

DSH-PLCSim PLC シミュレータ

2008年6月

株式会社データマップ

文書番号 DSHPLCSIM-08-30304-00



[取り扱い注意]

- ・ この資料ならびにソフトウェアの一部または全部を無断で使用、複製することはできません。
- ・ 本説明書に記述されている内容は予告なしで変更される可能性があります。
- Windows は米国 Microsoft Corporation の登録商標です。
- ・ MELSEC は三菱電機株式会社殿の登録商標です。
- ・ ユーザーが本ソフトウェアの使用によって生じた遺失履歴、(株)データマップの予見の有無を問わず発生 した特別損害、付随的損害、間接損害およびその他の拡大損害に対して責任を負いません。

改訂履歴

番号	改訂日付	項目	概 略
1.	2008.6	初版	
2.			
3.			
4.			







1.概要

本説明書は、DSH-PLCSim PLCシミュレータの役割、機能ならびに操作の概要について説明します。

サポートする PLC は、三菱電機(株)製 MELSEC シリーズの PLC で、バスインタフェース関数が使用できるものを対象としています。

[関連文書]

DSH-PLCSim関連文書一覧表 文書番号 注釈 # 文書名 DSH-PLCSim ユーザ・ーズ・カ゛イト゛ DSHPLCSIM-08-30304-00 この文書です。 1 2 DSHPLCSIM-08-30305-00 DSH-PLCSim 操作マニュアル 操作についての具体的な説明 書です。 装置コントローラ側が使用できる 3 DSH-PLCSim インタフェース関数説明書 DSHPLCSIM-08-30320-00 インタフェース関数の説明書です。 DSH-PLCSim PLC 制御デモ・プログラムについて VB6 のデモプログラムの説明書で 4 DSHPLCSIM-08-30390-00 す。 (VB6.0 言語版) DSH-PLCSim PLC 制御デモ・プログラムについて Visual C++のデモプログラムの説 5 DSHPLCSIM-08-30391-00 (VC++6.0 言語版) 明書です。



1.1 DSH-PLCSimの目的と役割

DSH-PLCSimの目的は、半導体製造工場内に設置される半導体製造装置の制御プログラムの開発をトータルシステム として総合的にシミュレーションできるようにするためのソフトウェアツールとして、きめ細かくその役割を果たす ことです。

DSH-PLCSim システムで提供されるソフトウェアはDSH-PLCSim シミュレータと DSHP1c ライブラリです。

下のブロック図は、システムの一例ですが、開発過程において、PLCの代わりに DSHPLC シミュレータを利用した構成を示しています。



この例のシステムの中で、DSHPLCシミュレータは、EQCONに接続される仮想のPLC(Programmable Logic Controller)、 キャリア ID リーダー、スロットマップリーダーの役割を果たします。

すなわち、DSH-PLCSimは、PLC、キャリア ID リーダー、スロットマップリーダーの役割と機能を論理的に受け持ち、 EQCON のプログラムテストを基本的なテストから総合的なシーケンスのテストまでのプログラム処理を、バグが無く なるまで繰り返して簡単に実行できるようにするための環境を提供するものです。

なお、装置コントローラは、プログラム開発とテスト段階では、MELSECのバスインタフェースライブラリプログラムの代わりに DSHPLC.DLL ライブラリプログラムを使用することになります。



1.2 構成

実機とDSH-PLCSimにおけるシステム構成は大体以下のようになります。



[DSH-PLCSimの構成]



DSH-PLCSimは、PLC, キャリア ID リーダー、スロットマップリーダーの基本的な3つの動作をシミュレートします。 EQCON との接続は、Ethernet を使って、TCP/IP プロトコルで行ないます。



1.3 機能概要

ここでは、①装置コントローラ、②PLCシミュレータのそれぞれについての機能を簡単に説明します。

1.3.1 装置コントローラの概略入出力機能

装置コントローラは、DSHP1c.dll ライブラリ関数を通して、PLC シミュレータとの入出力の制御を行ないます。

まず、接点入出力デバイス制御のために MELSEC が提供するバスインタフェース関数があります。



(1) バスインタフェース関数

QBF_Open() 関数を始めとしたバスインタフェース関数をサポートします。 MELSEC によって提供される関数のうち、Open, Close と入出力信号の Input, Output 関数のサポートを行な います。

他に、DSH-PLCSimでは、搬送単位であるキャリア ID リーダー機能と、その中に含まれるウェハー用スロットの有 無信号であるスロットマップリーダー機能が提供されます。(DSH-PLCSim 画面の仮想リーダーから読みます)



- (2) キャリア ID 読取り関数 UPLC_CarIdRead() 関数を使います。
- (3) スロットマップ読取り関数 UPLC_SlotmapRead()関数を使います。



1.3.2 PLC シミュレータ概略機能

PLC シミュレータ機能として、以下の機能があります。

I/0デバイス定義機能

入出力接点デバイス信号の定義を I/O マップ画面の操作でおこないます。 各信号には、名前とアドレスが与えられます。アドレスは入力、出力それぞれ 0~17FF(16 進)までの空間まで定義できます。 入力信号の名前には**"X_"**を、出力信号の名前には**"Y_"**を付けます。

また、定義された I/O デバイス名とそれに与えられたデバイス番号が対応付けられた C, VB6 用ソースファ イルが生成されます。EQCON のプログラムをヘダーファイルとして使用することができます。 入力信号定義画面例

🔚 Input	t Bit Device Definitio	n Editor – C:¥DshGer	Lib¥PlcSim¥	Sample1.	.pls
X-addr	0	1	2		3
0000					
001 0	X_Port1 RdyToLoad	X_Port1 StartLoad	X_Port1 LoadEn	d	X_Port1 C
0020	X_Port2RdyToLoad	X_Port2StartLoad	X_Port2LoadEn	d	X_Port20
0030	X_Port1 RdyToUnload	X_Port1 StartUnload	X_Port1UnloadB	End	
0040	X_Port2RdyToUnload	X_Port2StartUnload	X_Port2UnloadB	End	
0050	X_ProcessStart	X_ProcessEnd			
0060					
	C 用 hea VB6 用	ader file		EQCON 制御フ [°] 使用	ログラムで 可

(2)入出力制御機能

(1) で定義した I/O デバイスの I/O マップ制御画面上での操作であり、信号の On/Off の設定とその情報 をシミュレータに接続されている装置コントローラへ送信します。EQCON からは信号読込み関数を使って 信号の On/Off 状態を読み出すことができます。これらの操作はマウス操作で簡単にできます。

1/0定義	77-fil	8	65		C¥D	shGe	mLibi	VP1cS	ìim¥S	Sampl	le1.pl	8	1/0	操作	11 H	_	0	60710156	V	/0援	作样了		170	定网	7731	新 集	17	0774	1.67.9	终了		R258	去	Log 0
१_८ छ त	れな設立入力0n/0ff マップ ■回送 れた設立出力0n/0ff マップ																																	
ON	OF	FX	観	X_Po	rt1B	dyTol	Load			٠	•	llOn	A	lioff		司期	E	ON	OF	F N	訳	r_Plo	Star	rt				٠	A	llOn	A	llOff	6	199
X-addr 0000 0010 0020 0020 0040 0050 0050 0050 0070	• 0 0 - 0		-	3 - 0	- 0	-	- 0	-	-				C			r		Y-s4dr 0000 0010 0020 0030 0040 0050 0050 0050	0 0	-	-	-	-	-	-	-		9	A		C	D	<u> </u>	<u>·</u>
0000 0050 0060 0050 0000 0000 0000 0010																		0080 0030 0080 0080 0000 0000 0000 0000																
0100 0110 0120 0130 0140 0150 0150																		0100 0110 0120 0130 0140 0150 0160																
- 3170	_	_	_	_	_	_	_	_	-	-	-	-		_	-	-	-	0170		D.A.		_		_	_	_	-	-	-	_			_	-

これらの信号は、(4)で述べる簡易シーケンスプログラムによってシナリオに基づく制御も可能です。



(3) キャリアIDリーダー,スロットマップリーダーと処理データ入力機能

画面上にキャリアID、スロットマップ、プロセスデータの3種類の情報を設定できます。これらの情報は、EQCONから読み込むことができます。

٩c	キャリアID. スロット	₹7 _ □ X
-Ca	arrier ID	-Slotmap
	CARID_02	□ [slot-01]
1	CARID_01	IV slot-02
2	CARID_02	IV slot-04
3	CARID_03	🔽 slot-05
4	CARID_04	I I slot-06
5	CARID_05	IV slot-07
6	CARID_06	IV slot-09
7	CARID_07	🔽 slot-10
8	CARID_08	▼ slot-11
9	CARID_09	IV slot-12
10	CARID_10	IV slot-14
		□ slot-15
-7°	ከተፈ ታየ	🔽 slot-16
1		🔽 slot-17
1	ААААА	🔽 slot-18
2	BBBBB	🔽 slot-19
3	333.3	🗖 slot-20
4	444.4	IV slot-21
5	555.5	IV slot-22
6	666.6	□ slot-24
7	XXXXXX	🗖 slot-25
	確認	更新

(4) 簡易シーケンスプログラム画面による自動シーケンスの実行機能

簡単なコマンドを使って、表の中に信号の 0n/0ff 条件と設定を並べたり、ステップのジャンプ、時間遅延 などの制御を行ないながら、制御シーケンスを作り、それを実行させることができます。 実行すると、自動的に表上に組まれたプログラムを実行してくれます。これで一通りの搬入、処理、搬出 までのシナリオ制御を自動的に行なうことが可能です。

<mark>れ</mark> シーケン	ッス・プ	ログラミング - C:¥DshGemLib	¥PlcSim	n¥seq_300.prg					
-7ァイル	: 1				行ってもろし	tt ≤ 75			
新祝作成	<u> </u>		11未1子	利名 (1兼存 終) 10	仰に戻る	¥11 X797 2000ms ▼ 19 E			
補集 X_Port1R	**** X_Port1RdyToLoad ▼ 入力設定 Y_PlcStart ▼ 出力設定 行挿入 ファイル挿入 行削除 Beep Off ▼								
-1771	ステッフ°	設定対象信号名	On/Off	On/Off条件になる信号名	On/Off	 אנאב 			
ALLX	1	\$ALLX	Off			全Input signal = Off 📃			
ALLY	2	SALLY	Off			全Output signal = Off			
	3	\$SLEEP	3						
GOTO	4	404DID							
IFGO	0	DCARID	1						
	7	X Port 1BdyToL oad	0n	Y Port1Rdvlamp	0n	We CENCOOP の過点 Y Port1Rdvl amp = Op 待ち			
DELAT	8				on	Optortation for the start of th			
PAUSE	9	X Port1StartLoad	On						
WAIT	10	Y_Port1RdyLoadLamp	Off						
	11								
Carid	12	\$DELAY	2						
	13		-						
	14	X_Port1Clamped	On						
SlotMap	10	DELAT	4						
1 🗸	17	A_POPUISUARULOAD	UTT						
ProcData	18	\$WAIT	0n	Y_Port1FoupOpen	On				
	19								
	20	X_Port1FoupOpened	On			•			



2.装置コントローラの入出力機能

ここでは、EQCON における機能を詳細に説明します。

EQCON における機能は、対 DSH-PLCS im に対するインタフェース機能になります。 具体的には、EQCON の制御プログラムがどのように DSH-PLCS im との間で、提供される入出力インタフェース関数を 使ってプログラミングするかということになります。

MELSEC をサポートするバスインタフェース関数(QBF_xxxx の名前の関数)の機能を準備しています。

インタフェース関数としては、接点入出力のためのものと、DSH-PLCS im のセットアップのために準備された関数を 使用することができます。

また、ほかにDSH-PLCSimの固有の機能として、キャリア ID リーダー、スロットマップリーダーによる情報読取りのための関数も準備されていますので、そちらも使用することができます。

インタフェース関数の機能、使い方の詳しい内容については、「DSHPLC. DLL インタフェース関数説明書」に記述されていますので、そちらを参照してください。



EQCON と DSH-PLCS im との間のインタフェースは、DSHPLC. DLL ダイナミックライブラリに含まれる関数を使って取ります。

EQCON, DSH-PLCS im それぞれの DSHPLC. DLL には、I/O 情報を持っており、これらは、互いのインタフェース関数による制御によって、その内容の同期が取られています。

EQCON と DSH-PLCSim とは、TCP/IP プロトコルで接続されますが、Passive, Active の関係は以下のようになります。

EQCON: Passive (サーバー) DSH-PLCSim: Active(クライアント)



2.1 DSH-PLCSim 通信用インタフェース関数と使い方

EQCONがDSH-PLCSimとの間で使用される入出力信号のRead/Writeなどのインタフェース関数について説明します。

(1) バスインタフェース関数 (QBF_xxx)

DSHPLC. dl1 ライブラリが準備している MELSEC バスインタフェース関数として準備されている関数は次表の通りです。

	I/F 関数	機能
1	QBF_SetIOFile()	入出力定義ファイル名とDSH-PLCSim とのTCP/IP 通信のための情報を設定します。 DSH-PLCSim 特有の関数です。
2	QBF_Open()	^バ スまたは通信回線をオープンします。 DSH-PLCSim と接続します。
3	QBF_Close()	^バ スまたは通信回線をクローズします。 DSH-PLCSim との接続を切断します。
4	QBF_X_In_BitEx()	入力信号(X)を1点入力します。
5	QBF_X_In_WordEx()	入力信号(X)をワード単位で入力します。
6	QBF_Y_Out_BitEx()	出力信号(Y)を1点出力します。
7	QBF_Y_Out_WordEx()	出力信号(Y)をワード単位で出力します。
8	QBF_Y_In_BitEx()	出力信号(Y)を1点出力します。
9	QBF_Y_In_WordEx()	出力信号(Y)をワード単位で出力します。
10	QBF_ToBuf()	デバイスメモリにデータを書き込みます。 DSH-PLCSimでは、データ部分のみを書込み対象とします。
11	QBF_FromBuf()	デバイスメモリからデータを読み出します。 DSH-PLCSimでは、データ部分のみを読出し対象とします。
12	QBF_RefreshLinkDevice() QBF_WriteLinkDevice() QBF_SEND() QBF_RECV() QBF_UnitInfo() QBF_StartWDT() その他の関数	これらの関数については、何も処理しません。 返却値が必要な関数については、全て = 0 を返却します。



(2) その他の関数 (DSH-PLCSim との通信のための)

DSH-PLCSim シミュレータに対する固有の関数

	I/F 関数	機能
1	UPLC_CkCommReady()	DSH-PLCSim との接続が確立しているかどうかを調べるための関数です。
2	UPLC_XBitSyncReq()	DSH-PLCSimが有する入力信号(X)のデータを読出し、デバイスコントローラ側のデバイスメモリに格納します。
3	UPLC_YBitSyncReq()	DSH-PLCSimが有する出力信号(Y)のデータを読出し、デバイスコントローラ側のデバイスメモリに格納します。
4	UPLC_CarIdRead()	キャリア ID を読出します。 読出すキャリア ID は、DSH-PLCS im の画面上に選択設定された値です。
5	UPLC_SlotmapRead()	スロットマップ情報(25 スロット分)を読出します。 読出すスロットマップ情報は、DSH-PLCSimの画面上に選択設定された On/off 情報です。
6	UPLC_ProcDataRead()	指定した番号のプロセスデータを1個読出します。 読出すプロセスデータは文字列データです。 画面には7個のデータが設定されています。
7	UPLC_GetXDvName()	入力デバイス番号からそのデバイス名を取得する。
8	UPLC_GetYDvName()	出力デバイス番号からそのデバイス名を取得する。



2.2 インタフェース関数によるプログラミング手順

EQCON における、バスインタフェース関数のプログラミング手順は次の通りです。





3.PLC シミュレータの機能

DSH-PLCSim シミュレータは、PLC(シーケンサー)の機能をシミュレートするものです。

3.1 I/O デバイス定義機能

I/0 デバイスの定義機能とは、PLCのX,Y デバイスメモリの各ビットをどのような信号名として使用するか、そのアドレス(=デバイス番号)と信号名を対応付け、XY メモリのマップ上に割り付ける機能です。

システムの外部仕様が決まり、PLC の信号をどのように使用するかの I/O の割付が仕様上で決定されたものを DSH-PLCSim の I/O アドレスマップ画面上で入力操作し、割付情報の結果をディスクファイル上に保管します。

また、一度作成保存した I/0 定義ファイル内で定義した内容を変更、追加などの編集も I/0 定義操作画面上で操作 することができます。

(1) I/0 定義画面

下に示す画面は、入力デバイスの設定画面です。 X_addr (16 進で 4 桁) に対応するセルを選択し、そこに入力デバイス名を入れます。

🔚 Input	🔚 Input Bit Device Definition Editor – C:¥DshGemLib¥PlcSim¥Sample1.pls									
X-addr	0	1		3						
0000										
001.0	X_Port1 RdyToLoad	X_Port1 StartLoad	X_Port1 LoadEnd	X_Port1 C						
0020	X_Port2RdyToLoad	X_Port2StartLoad	X_Port2LoadEnd	X_Port2C						
0030	X_Port1 RdyToUnload	X_Port1 StartUnload	X_Port1UnloadEnd							
0040	X_Port2RdyToUnload	X_Port2StartUnIoad	X_Port2UnloadEnd							
0050	X_ProcessStart	X_ProcessEnd								

X_addrのセルのダブルクリックによって、ポップアップされる次画面を使って定義することもできます。

Ac Input Signal	Setting	×
Address	33	
Name	XJ	
Default(On/Off)	Off 💌	
Caption		
	OK	Cancel

(2) 生成されるファイル

画面に定義された I/0 定義情報をファイルに保存すると、下図に示すようなファイルが生成されます。





ファイル保存時に、例えば、ファイル名を"sample.pls" と指定された場合に生成されるそれぞれのファイル名の種類と内容は以下のようになります。

	保存ファイル名	拡張子
1	sample.pls : I/O 定義のテキストファイルです。保存操作時に指定さ	.PLS
	れた名前で保存されます。	
2	sample.plo: sample.plsをコンパイルした結果ファイルです。	.PLO
	I/0 制御操作画面開始時にしようされます。	
3	sample.h : C/C++言語プログラム用のヘダーファイルです。	.h
	I/0の名前に対するアドレス値のマクロ定義です。	
4	sample.bas : VB6 用プログラム用のファイルです。	.bas
	I/0 の名前に対するアドレス値のマクロ定義です。	

表の3,4の.h,.bas ファイルは、EQCON制御プログラムのソースファイルとして使用できるようにフォーマットが整えられて生成されます。

(3) ファイルの内容(例)

① .pls - I/0 定義ソースファイル

// Inp	ut Device	Bit Definition
def_in_bit	X_Port1R	dyToLoad {
	addr:	0x10
	nominal:	0
	caption:	""
	}	
def_in_bit	X_Port1S	tartLoad{
	addr:	0x11
	nominal:	0
	caption:	""
	}	
// Out	put Device	e Bit Definition
// Outj def_out_bit	put Device Y_PlcSta	e Bit Definition rt{
// Outj def_out_bit	put Device Y_PlcSta addr:	e Bit Definition rt{ 0x1
// Outj def_out_bit	put Device Y_PlcSta addr: nominal:	e Bit Definition rt{ 0x1 0
// Outj def_out_bit	put Device Y_PlcStan addr: nominal: caption:	e Bit Definition rt{ 0x1 0 ""
// Outj def_out_bit	put Device Y_PlcSta: addr: nominal: caption: }	e Bit Definition rt{ 0x1 0 ""
// Outj def_out_bit def_out_bit	<pre>put Device Y_PlcStar addr: nominal: caption: } Y_PlcStop</pre>	e Bit Definition rt{ 0x1 0 ""
// Outj def_out_bit def_out_bit	<pre>put Device Y_PlcStar addr: nominal: caption: } Y_PlcStop addr:</pre>	e Bit Definition rt{ 0x1 0 "" o{ 0x2
// Outj def_out_bit def_out_bit	<pre>put Device Y_PlcStar addr: nominal: caption: } Y_PlcStop addr: nominal:</pre>	e Bit Definition rt{ 0x1 0 "" o{ 0x2 0
// Outj def_out_bit def_out_bit	<pre>put Device Y_PlcStar addr: nominal: caption: } Y_PlcStop addr: nominal: caption:</pre>	e Bit Definition rt{ 0x1 0 "" o{ 0x2 0 ""

② .plo – I/0 定義オブジェクトファイル

INB:NAME00000010X_Port1RdyToLoadADDR00000010NOMI0000000CAPT00000000 INB:NAME00000010X_Port1StartLoadADDR00000011N0MI0000000CAPT00000000 INB:NAME0000000eX_Port1LoadEndADDR00000012N0MI0000000CAPT00000000 INB:NAME00000001X_Port1C1ampedADDR00000013N0MI00000000CAPT00000000 INB:NAME00000011X_Port1FoupOpenedADDR00000014N0MI00000000CAPT00000000 INB:NAME00000011X_Port1FoupClosedADDR00000015N0MI00000000CAPT00000000 INB:NAME00000013X_Port1FoupClosedADDR00000016N0MI00000000CAPT00000000 INB:NAME00000013X_Port1MovedCarWorkADDR00000016N0MI00000000CAPT00000000 INB:NAME00000012X_Port1MovedCarDstADDR00000017N0MI00000000CAPT00000000



③ .h - C/C++用ファイル

/*			*/	
/* C:¥D	shGemLib¥P1cSim¥Samp1e	1.h - PlcSim Addr Defi	nition*/	
/*			*/	
#define	X_Port1RdyToLoad	0x0010	// 16	
#define	X_Port1StartLoad	0x0011	// 17	
#define	X_Port1LoadEnd	0x0012	// 18	
#define	X_Port1Clamped	0x0013	// 19	
#define	X_Port1Foup0pened	0x0014	// 20	
#define	X_Port1FoupClosed	0x0015	// 21	
#define	X_Port1MovedCarWork	0x0016	// 22	
#define	X_Port1MovedCarDst	0x0017	// 23	
#define	X_Port1LoadReq	0x0018	// 24	
#define	X_Port2RdyToLoad	0x0020	// 32	
#define	X_Port2StartLoad	0x0021	// 33	

④ . bas - VB6 用ファイル

Attribute VB_Name = "PLC_addr.bas"		
, C:¥DshGemLib¥PlcSim¥Sample1.bas - P.	lcSim Add	hr definition
Public Const X_Port1RdyToLoad =	16	' 0x00000010
Public Const X_Port1StartLoad =	17	' 0x00000011
Public Const X_Port1LoadEnd =	18	' 0x00000012
Public Const X_Port1Clamped =	19	' 0x00000013
Public Const X_Port1FoupOpened =	20	' 0x00000014
Public Const X_Port1FoupClosed =	21	' 0x00000015
Public Const X_Port1MovedCarWork =	22	' 0x00000016
Public Const X_Port1MovedCarDst =	23	' 0x00000017
Public Const X_Port1LoadReq =	24	' 0x00000018
Public Const X_Port2RdyToLoad =	32	' 0x0000020
Public Const X_Port2StartLoad =	33	'0x00000021



3.2 I/O 制御画面での制御機能

3.1で作成した I/O 定義ファイルの入出力制御を実際に EQCON と接続して実行する機能です。

3.2.1 I/O 制御の開始

I/O 定義ファイルをメイン画面で指定し、I/O 操作開始操作を行なうと、DSH-PLCS im は次のような準備処理をします。

- (1) 相手の EQCON との TCP/IP の接続を行ないます。
 TCP/IP の接続上は、EQCON が Passive で、DSH-PLCS im が Active として接続します。
 (注) DSH-PLCS im 側では、接続のための EQCON の IP と ポートを通信設定メニューで設定できます。
- (2)入力、出力のマップ画面を別々に表示し個別に I/0 信号を選択し、0n/0ff の制御ができるようにします。

1/0定義7	1/0定義7+18名 参照 C#DshGemLb¥PkSim¥Sample1.pla 1/0接作符化 m 6807前556 1/0接件件了 1/0定规7555度 1/0万元55度 1/0万元55度 1/0万元55度 1/0万元55度																																		
<mark>礼(</mark> 根点)	№ 接点入力0n/0ff マップ																																		
ON	OF	E X	依	X_Po	rtiß	dy/Tol	Load			¥	A	llOn	A	lioff	1	同期		ON		OFF	澎	訳	r_P1	Sta	r۱				٣	A	llOn	A	lioff	6	349
X-addr		1	2	3	4	5	6	7	8	3			C	0	£	F	-	Y-ade	r	0	1	2	3	4	5	6	1	8	3			¢	D	E	F A
0000	8	-	-	-	-	-	-									-		000	0	-	-	-	-	-			-								
0020	0	-	-	0	0	-	0	-	-									002	0	ŏ	-	-	-	-	-	-	-								
0030	-	-	-			_				_						_		003	0	-															
0040	0	-	-			-	-	-	-	-		-				-		004	0	-							-								
0000	Ŭ																	000	0	-															
0070																		007	0																
0000						-	-	-	-	-		-				-		001	0																
0040									-							-		003	0																
0080																		008	0																
0000						-	-	-	-	-		_				-		000	0																
0000									-							-		000	0																
OOFO																		001	0																
0100																		010	0																
0110						-	-	-	-	-		-				-			0																
0130																		011	0																
0140																		014	10																
0150						-	-	-	-	-		-				-		013	0																
0170									-							-			0																
																	-	_	-																_

マップ画面上では、信号が On になっているものは、Oで表示され、Off になっているものは – で表示されます。



3.2.2 I/O 制御の内容と操作

DSH-PLCSimにおける入力と出力信号の操作は、それぞれの画面で行ないますが、操作の方法は同じです。

(1) 信号の 0n/0ff 制御

2つの方法があります。

①信号名をコンボボックスで選択し、ON または OFF ボタンをクリックする方法

②目的の信号が割当てられているマップ上のセルをダブルクリックする方法

この場合、On<->Off が交互に変化します。

また、全信号を ON/OFF したい場合は、それぞれ、ALLON、ALLOFF ボタンのクリックします。

操作された信号の状態は、I/0マップ上の当該セルに 0n の場合は、"○"で、0ff の場合は "-" が表示されます。

- (注)入出力デバイスに対する On/off 制御は3. 4 で説明するシーケンスプログラムによっても制御 できます。
- (2) EQCON への送信

(1) で制御された入力信号は、EQCON に送信され、EQCON の DSHPLC. DLL が有する入力または出力デバイスメモリに反映されます。(ただし、DSH-PLCSim と EQCON が接続状態である必要があります。)



(3) I/0 操作のログ表示

(1) で行なわれた操作は、ログ表示画面に随時表示されます。 表示と同時に、dshplc.log ファイルに記録保存されます。

3.2.3 装置コントローラからの I/O 要求応答機能

EQCON との通信確立の後、EQCON からの入出力要求に対し、DSH-PLCS im は自動的に応答します。

(1)入力デバイスの読出し要求

EQCONからは、入力デバイスの1点入力、データ単位(16ビット)の入力に対し、DSH-PLCSimは、I/0デバイスメモリに保存されている情報を自動的に応答します。

(2) 出力デバイスに対する制御情報

EQCONから送信されてくる出力デバイスの1点出力、データ単位(16ビット)の出力に対し、DSH-PLCSim は送られてきた情報をI/0デバイスメモリに反映させるとともに、出力画面のマップ表示にも反映させま す。



3.3 キャリアID、スロットマップ、プロセスデータ操作機能

本機能は、PLC とは直接関係がありませんが、以下の情報に対する設定と EQCON の読出し要求に対する応答機能があります。

- 1. キャリア ID の設定、選択と、EQCON への応答
- 2. スロットマップ(収納されているウェハーの有無)情報の設定と、EQCONへの応答
- 3. プロセスデータの設定と、EQCONへの応答

٩c	キャリアID. スロット	₹7 ∎□×
_Ca	arrier ID	Slotmap
	CARID_02	🗖 (slot-01)
1	CARID_01	Slot-02
2	CARID_02	slot-03 ▼ slot-04
3	CARID_03	Iv slot-05
4	CARID_04	🔽 slot-06
5	CARID_05	I Slot-07
6	CARID_06	IV SIOT-08
7	CARID_07	IV slot-10
8	CARID_08	🔽 slot-11
9	CARID_09	I Slot-12
10	CARID_10	IV slot-13 IV slot-14
	, - ,	slot-15
⊏7°	ロセス データー・・・・	🔽 slot-16
1	AAAAA	slot-17
2	BBBBB	IV slot-18
3	333.3	□ slot-20
4	444.4	🔽 slot-21
5	555.5	▼ slot-22
6	666.6	IV slot-23
7	XXXXX	□ slot-25
	確認	

「キャリア ID]

10 個分準備されており、 それらの中の1つを選択します。 や入力でキャリア ID を変更できます。 選択はンーケンスプログラムでも可能です。

[Slotmap]

25 スロット分準備されており、各スロット の在荷状態を変更できます。 在荷状態の変更はンーケンスプログラムでも 可能です。

[プロセスデータ]

7 個のデータが準備されています。 キー入力でデータを変更できます。 プ 咜ヒスデータの変更はシーケンスプログラム でも可能です。

EQCON との情報のやり取りは次のようになります。





3.4 簡易シーケンスプログラム機能

ここまでは、I/O 制御画面上での操作について説明してきましたが、もう一つ、強力な機能としてシーケンスプロ グラムによる I/O の自動制御があります。

この機能は、画面に表示される表の中に、コマンドとパラメータを設定し、表の上から順にコマンド群から成るプログラムを実行するものです。実行ステップをビジュアルに確認しながら EQCON の機能をテストすることができます。

プログラム画面はつぎのような画面です。この画面上でプログラミングと実行の全ての操作ができます。

<mark>れ</mark> シーケ	はシーケンス・プログラミング - C:¥DshGemLib¥PlcSim¥seq_300.prg										
ー Jァイル 新規作の	ŧ 📘	開く (開きなおす) 上書	保存	別名で保存 終了 て	iT opに戻る	実行 ステップ2000ms ▼ 停止					
編集 X_Port1R	dyToLo	ad 💽 入力設定 Y	_PlcStar	't J 出力設定	2 行报	间入 7ァイル挿入 行削除 Beep Off ▼					
-1771	7:507	設定対象信号名	On/Off	On/Off条件になる信号名	On/Off	▲ // ۲/۲					
ALLX	1	\$ALLX	Off			全Input signal = Off 📃					
ALLY	2	\$ALLY \$SLEEP	Off 3			全Output signal = Off					
GOTO	4	******	-								
IFGO	<u> </u>	\$CARID	1			T+971D 先頭(CARID_U1) ** ここがLoop の起点					
DELAY	7	X_Port1RdyToLoad	On	Y_Port1Rdy1amp	On	Y_Port1RdyLamp = On 待ち					
DALISE	8					OnになったらX_Port1StartLoad=(
FAUSE	9	X_Port1StartLoad	On								
WAIT	10	Y_Port1RdyLoadLamp	Off								
Carid	12	\$DELAY	2								
1 🗸	13	V. Dauk 10 Januard	0-								
our second	14		UN A								
SlotMap	16	X Port 1Start Load	4 Off								
1 -	17										
ProcData	18	\$WAIT	On	Y_Port1Foup0pen	0n						
	19										
	20	X_Port1FoupOpened	On			•					

プログラムコマンドとして以下のものがあります。

	コマント	機能
1	X_signal On/Off	入力信号の 0n/0ff、他の信号の 0n/0ff の条件付き 0n/0ff も可能です。
2	Y_SIgnal On/Off	出力信号の On/Off、他の信号の On/Off の条件付き On/Off も可能です。
3	\$ALLX	入力信号を全て On/Off にします。
4	\$ALLY	出力信号を全て 0n/0ff にします。
5	\$DELAY	秒単位で処理を遅らせます。
6	\$GOTO	無条件に別のステップにジャンプします。
7	\$IFGO	信号の On/Off 条件によってジャンプします。
8	\$WAIT	信号が On/Off になるのを待機します。
9	\$PAUSE	実行を一旦停止します。
10	\$CARID	キャリア ID の選択をします。
11	\$SLOTMAP	スロットマップ 情報を変更します。
12	\$PROCDATA	プロセスデータを変更します。

詳しくは、4章で説明します。



3.5 ログ表示機能

画面上で操作された、例えば入出力のOn/Off 制御した内容は、ログ画面に随時表示されます。 また、EQCON から要求された要求内容も同様に表示されます。

そして、これらのログ情報は、ログファイル上にログメッセージにタイムスタンプ付きで記録されます。

[ログ画面]

Ac PLC Operation Log	
** StartPlcMon(opt=1 **	
** UPLC Open(5930, 192.168.1.2, 1) **	
** 2008706/10 13:26:38> Connected with the Application	
*OUT - addr: 1 -> On (Y_PlcStart)	
*** program Start ***	
*IN - addr: 0 -> On (??)	
*IN - addr: 10 -> On (X_Port1RdyToLoad)	
*IN addr: 11 -> On_(X_PortIStartLoad)	
*OUT - addr:10 -> Off (Y_Port1RdyLoadLamp)	
* CARID = CARID_01 sent	
*NN - addr: 13 -> On (X_PortIClamped)	
#UUI - addr: 10 -> Un (I FortlKdyLoadLamp)	
#UUI - addr: 2U -> Un (I_Port2kdyLoadlamp)	
** Sync Request Acceptea, will try to sena all A memory	
MUUI - addr. 10 -/ UN (I_FORTINGYLOAdlamp)	
MOUT - addr. 20 -/ On (I_rortzkayLoadlamp)	
moor = addr: 10 -> 0n (1_rorthuyLoadamp)	
W (ARI) = (ARI) (1 - on (1 - or (LAGYLGAULAMP)	
WIN - addr: 11 -> Off (Y Port StartLand)	
*(N - addr: 14 -> 0n (X - Port)FoupDepend)	
	<u> </u>
62	1

[ログファイル]

```
08-06-10 13:24:26 ** StartPlcMon( opt=1 **
08-06-10 13:24:26 ** UPLC_Open(5930, 192.168.1.2, 1) **
08-06-10 13:26:38 ** 2008/06/10 13:26:38 -> Connected with the Application
08-06-10 13:26:39 *0UT - addr:
                                1 \rightarrow 0n (Y_PlcStart)
08-06-10 13:26:40 *** program Start ***
08-06-10 \ 13:26:40 \ *IN - addr: 0 \rightarrow On (??)
08-06-10 13:26:40 *IN - addr: 10 -> On (X_Port1RdyToLoad)
08-06-10 13:26:40 *IN - addr: 11 -> On (X_Port1StartLoad)
08-06-10 13:26:40 *OUT - addr: 10 -> Off ( Y_Port1RdyLoadLamp )
08-06-10 13:26:40 * CARID = CARID 01 sent
08-06-10 13:26:42 *IN - addr: 13 -> On (X_Port1Clamped)
08-06-10 13:26:43 *0UT - addr:
                               10 \rightarrow 0n (Y_Port1RdyLoadLamp)
08-06-10 13:26:43 *0UT - addr:
                                20 \rightarrow 0n (Y_Port2RdyLoadLamp)
08-06-10 13:26:43 ** Sync Request Accepted, will try to send all X_memory
08-06-10 13:26:43 *OUT - addr: 10 -> On (Y_Port1RdyLoadLamp)
08-06-10 13:26:43 *OUT - addr: 20 -> On (Y_Port2RdyLoadLamp)
08-06-10 13:26:43 *OUT - addr: 10 -> On (Y_Port1RdyLoadLamp)
08-06-10 13:26:43 *OUT - addr:
                                20 \rightarrow 0n (Y_Port2RdyLoadLamp)
08-06-10 13:26:43 * CARID = CARID_01 sent
08-06-10 13:26:45 *OUT - addr: 14 -> On (Y_Port1FoupOpen)
08-06-10 13:26:47 *IN - addr: 11 -> Off (X_Port1StartLoad)
08-06-10 13:26:47 *IN - addr: 14 -> On (X_Port1FoupOpened)
```



4.簡易シーケンスプログラムによる自動テスト機能

I/0制御をプログラムのシーケンスに従って自動的にテスト実行させるための機能です。

シーケンスを実行するには、ある出力デバイスが 0N になったらという条件で、ある入力デバイスを 0N にするよう な判断した上で制御するというような論理で進める必要があります。また、実機に近い形でテストするには、時間的 な遅延などを行なう必要もあります。

このような自動テスト機能を簡易なコマンド(命令)で簡単にプログラミングし実行することを可能にする機能で す。キャリア ID, スロットマップ、プロセスデータに対するプログラミングもできます。

4.1 プログラミングと操作画面

画面上で、入出力選択用コンボボックス、ボタン、マウスとグリッドへのキー入力操作でプログラミングします。

<mark>የ</mark> ይ-ታጋ	シス・プログラミング					_		_	
-7ァイル 新規作成	i 開く	開きなおす 上	書保存	別名で保存	第7 下の	î⊐ oplE戻る┃	実行 ステシ	7° 2000ms 🖵	停止
編集 X Port1R	beo IoTvb		V PloStar	•+	■ 出力設定	2 行捕	入 ファイル挿入	行用(R全 Beep ()ff -
-17:16				v In Versenville			- 0.1	TTHAKE	
-1571	がり川設定対象的	青安治	Un/Utt	Un/Utt条件に	なる信ち名	Un/Utt	40%		▲
	2								
ALLY	3								
GOTO	4								
IFGO	5								
	7								
	8								_
PAUSE	9								
WAIT	10								
Carid	12								
1 🗸	14								
SlotMap	15								
	16								
	17								
ProcData	18								
1 🗸	19								

(1) プログラム設定グリッド(表)画面

画面の表には、6個の列がありますが、それぞれの列の用途は次の通りです。

列	見出し	意 味
1	ステップ	プログラムの実行ンーケンスです。実行は、正順に進みます。
2	設定対象信号名	On/Off したい信号名または制御コマント を設定します。
3	On/Off	設定対象信号の On/Off または制御マット、のパラメータを設定します。
4	0n/0ff 条件になる信号名	設定対象信号を 0n/0ff にする際に他の信号の 0n/0ff 条件になります。
		また、制御コマンドにパラメータにも使用されます。
5	0n/0ff	条件になる信号の On/Off 条件を設定します。
6	コメント	注釈の欄です。プログラム実行上の意味はありません。



(2)入出力デバイス名選択コンボボックス

3. 1で説明した I/0 デバイス定義ファイルに登録された入出力信号名が入力、出力それぞれにコンボボックスのリストに表示されます。

コンボボックスをプルダウンし、コマンドとして設定したい信号名を選択します。そして、人力設定また は出力設定ボタンを使って所望のステップと列に信号名を設定します。

M_Port1RdyToLoad	▼ 入力設定	Y_PlcStart	▼ 出力設定
X Port1RdyToLoad X_Port1StartLoad X_Port1LoadEnd	<u>_</u>	On/Off On/Off≸	条件になる信号名 0
X_Port1Clamped X_Port1FoupOpened X_Port1FoupClosed			
X_Port1MovedCarWork X_Port1MovedCarDst	-		

[設定例]

ステッフ゜	設定対象信号名	0n/0ff	0n/0ff 条件になる信号名	0n/0ff	
1					
2	X_Port1StartLoad	On	Y_Port1RdyToLoad	On	
3			Y_Port1RdyLoadLamp	On	
4					
5	\$DELAY	5			
6	Y_Port1RdyToLoad	Off			

上の例は、Y_Port1RdyToLoad=On で、かつ、Y_Port1RdyLoadLamp=On の状態になったら X_Port1StartLoad=On にし、ステップを5 に進め、5 秒間遅延の後、ステップ-6 に進みます。 因みに、通常、条件になる出力信号の On は EQCON が制御します。 (ブランクのステップは無処理 (no operation) となり、次のステップに進みます。)

(3) 制御コマンドボタン

制御コマンドの設定は制御ボタンを使って行ないます。 コマンド名は、設定対象信号名の列に設定されます。 そして、コマンドのパラメータはコマンドによって所定の列に設定することになります。 コマンド名は、大文字で、ボタンの表示名の頭に**\$**を付けたものになります。

設定対象信号名	0n/0ff	0n/0ff 条件になる信号名	0n/0ff	機能
\$ALLX	0n/0ff			全入力デバ イスを 0n/0ff する。
\$ALLY	On/Off			全出力デバイスを 0n/0ff する。
\$DELAY	<t></t>			<t>秒間遅延します。</t>
\$GOTO	<step></step>			<step>位置に無条件ジャンプします。</step>
\$IFGO	<step></step>	<i 0="" 信号名=""></i>	0n/0ff	<i 0="" 信号名="">の 0n/0ff 条件で<step>ジャンプします</step></i>
\$WAIT		<i 0="" 信号名=""></i>	0n/0ff	<i 0信号名="">の0n/0ff条件になるまで待機します。</i>
\$PAUSE				一旦停止します。
\$CARID	<index></index>			<index>(1~10)のキャリア ID を選択します。</index>
\$SLOTMAP	<index></index>		0n/0ff	<index>のスロットを 0n/0ff します。</index>
\$PROCDATA	<index></index>	〈データ値〉		<index>(1~7)の値を<データ値>にセットします。</index>

(注) <データ値>は文字列データです。



4.2 編集と保存

(1) 編集

編集ボタンブロックを使って操作します。

相太			
X_Port1RdyToLoad	「人力設定」 Y_PlcStart	▼ 出力設定	行挿入 7ヵ4批挿入 行削除

4. 1のプログラミング操作の中で、コマンドの挿入、削除などの編集作業が必要になります。

ステップ行の挿入は、行挿入ボタン、ステップ行の削除は、行削除ボタンのクリックで行ないます。行挿 入ではブランク行が挿入されます。また、行削除では、そのとき選択されているステップ行を削除し、後 ろのコマンド群を前方向にプルアップします。

この際、\$GOT0, \$IFG0 コマンドのジャンプ先は、挿入、削除によって変わる可能性がありますが、そのジャンプ先の調整も自動的に行われます。

既に作成されているファイルを行の間に挿入したい場合は、77個挿入ボタンを使用します。この場合の挿入によって影響を受ける\$60, \$IFG0のジャンプ先ステップの調整も自動的に行なわれます。

(2)保存形式と保存

ファイルボタンブロックのボタンを使用します。

-7-74元 | 新規作成 | 関く | 関きなおす | 上書保存 | 別名で保存 | 終了 |

画面上にプログラミングされたプログラムをファイルに保存することができます。また、保存したプログ ラムを再度開いて使用することができます。

プログラムファイルは、拡張子.prg で保存されます。

形式は、プログラム画面の表に表示されているステップ順に、2列目から6列目までのセルの内容をカンマ切りされて保存されます。

下に、サンプルを示します。

\$ALLX,	0ff,	,	,	全Input signal = Off,
\$ALLY,	0ff,	,	,	全Output signal = Off,
\$SLEEP,	3,	,	,	,
,	,	,	,	,
\$CARID,	1,	,	,	キャリア ID 先頭(CARID_01),
,	,	,	,	** ここがLoop の起点,
X_Port1RdyToLoad,	0n,	Y_Port1Rdylamp,	0n,	Y_Port1RdyLamp = On 待ち,
,	,	,	,	On になったら X_Port1StartLoad=On,
X_Port1StartLoad,	0n,	,	,	,
Y_Port1RdyLoadLamp,	0ff,	,	,	,
,	,	,	,	,
\$DELAY,	2,	,	,	,
,	,	,	,	,
X_Port1Clamped,	0n,	,	,	,
\$DELAY,	4,	,	,	,
X_Port1StartLoad,	0ff,	,	,	,
,	,	,	,	,
\$WAIT,	0n,	Y_Port1Foup0pen,	0n,	,
,	,	,	,	,
X_Port1Foup0pened,	0n,	,	,	,
Y_Port1Foup0pen,	Off,	,	,	,
<pre>\$DELAY, , X_Port1Clamped, \$DELAY, X_Port1StartLoad, , \$WAIT, , X_Port1FoupOpened, Y_Port1FoupOpen,</pre>	2, , On, 4, Off, , On, , On, Off,	, , , , Y_Port1FoupOpen, ,	, , , , , , , , , , , , ,	<pre> , , , , , , , , , , , , , , , ,</pre>



4.3 シーケンスプログラムの実行と停止

実行はボタンブロックのボタンのクリック行ないます。

実行 Topに戻る 実行 ステップ 2000ms ↓ 停止

(1) 実行

実行開始は、画面上の開始したいステップを選択し、実行ボタンのクリックで開始します。 実行中は、コマンドとコマンドの実行にインターバルをとることができます。 ステップボタンの右横にある インターバル(単位はms)の値を選択できます。このことによって、実行状態を目視しながらステップを 確認しテストすることができます。

ステップ ボタンを使うとコマンドを1個ずつステップ順に実行できます。

EQCON との通信が必要なコマンドの実行の条件として、DSH-PLCSim と EQCON との間で TCP/IP 通信接続が確立していることが必要です。接続しているかどうかの確認は、メニューバーの下のランプで確認できます。



(2) 停止

プログラムの実行停止は、一旦停止も含め、次のことによって停止します。 ①プログラムが終端に達したとき。 ②停止ボタンがクリックされたとき。 ③\$PAUSE コマンドに達したとき。