンフィグ 1 の処理
copy %CONFIG1%\NetworkConf.xcc %ISPffe0d
```

4.3.4. ターゲットの実行

以上のファイルをターゲットフォルダ (ここでは Ccmds) にコピーし、ISaGRAF.exe を実行します。