TOPPERS 活用アイデア・アプリケーション開発コンテスト

部門　：　活用アイデア部門　（アプリケーション開発部門）

作品のタイトル　：　Toppers_ASP カーネルと Scilab による
組込みメカトロニクス制御シミュレーション

作成者　：　塩出　武（ シオデ　タケシ）

対象者　：　実機レス環境でモーター含むメカ制御プログラムの設計
および検証、学習をしてみたい方

使用する開発成果物　：
- Toppers_ASP カーネル_SkyEye シミュレーション環境
- Cygwin 環境(32bit 環境としました)
- Eclipse3.6_Helios(+ Zylin Embedded デバッガ)
- VisualC++2010Express 版
  (SkyEye 付属の devicemanager に含まれる COM オブジェクトを用いた割込み＋メモリ共有 DLL アプリ作成用)
- Scilab5.5.0(モーター制御用シミュレーション環境)

目的・狙い

・組込みメカ制御を始めてみたいが、機材、予算に制約がある。
・机上の制御方法を実機で動かす前に事前検証したい。
・評価機材が上がってくる前に先行で制御プログラムの開発を進めたい。
・組込みによるメカ制御を学習、体験したい。
といった要求に対応するために、オープンソース環境による、メカ制御の設計/検証環境を提案します。(2013 年度応募の JSP カーネル版での内容を ASP シミュレーション環境へ展開しました。)

目次

1　アイデアアプリケーションの概要

2013 年度の JSP カーネルと Scicos_lab によるメカトロニクスシミュレーション の内容をもとに、機能を付加し、ASP カーネルでのシミュレーション環境へ展開しています。

1.1　構成

ASP カーネル側のシミュレーションは Toppers カーネル向けシミュレーション環境 (SkyEye シミュレーション環境) を用いました。また、モーター側のシミュレーション環境も従来の Scicos_Lab から Scilab(バージョン 5.5.0)32bit 版 に変更しています。基本的な操作方法はほとんど変わりません。*1

上記構成にて、2 モーターの制御を行い、タスク/割込みの動作確認をします。

1.2　今回応募内容の主要な変更点

1.2.1　Toppers↔Scilab 間プロセス間通信の方法

■(前回)JSP カーネル

- 割込み → デバッグコンソールへ追加した割込み機能へ Scicos_Lab 側からメッセージ送信 (DLL アプリ)
- 共有メモリ → DLL アプリ内に作成した共有セクションを使用 (#pragma data_seg)

■(今回)ASP カーネル

- 割込み → SkyEye 付属の devicemanager より提供される COM オブジェクト機能を使用”RaiseInterrupt(ID)”
- 共有メモリ → SkyEye が提供する共有メモリ機能 (skyeye.conf にて設定) により設定し、devicemanager により提供される COM オブジェクト機能を用いて、共有メモリを読み/書き”CKernel− >Write(...)”、”Read(...)”。

1.2.2　デバッグ用統合環境の変更

VisualC++2010Express → Eclipse(Herios 版:日本語化環境) に変更 (Cygwin 上で実行のため)。*2

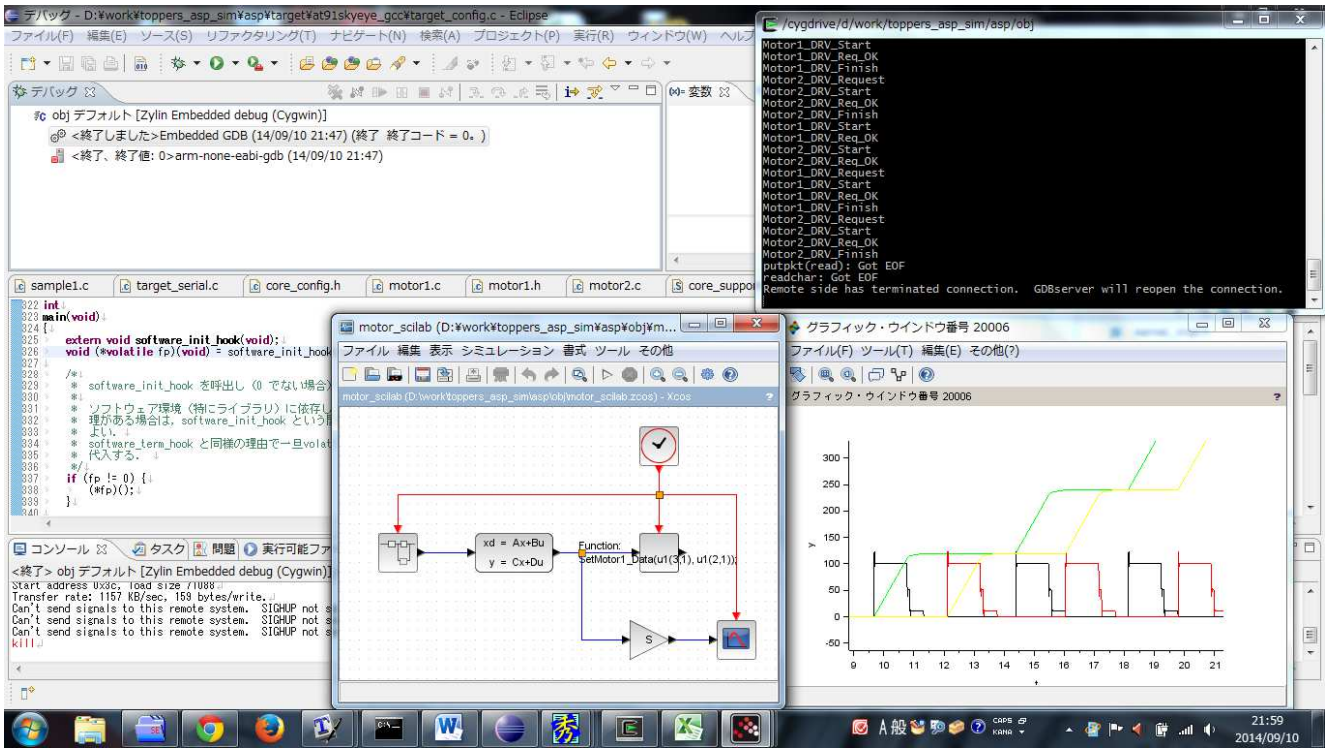

図 1　アプリケーションの概要

*1 現在は 5.5.1 にバージョンアップしています。
*2 最新の 4.4(Luna) 環境でも構いません。

2 アプリケーションの作成

2.1 ASP カーネルシミュレーション環境の立上げ

ASP カーネルシミュレーション環境の立上げは、(CQ 出版社様) インターフェース 2012 年 5 月号 の「TOPPERS カーネル用シミュレーション環境 TISE の使い方」を参照し行いました。一部追加作業がありましたので、追記します。

■devicemanager のインストールについて (regist.bat ファイルの変更)

```
REM デバイスマネージャの登録（exe と dll ファイルをフルパスで指定:"/"は"￥"に読み替えてください）
devicemanager /REGSERVER
regsvr32 devicemanagerps.dll  → win32PC は問題ありませんでした。
C://Windows/System32/regsvr32.exe
D://work/skyeyeskyeye_devm_package-1.0.5/skyeye_devm_package-1.0.5/bin/devicemanagerps.dll
```

■Cygwin のインストールについて

”make”パッケージは”devel”パッケージに含まれるようですのでこれをインストール。

その他は紙面の通りとなります。ASP カーネルが動けば OK ですので、同記事の”4 実行トレースログ以降は省略しています。なお、Eclipse 環境の設営については、こちら (おそらく松浦様) のサイトを参照しています。

2.2 ASP カーネル ↔ Scilab 間のプロセス間通信

2.2.1 デバイスマネージャー機能の適用

TOPPERS カーネル向けシミュレーション環境 に含まれる”devicemanager 機能 (COM オブジェクト)”を用いて、Scilab 側とのメモリの共有/タイマ割込みの機能を実装しました。SkyEye 環境を解凍すると devicemanager フォルダがあるので、”devicemanager.h”、”devicemanager_i.c”ファイルを使用しています。[*3]

2.2.2 VisualC++2010Express 環境での COM オブジェクトアプリケーションの作成

”devicemanager.h”、”devicemanager_i.c”から、デバイスマネージャー部の COM オブジェクトによるプロセス間通信機能にアクセスするアプリケーションを作成します。作成には VisualC++2010Express を使用しました。Express 版ですので COM オブジェクトそのものを作ることは出来ませんが、既存の COM オブジェクトを使用するアプリケーションは作成可能です。作成したアプリケーションは DLL ファイルとして、Scilab 側へ読み込みます。

表 1 に使用した COM 機能をまとめます。プロセス間通信自体は実現出来たのですが、私自身が COM オブジェクトのプログラミングに疎いため、CDevice 機能と CKernel 機能が混在してしまいました。[*4]COM プログラミングについては、

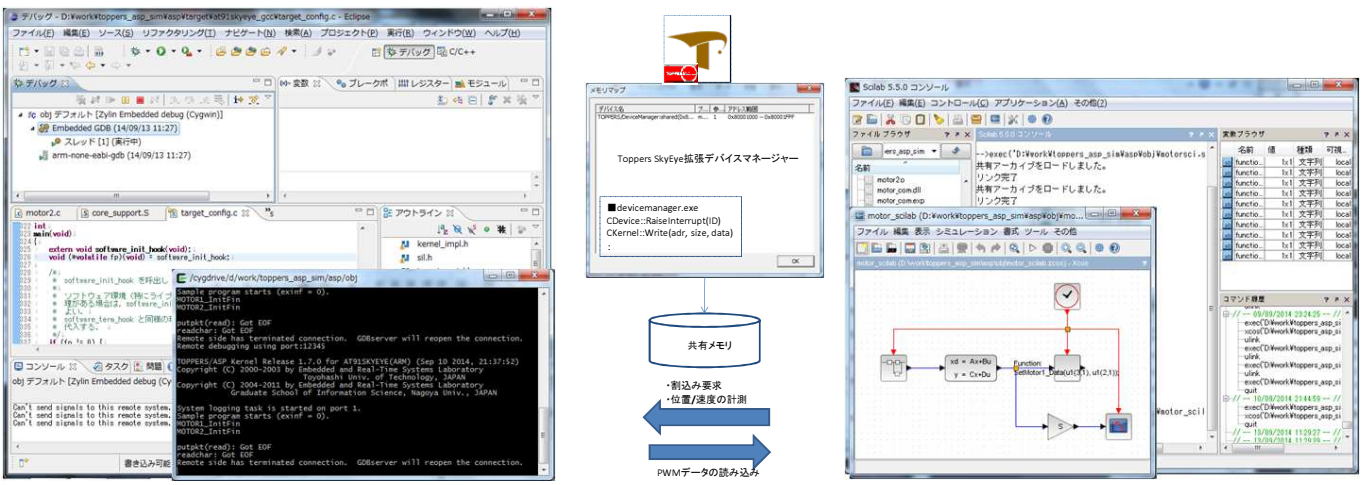

図 2 SkyEye 拡張デバイスマネージャーによるプロセス間通信

[*3] devicemanager は COM オブジェクトの定義、devicemanager_i.c は COM オブジェクトのクラス ID を定義

[*4] Scilab 側はデバイス側なので、基本 CDevice でまとめようとしましたが断念しました。

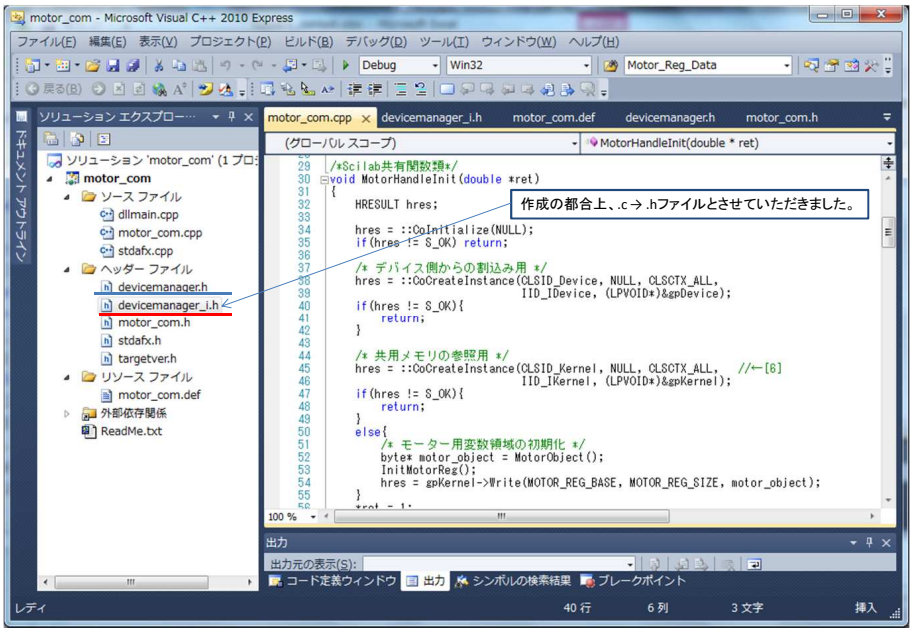

図 3　VisualC++2010Express による COM オブジェクトによるプロセス間通信アプリの作成 (DLL ファイル)

プロフェッショナル VisualC++2010(日経 BP 社様) の第 2 章「COM プログラミング」を参考にしました。

表 1　使用した COM オブジェクト類まとめ

機能	使用クラス	関数	備考
共有メモリ読み込み	Ckernel	Read(...)	CDevice → OnRead のアクセスが分からず
共有メモリ書き込み	Ckernel	Write(...)	CDevice → OnWrite のアクセスが分からず
割込み要求発行	CDevice	RaiseInterruput(ID)	ID は 0 番を指定 (sample.cfg では 16 番を設定)

なお、共有メモリの生成は SkyEye.exe 起動時に読み込まれる skyeye.conf で設定しました。

■SkyEye.conf の設定

途中省略...

```
mem_bank: map=M, type=RW, addr=0x00000000, size=0x08000000
mem_bank: map=S, type=RW, addr=0x80001000, size=0x00001000#, devm_addr=0x80001000～0x80001FFF
mem_bank: map=I, type=RW, addr=0xf0000000, size=0x10000000
```

以下省略...

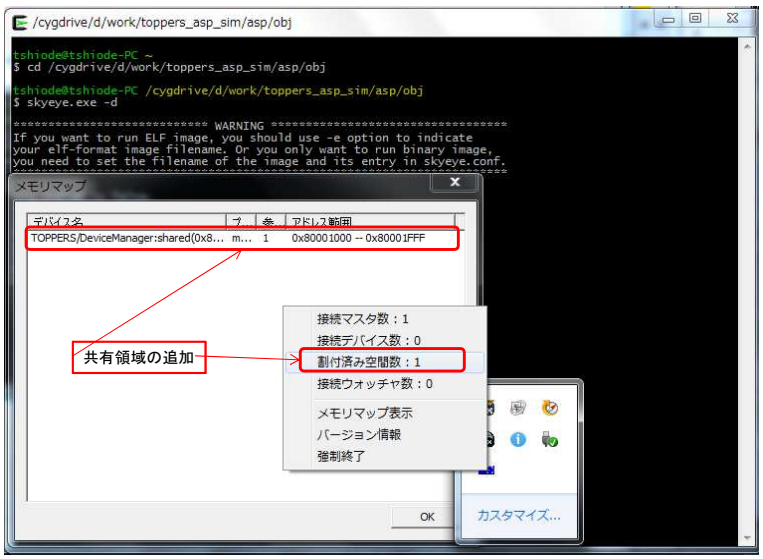

図 4　SkyEye.conf で共有メモリを設定

共有メモリは表 2 のように区画しました。評価用 (未記載) のデータも含めて、計 44 バイトを使用しています。[*5]

表 2 共有メモリの内訳 (各データサイズはいずれも 4byte です)

アドレス名	機能	ベースアドレスからのオフセット	備考
MOTOR_INT_ENABLE	割込み発行許可	0byte	割込みデバッグ用
MOTOR1_PWMDUTY	モータ 1 用 PWMDuty	4byte	
MOTOR1_PWMPERIOD	モータ 1 用 PWM 周期	8byte	未使用 (評価用)
MOTOR2_PWMDUTY	モータ 2 用 PWMDuty	12byte	
MOTOR2_PWMPERIOD	モータ 2 用 PWM 周期	16byte	未使用 (評価用)
MOTOR1_SPEED	モータ 1 用速度	20byte	
MOTOR1_POSITION	モータ 1 用位置	24byte	
MOTOR2_SPEED	モータ 2 用速度	28byte	
MOTOR2_POSITION	モータ 2 用位置	32byte	

2.3 Scilab 側 DC モーターモデルの作成

DC モーターモデルは昨年度と同等です。前回、2 モータにしては、モデル規模が大きかったこともあり、今回は各モータの要素を行列パラメータにしてスッキリさせています。[*6]以下モデルを掲載します。パラメータ行列 ABCD の内訳は以下のようにしています。左上がモータ 1 用で、右下がモーター 2 用です。簡単のためどちらも同じパラメータとしています。

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} + B\boldsymbol{u}$$
$$\boldsymbol{y} = C\boldsymbol{x} + D\boldsymbol{u}$$

$$\begin{bmatrix} \dot{i_1} \\ \dot{\omega_1} \\ \dot{\theta_1} \\ \dot{i_2} \\ \dot{\omega_2} \\ \dot{\theta_2} \end{bmatrix} = \begin{bmatrix} -R/L & -Ke/L & 0 & 0 & 0 & 0 \\ Kt/J & -D/J & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -R/L & -Ke/L & 0 \\ 0 & 0 & 0 & Kt/J & -D/J & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} i_1 \\ \omega_1 \\ \theta_1 \\ i_2 \\ \omega_2 \\ \theta_2 \end{bmatrix} + \begin{bmatrix} E/L & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & E/L \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u1_{pwm1} \\ u2_{pwm2} \end{bmatrix}$$

$$\begin{bmatrix} i_1 \\ \omega_1 \\ \theta_1 \\ i_2 \\ \omega_2 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i_1 \\ \omega_1 \\ \theta_1 \\ i_2 \\ \omega_2 \\ \theta_2 \end{bmatrix} \quad (D \text{ 項は } 0 \text{ のため省略しています})$$

R : 巻線抵抗、L : コイルインダクタンス、E : 印可電圧 : Ke モータ逆起電力定数、J : モータ慣性モーメント、D モータ粘性抵抗 : 、Kt モータ電流トルク係数 : 、i : 電流、ω : モータ角速度、θ : モータ角度

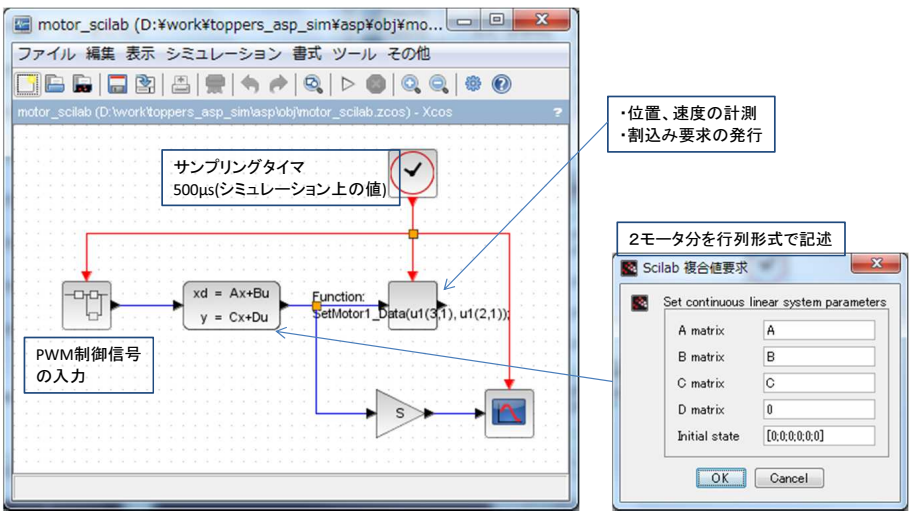

図 5 Scilab 側の DC モータシミュレーションモデル

[*5] 簡単のため SkyEye.conf では、共有領域を 4096byte(0x1000) としています。
[*6] モデルを整理することでシミュレーション速度 UP を期待しましたが、特に変わりませんでした...

2.4 サンプルプログラムの作成

2.4.1 ファイル構成 (JSP カーネル版と同等です)

1. Sample1.cfg：モーター用タスク等 OS リソース追加
2. Sample1.c/h：評価用プログラム追加
3. Motor1.c/h：モーター 1 制御タスク、割込み、HAL 層の実体 / 制御パラメータ定義、公開関数定義
4. Motor2.c/h：モーター 2 制御タスク、割込み、HAL 層の実体 / 制御パラメータ定義、公開関数定義
5. Rtos_api.h：OS サービスコールのラッパ関数定義

2.4.2 制御仕様

モータの制御仕様を図 6 に示します。今回のサンプルプログラムでは減速制御開始距離を短くしています。

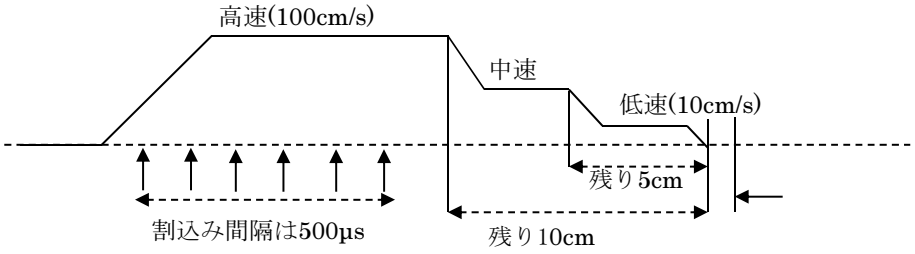

図 6 評価用モータ制御仕様

表 3 評価用キーボタンコマンドの内訳

ボタン名	動作内容	備考
”i”	両モータの初期化	最初に必ず実行願います。
”a”	モータ 1 駆動 (停止待ち有)	
”b”	モータ 1 駆動 (停止待ち無)	
”c”	モータ 1 駆動 (モータ 2 の減速開始を待つ)	
”d”	モータ 2 駆動 (停止待ち有)	
”e”	モータ 2 駆動 (停止待ち無)	
”f”	モータ 2 駆動 (モータ 1 の減速開始を待つ)	

3 アプリケーションの動作確認

3.1 サンプルアプリケーションの起動

3.1.1 TOPPERS 側

2.1 の作業で準備した eclipse 上の ASP 環境を動かします。まず、Cygwin 上で SkyEye を起動します。

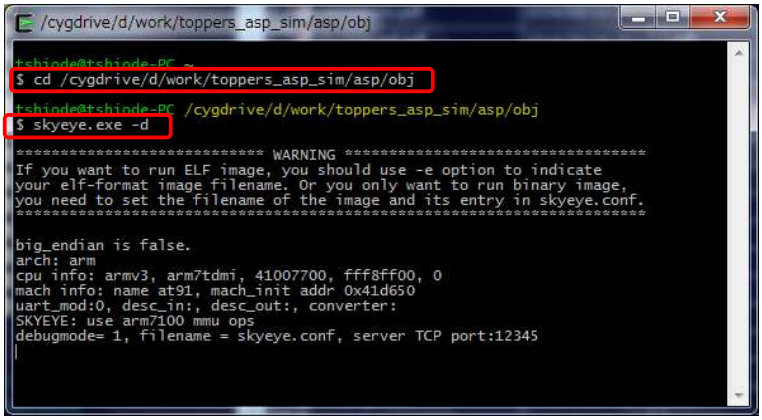

図 7 SkyEye の起動

この時、SkyEye の起動に合わせて、devicemanager も起動すると思いますが、起動しない場合は、devicemanager.exe を手動で立上げてください。*7次に、Eclipse 環境を起動します。デバッグ構成の設定が完了していると、ショートカットから起動できます。デバッグ環境が起動すると、動作前に一旦ブレークで止まりますので、デバッグの再開ボタンを何度か押して実行してください。私の環境では 3 回でした。

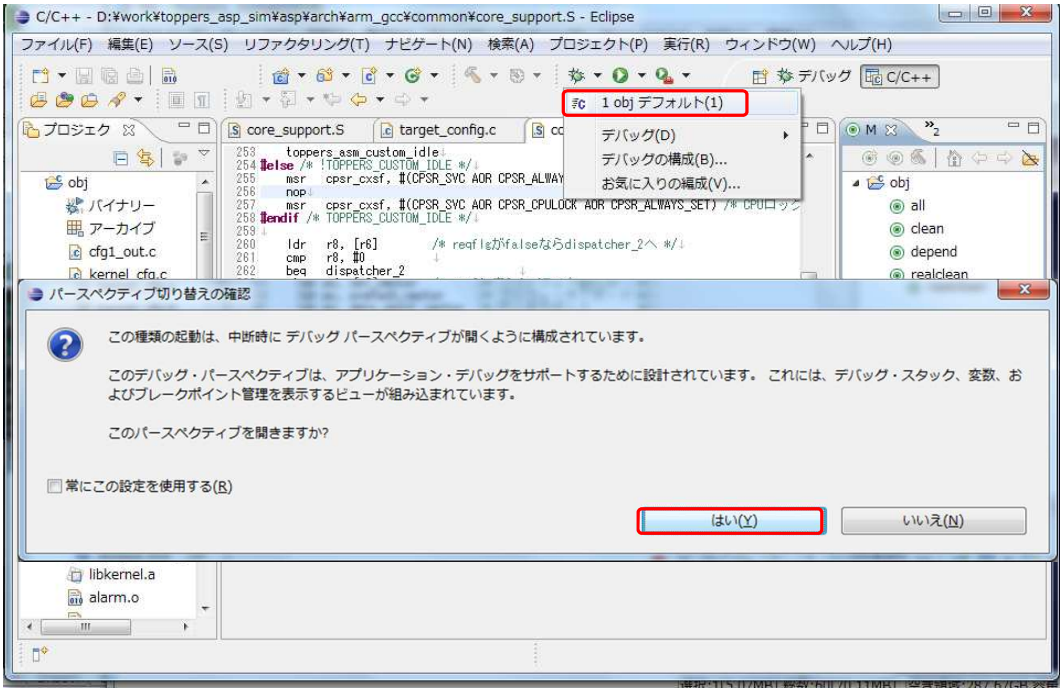

図 8 デバッグ環境の起動

3.1.2 Scilab 側

次に Scilab 側を起動します。まず asp/obj フォルダに移動します。”motorsci.sci”ファイルを実行すると、デバイスマネージャー部の COM プロセス間通信アプリケーション (DLL ファイル) を Scilab 側へダウンロードします.

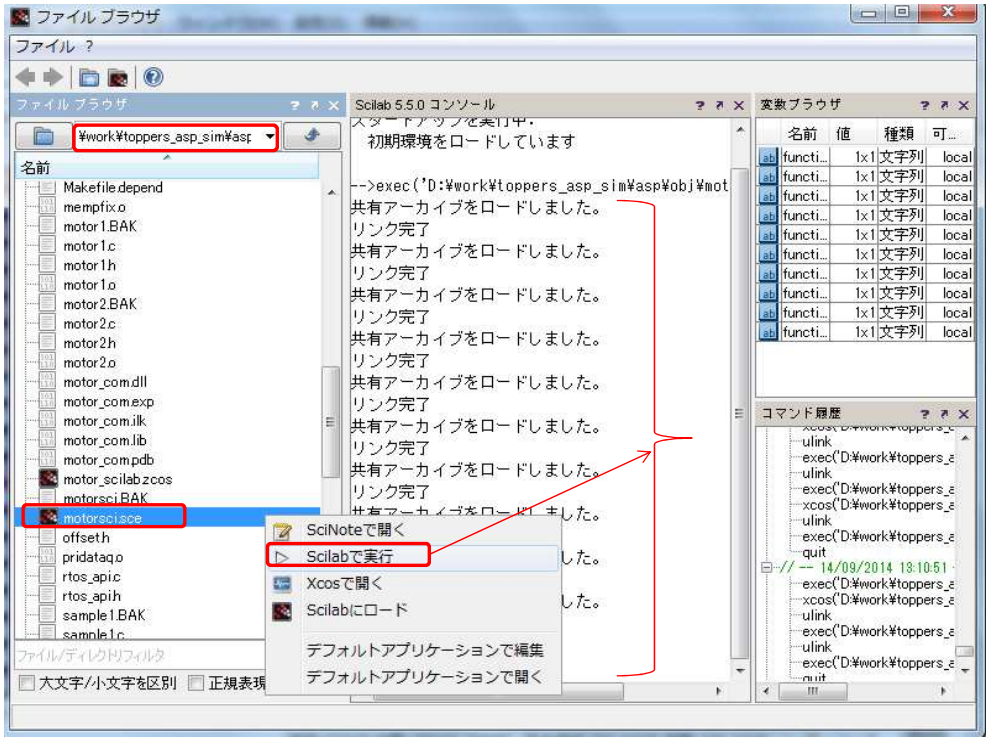

図 9 プロセス間通信用の DLL ファイルの読み込み

*7 私の環境では途中から自動で起動しなくなりました。

motor_scilab.zcos を立ち上げます。シミュレーション開始ボタン (再生ボタン) を押して実行してください。

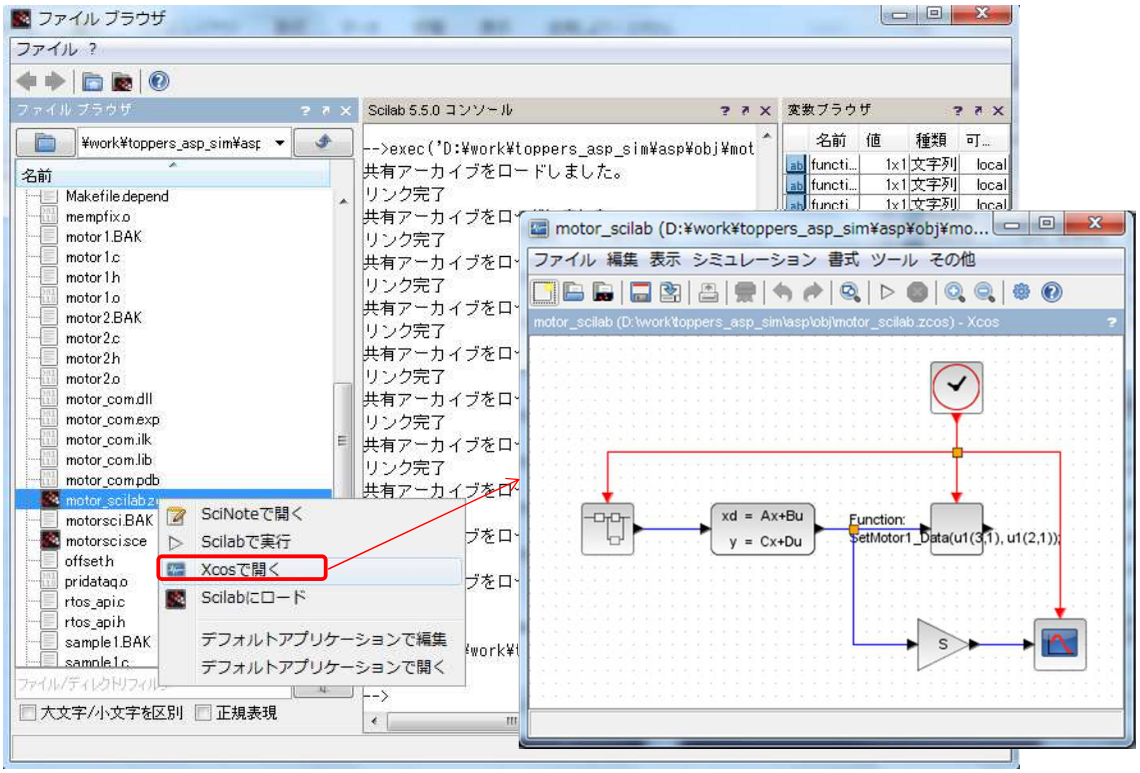

図 10　モータ用シミュレーションモデルの立上げ

3.2　シミュレーションによる動作確認

3.2.1　モーターの駆動確認

両者が起動した状態で、Cygwin 上で表 3 のキーボタンを押すと、動作を確認出来ます。

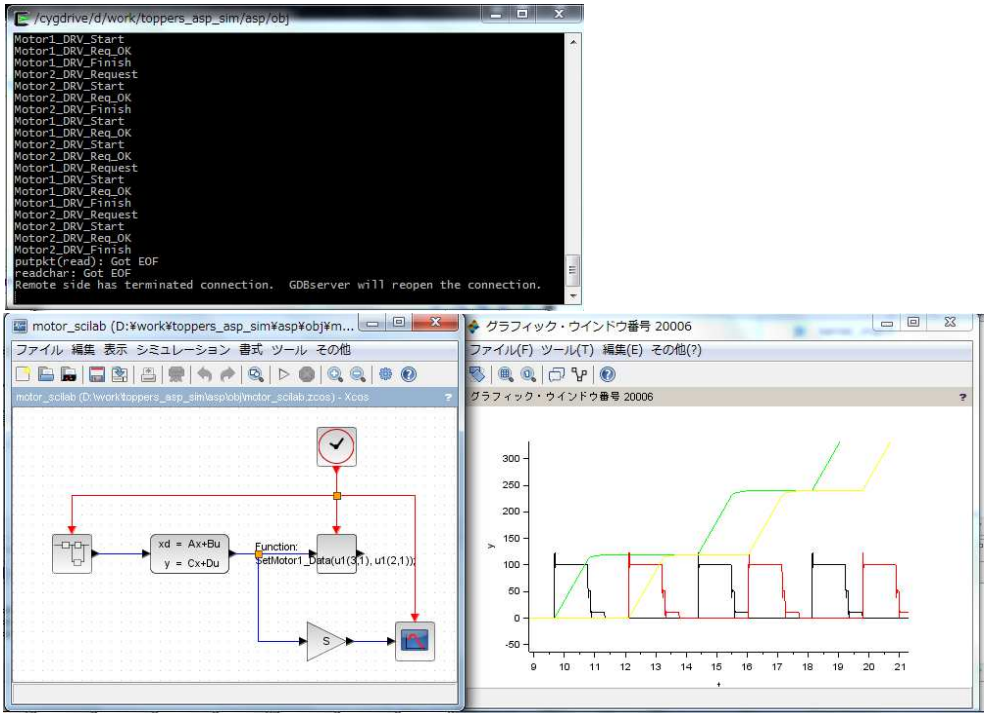

図 11　シミュレーションによるモータ制御動作確認 (最初に”i”ボタンで初期化をお願いします)

3.2.2 モータの待ち合わせ駆動

イベントフラグの動作確認のため、相手側モータが減速に移行したのを待って駆動開始する動作を追加しています。

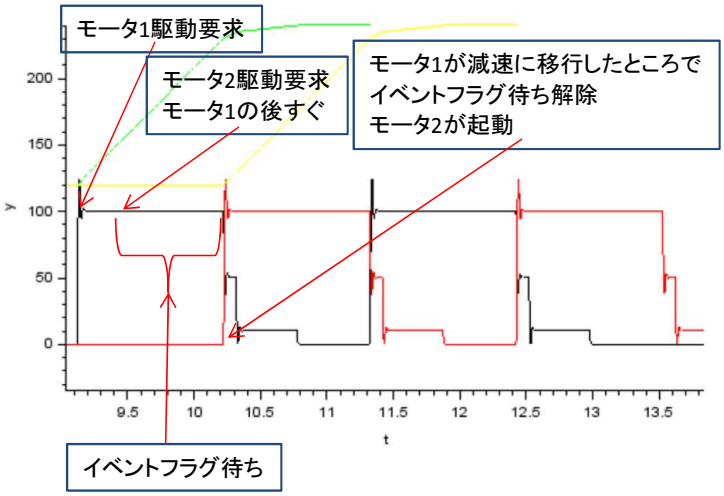

図 12 モータ 1、2 の待ち合わせ駆動 (黒:モータ 1 / 赤：モータ 2 です)

3.3 (評価用) 割込みでのブレークについて

割込み制御の評価用に、共有メモリ上にフラグを設けて簡易的なブレーク機能を持たせています。表 2 の MOTOR_INT_ENABLE を 0/1 フラグとして動作させ、以下のように制御しています。

1. Scicos 側の割込み発行時に 1 設定
2. Scicos 側は同フラグが 0 になるのを待つ (while 待ち)
3. TOPPERS 側のモータ割込終了時に 0 設定
4. Scicos 側でフラグが 0 になったことを確認して待ち解除

こうすることで、TOPPERS 側に連動して、Scicos 側をブレーク (停止) させることが出来ます。ただし以下の制限があるためデフォルトは無効にしています。

■制限事項

TOPPERS 側を中断させた状態で Scilab を強制終了すると、while 待ちから抜けてこないため終了できない場合があります。そのため同機能を有効時は TOPPERS 側を再開させてから、Scilab を終了してください。

■設定切り替え部 (VisualC++ プロジェクト側の motor_com.h ファイル)

```
/* 評価用 シミュレーションの簡易ブレーク機能（デフォルト無効）*/
#define DEBUG_SCICOS (0) /* 0:無効 1:有効 */
/* COM オブジェクト用 アドレス定義（SkyEye の定義と合わせる）*/
#define MOTOR_REG_BASE (0x80001000)
以下省略…
```

4 まとめ

TOPPERS_ASP カーネルシミュレーションと、Scilab による DC モータシミュレーションを連動させて、組込み制御のシミュレーション評価環境を作成しました。プロセス間通信は、SkyEye 付属のデバイスマネージャー機能を用いています。作成時に課題はありましたが、なんとか動く形に持って行けましたので、こちらの内容にて応募致します。[*8]

時間的な制約もあり、サンプルプログラムの内容まで説明が及びませんでした。申し訳ございません。

以上となります。よろしくお願致します。

[*8] デバイスマネージャの使い方について、追加の情報がございましたら、ホームページ等でご提供お願い致します。