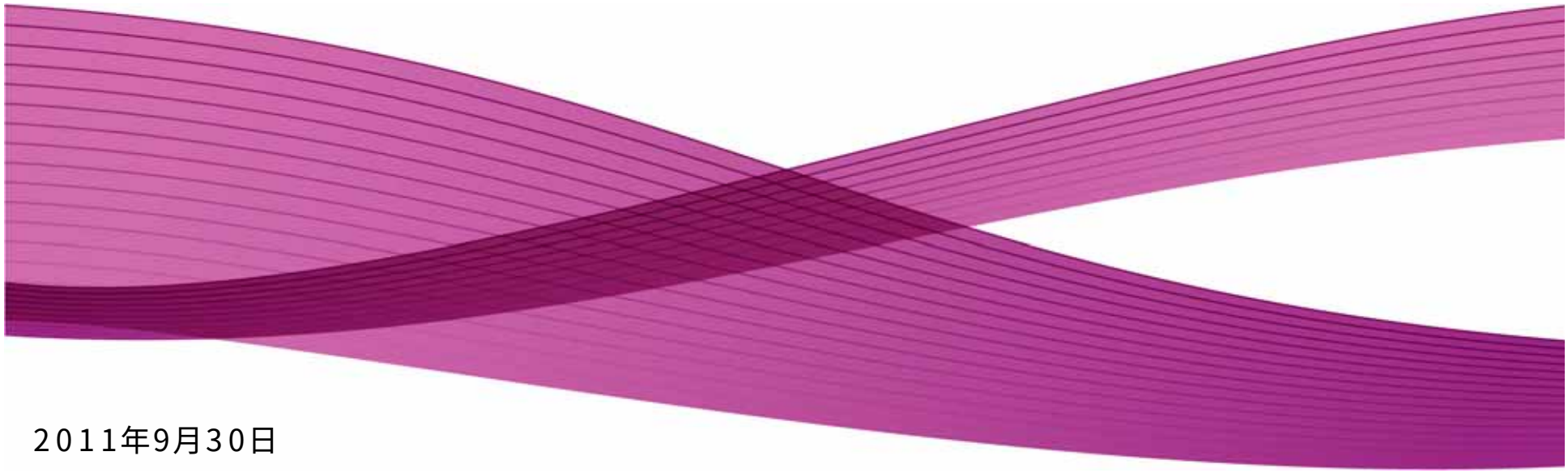
サブシステム部にESLを適応した事例

Carbon Design Systems社のSoC Designerを利用して



2011年9月30日

富士ゼロックス株式会社
コントローラ開発本部コントローラプラットフォーム第五開発部　三角 晃、橋本貴之

FUJI Xerox

この資料で使用するシステム名、製品名等は一般にメーカーや
団体の登録商標などになっているものもあります

なお、この資料の中では、トレードマーク、コピーライト等の表示は
明記しておりません

富士ゼロックス(株)の宣伝

高速インクジェットプリンター

2800 Inkjet Color Continuous Feed Printing System

・フルカラーバリアブル印刷

・プリントスピード フルカラーで200m/分

（A4サイズ：2,624ページ）

FUJI Xerox

本日の内容

ESLを使ったシステム性能の検討では, 主にメインCPU/メモリ/画像処理などのメインシステムでの利用が多いものです。

今回の発表ではメインシステムではなく, サブシステム(メカコン)部にESLを適応した事例について発表致します。

具体的には、メカコン用のMCUが処理するソフトウェアと外部デバイス(I/O)との関係をCarbon社のSoC Designer*を利用して確認致しました。

* SoC DesignerはCarbon Design Systems社の登録商標です。

目次

1)、一般的なESLの使い方

2)、適用したシステムの全体像と概要

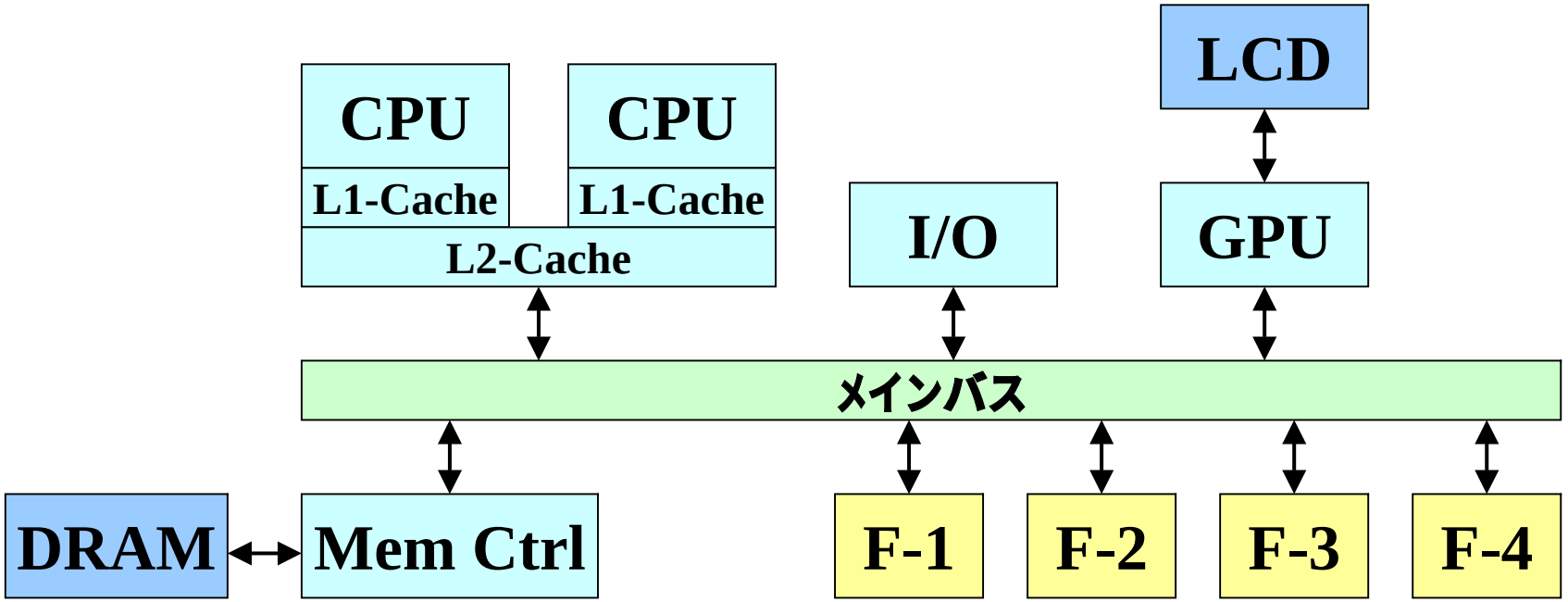
3)、メカコン部のモデル化

4)、まとめ

FUJI xerox

1)、一般的なESLの使い方

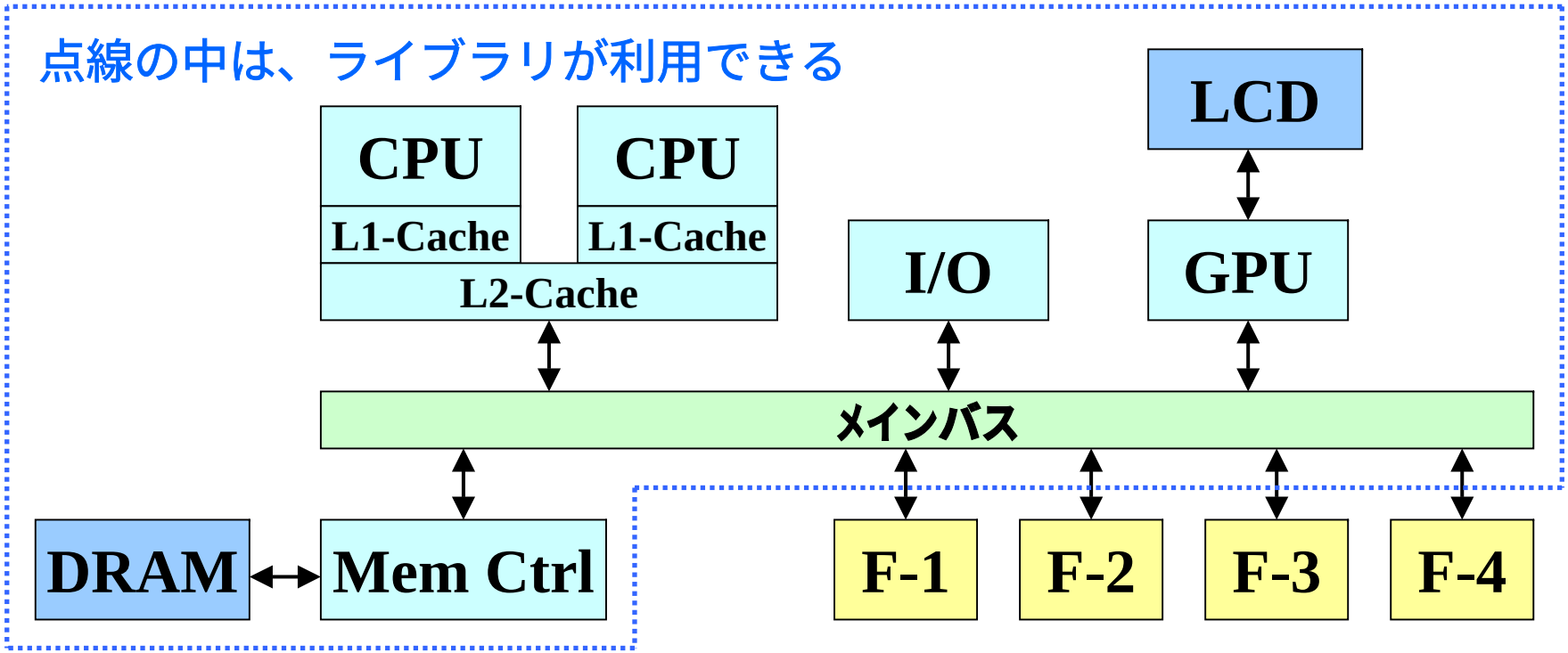
メインシステムのアーキテクチャや性能検証などに使う



- CPUの周波数、L1/L2サイズ
- 内部バスの幅、周波数、構成
- メモリコントローラのポート数
- 機能モジュール(F-n)の位置、周波数
  などなど

FUJI xerox

Carbon Design Systems社のSoC Designerを使うと



- CPUの周波数、L1/L2サイズ
- 内部バスの幅、周波数、構成
- メモリコントローラのポート数

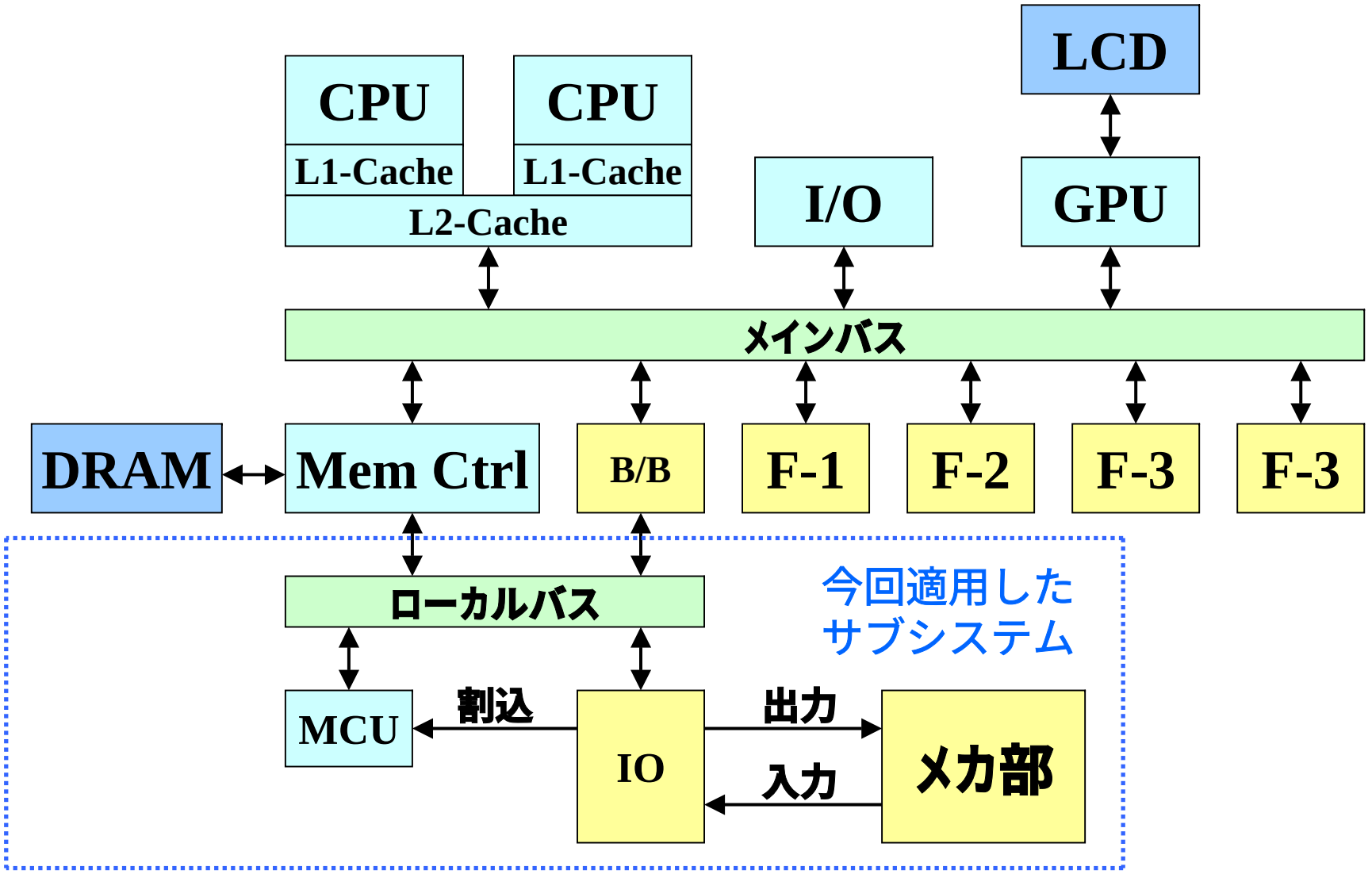
ライブラリを
利用すればいい

- 機能モジュール(F-n)の位置、周波数
  などなど

FUJI Xerox

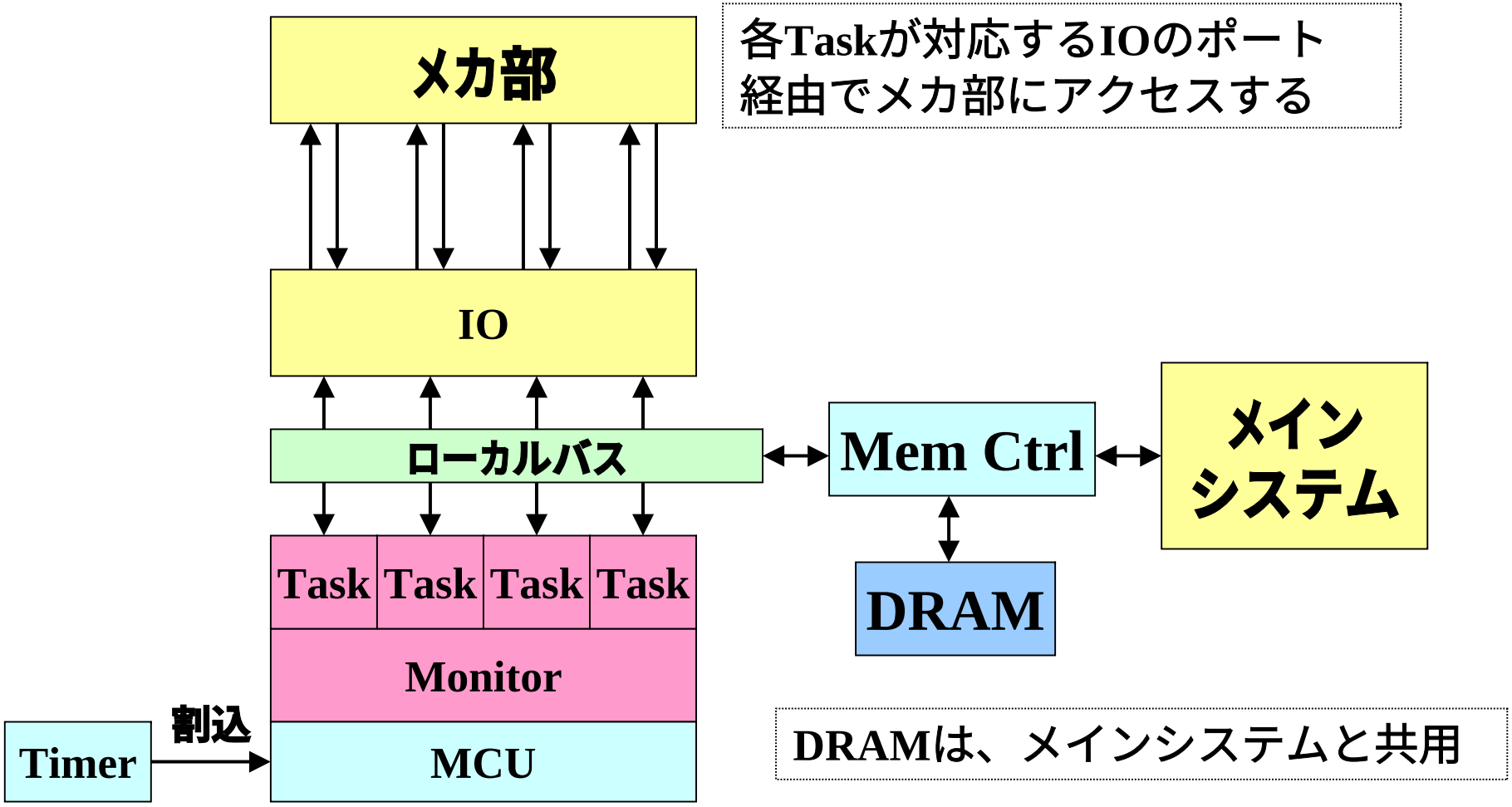
2)、適用したシステムの全体像と概要

今回の事例：サブシステム(メカコン部)



FUJI Xerox

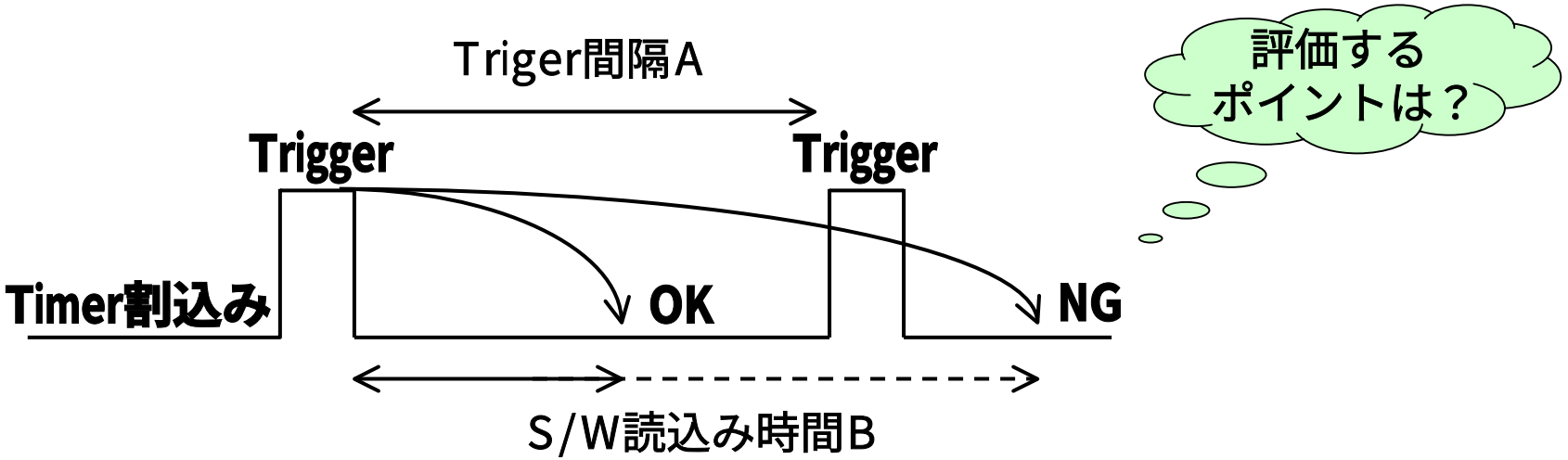
メカコン部のソフトウェア



FUJI Xerox

ESLで何を評価するか？

システムのスケーラビリティーの評価



Triger間隔（プリンタの仕様に依存）

S/W読込み時間（使用可能バス帯域に依存）

A≧B　OK

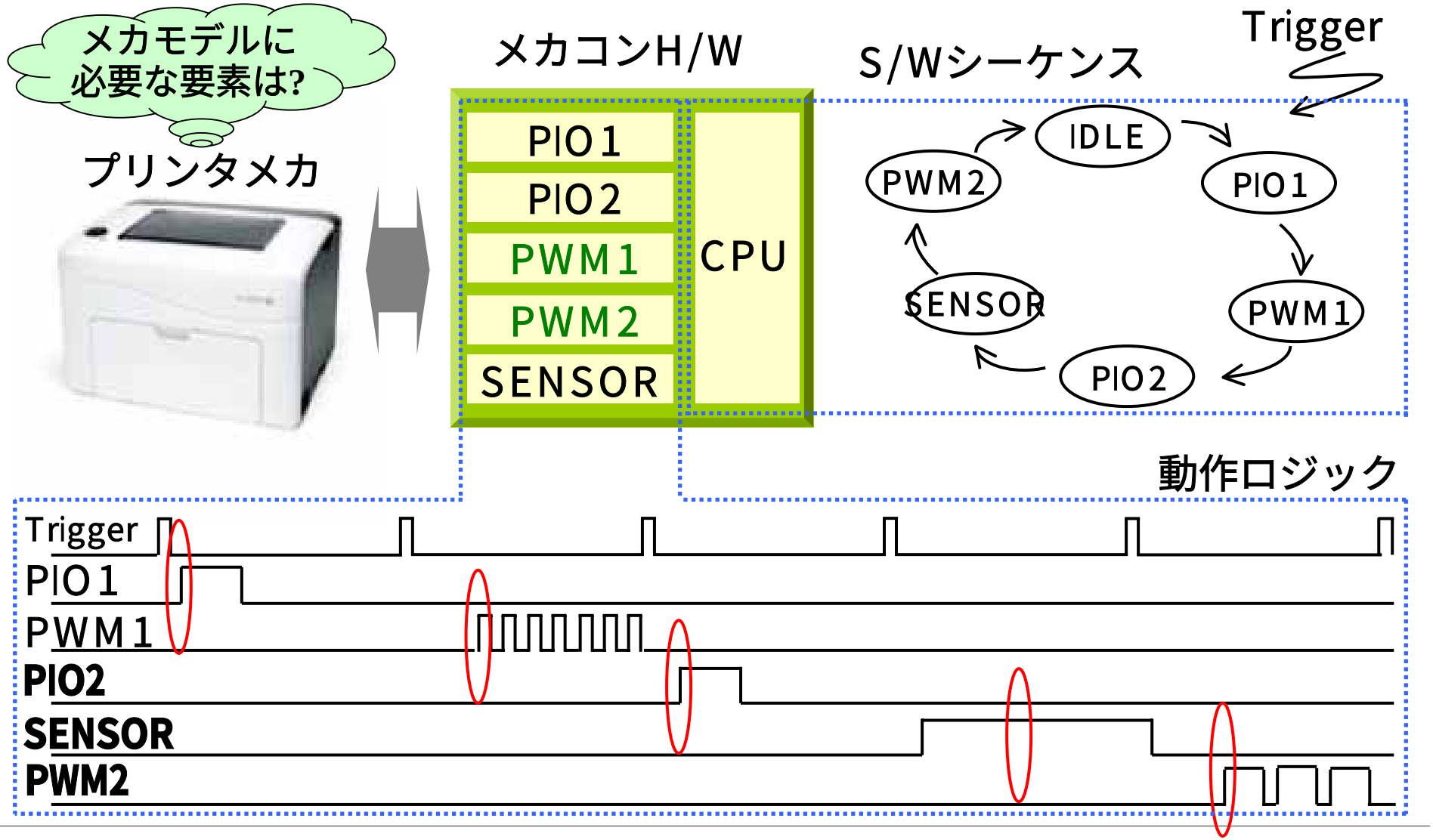
A＜B　NG

ESLパラメータ　：プリントスピード、使用可能バス帯域

FUJI Xerox

プリンタメカモデルの作成

目的はメカコンがリアルタイム処理できるかを確認すること



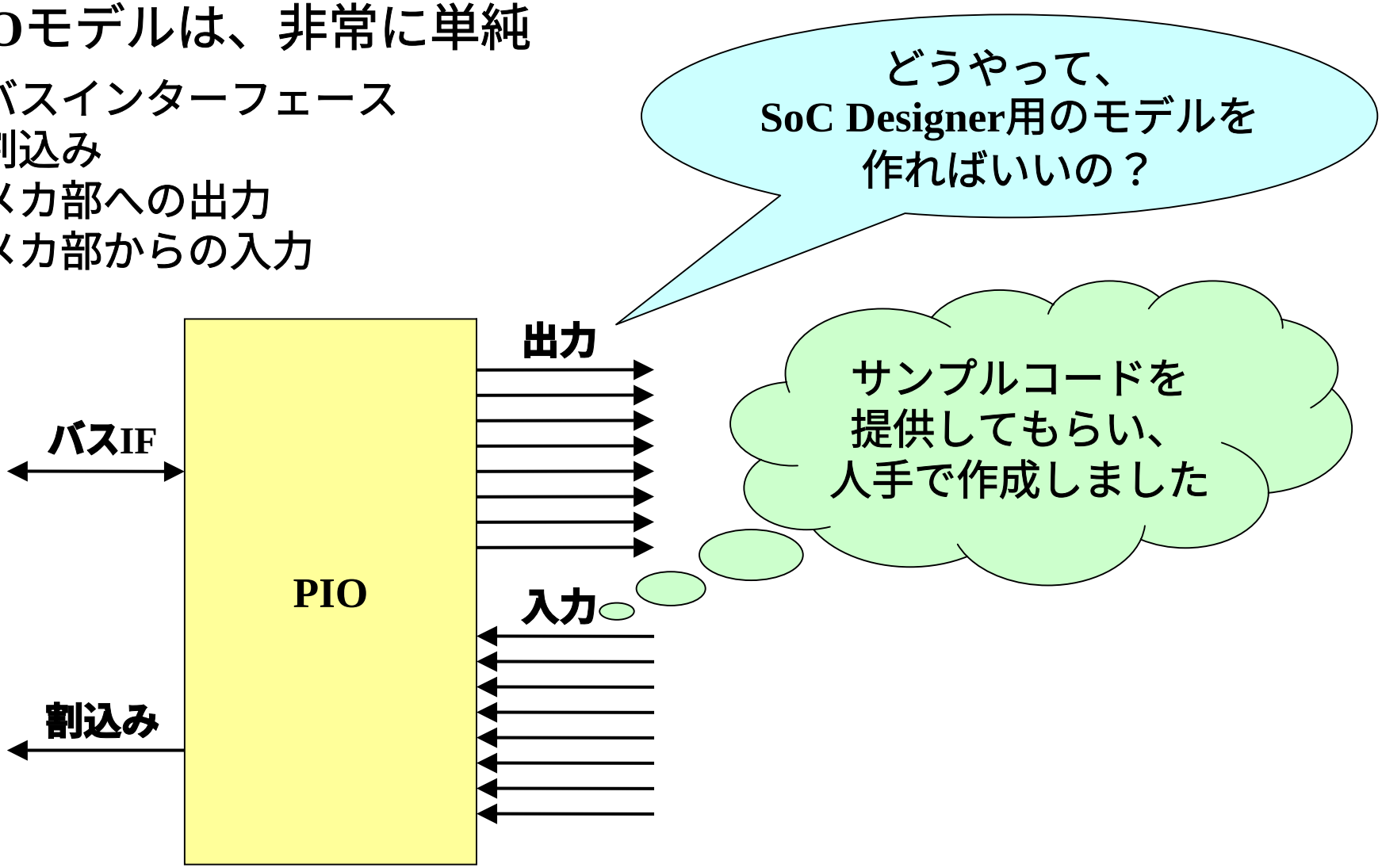
FUJI Xerox

3)、メカコン部のモデル化

PIOモデル

PIOモデルは、非常に単純

バスインターフェース
割込み
メカ部への出力
メカ部からの入力



FUJI xerox

PIOモデルに必要なもの

PIOモデルを作成するには、こんなにファイル(9個)が必要！

PIOモデル
- **My_PIO.cpp**
- **My_PIO.h**
- **My_PIO.def**
- **My_PIO_CADI.h**
- **My_PIO_CADI.cpp**

バスインターフェース
- **My_BusPort.h**
- **My_BusPort.cpp**

メカ部からの入力
- **My_inPort.h**
- **My_inPort.cpp**

割込みとメカ部への出力は、
SystemCのsc_portが利用できるので、
ファイルは必要なかった。

各ファイルの内容を
知りたいのなら？

マニュアル : Carbon ESL API

を読みましょう！

FUJI xerox

Carbon ESL API

とはいっても、Carbon ESL APIでは、

CASI : Cycle Accurate Simulation Interface

CADI : Cycle Accurate Debug Interface

CAPI : Cycle Accurate Profile Interface

の機能があります。

今回のPIOモデルでは、

CASI と CADI

を実装しました。

FUJI xerox

テンプレートから生成できる

SoC Designer CanvasのToolsメニューの

Component Wizard …

にて、モデルのテンプレートが生成できる

各ステップに従って、名前、ポートの追加を行うだけで、
9個のファイルの他に、

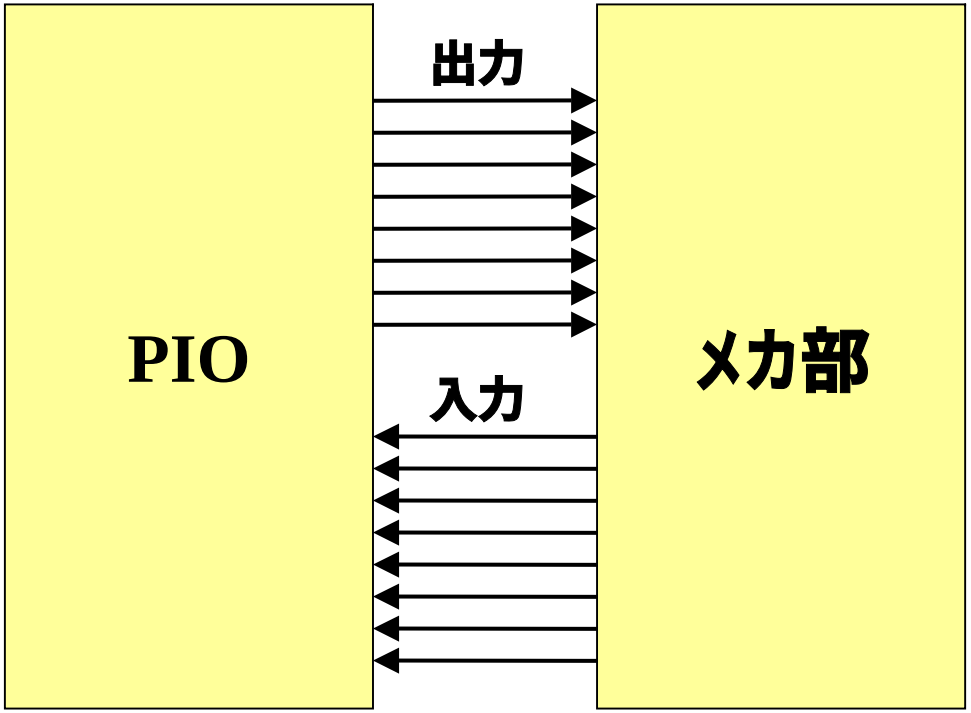
Makefile	**Linux用Makefile**
My_PIO.mak	**NMAKE用Makefile**
My_PIO.vcproj	**VC++ 2005用プロジェクトファイル**

人手で書くのは、止めましょう！
間違えるし、時間の無駄です！

メカ部モデル

メカ部は、PIOからの入力、PIOへの出力であるが、詳細まではよくわからない。

→ モデル化できない？



Carbon MxScriptを使えば？

Carbon MxScriptとは？

- C言語ライクな言語
- スクリプトファイルを書くだけでOK!
- コンパイル不要

- シミュレーションのバッチモードで利用可能
- MxStubなどのコンポーネントで利用可能

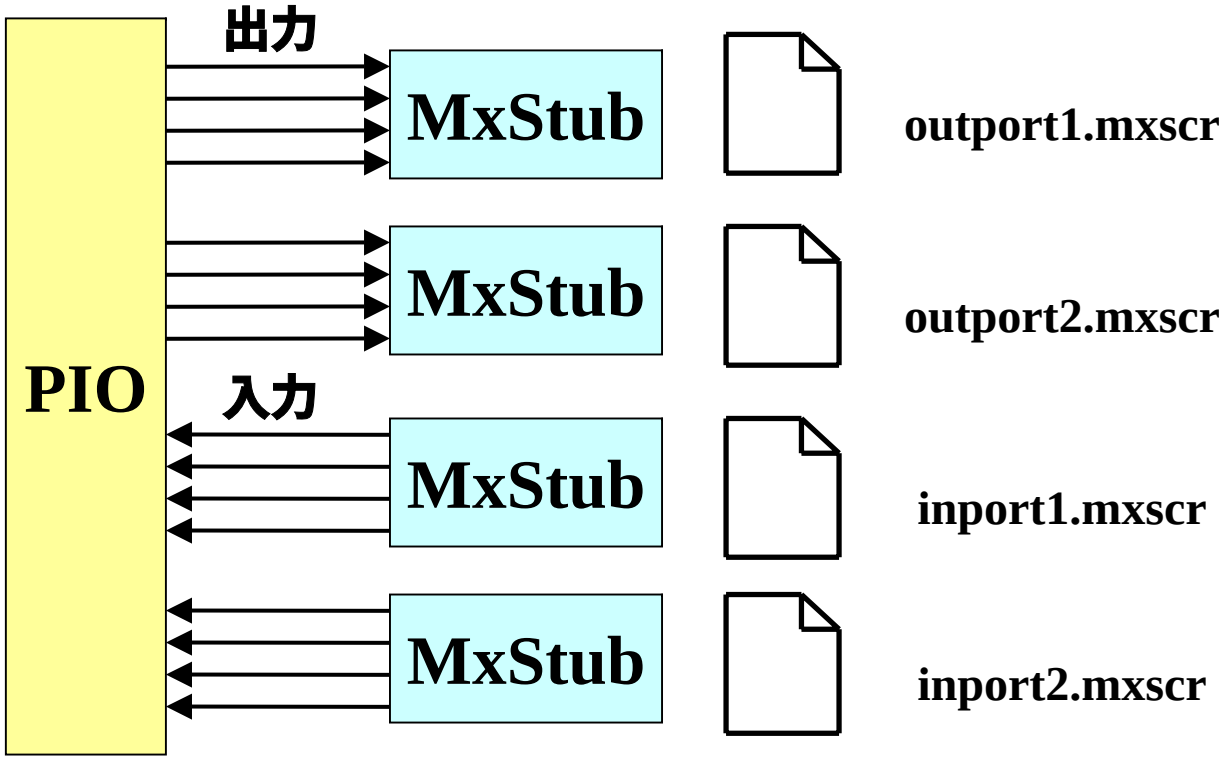
MxStubの例題を調べてみました

examples/MxStub

FUJI xerox

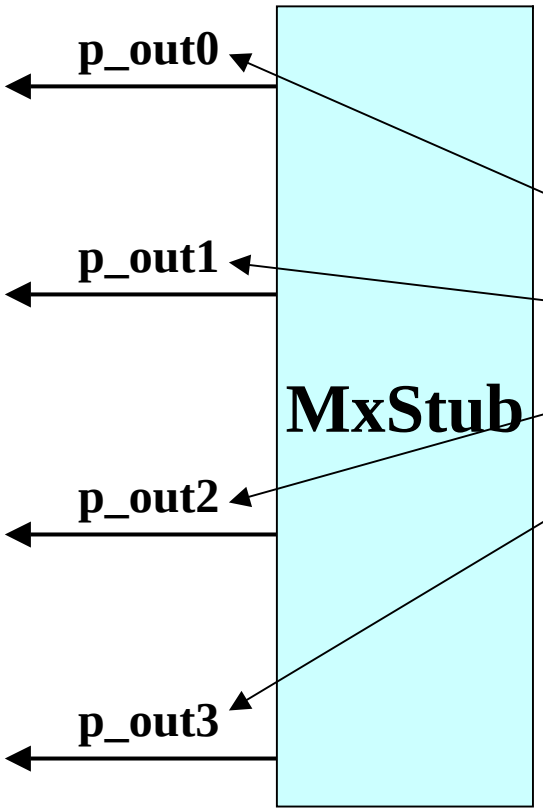
MxStubを利用してメカ部をモデル化！

出力/入力を各4本に分けて、
それぞれに対応したCarbon MxScriptを用意し、
そのスクリプトの記述にてモデル化を行った。



FUJI xerox

MxScriptの内容(設定部)



inport1.mxscr

```
int p_out0, p_out1, p_out2, p_out3;
int length = 100000;

p_out0 = getPortID("p_out0");
p_out1 = getPortID("p_out1");
p_out2 = getPortID("p_out2");
p_out3 = getPortID("p_out3");
```

ポート情報の獲得

```
message(MX_MSG_INFO,
        "GIO Initialized", 0);
```

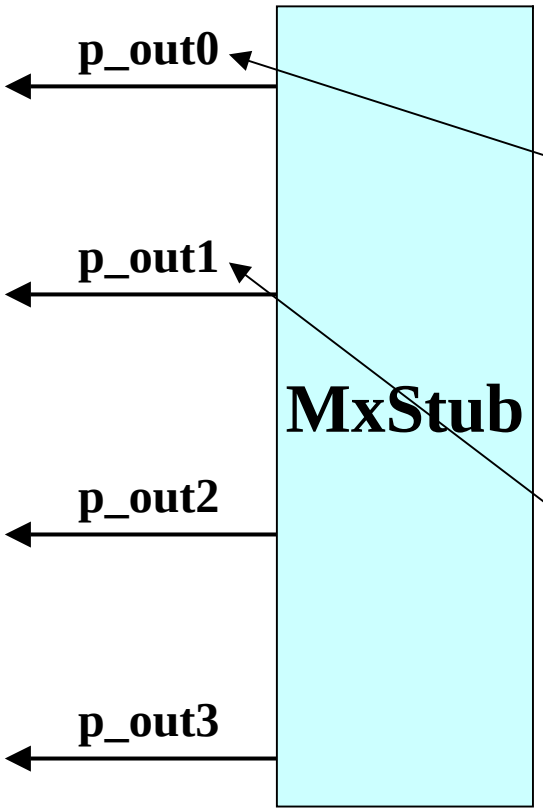
メッセージ表示

```
wait(100000);
```

ウエイト

FUJI Xerox

MxScriptの内容(信号制御部)



inport1.mxscr

```
while( true ){
  setPortData(p_out0, "DATA", 0x1);
  drivePort(p_out0);  wait(length);
  setPortData(p_out0, "DATA", 0x0);
  drivePort(p_out0);
```

P_out0を 0 => 1 => 0 のパルス生成

```
  setPortData(p_out1, "DATA", 0x1);
  drivePort(p_out1);  wait(length);
  setPortData(p_out1, "DATA", 0x0);
  drivePort(p_out1);
```

P_out1を 0 => 1 => 0 のパルス生成

```
  ....
```


FUJI xerox

4)、まとめ

まとめ

ここまでに行ったこと

- **SoC Designer上にメカコン部をモデル化**
- **MCU上のモニタ/タスクプログラムを実行**

今後の活動

- メカ部の詳細なモデル化をどうするか？
- メインシステム側からのメモリアクセスの帯域見積もり精度をどのように向上させるか？

Carbon Design Systems社への要望

- モデル化のための情報が少ない
  - => 日本語マニュアル
  - => アプリケーションノート

- **Windows版は、Visual C++ 2005のみ対応**
  - => **Visual C++ 2005は購入できない**
  - => **購入できるVisual Studio 2010に対応してほしい**

- **MxScriptは便利だが、**
  構文エラーはシミュレーションを起動しないとわからない
  - => 構文チェッカーを作ってほしい！


FUJI Xerox