

# PPDからCPDへ

PCI版移行ガイド

ソフトウェア編

第1.0版 2011/5/16

# はじめに

CPDシリーズは円弧補間・直線補間・位置決め制御ボードです。

旧タイプのPPDシリーズと比べて円弧補間・直線補間機能が追加され、コンパレータ機能、原点復帰等が強化されています。

また、CPDシリーズとPPDシリーズはレジスタ構成やステータス構成、添付されるソフトウェアも異なります。

本資料はソフトウェアを作成する上でのPCI版PPDシリーズとCPDシリーズの主な差異についてまとめたものです。またPPDシリーズに標準添付されるAPI関数を使用している前提で説明しています。

実際にCPDシリーズ制御ソフトウェアを作成される時は、CPDシリーズ各種マニュアルをご参照ください。

# Contents

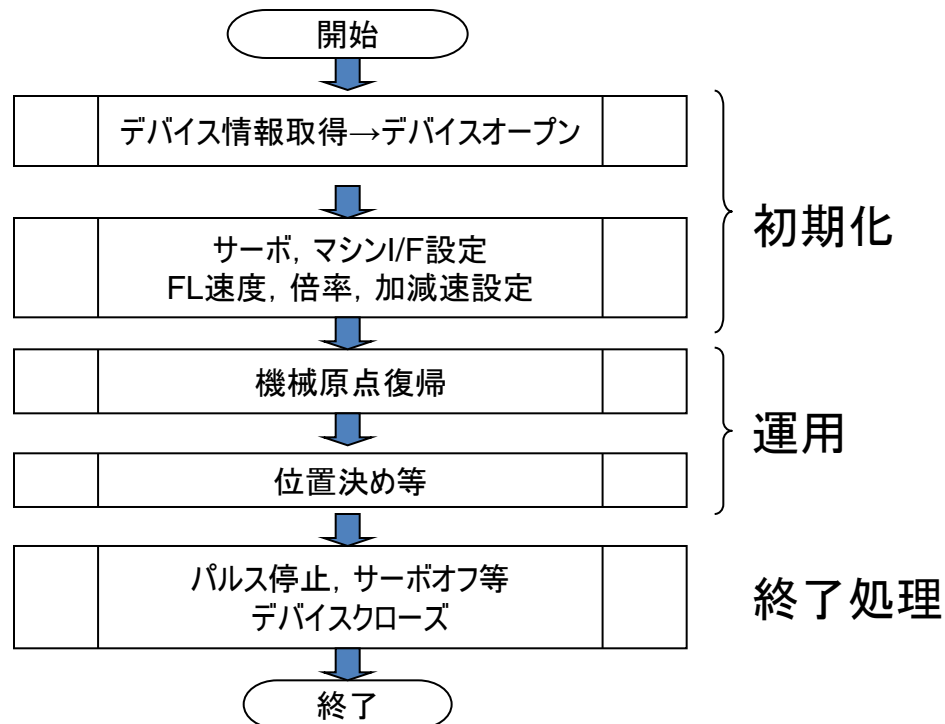
1. 用語
2. 運用概略
3. 初期化・終了処理手順
4. 動作手順
5. 動作の終了監視
6. ステータス
7. 環境レジスタ
8. 動作用レジスタ
9. 制御モード・動作モードバッファ
10. コマンド
11. 原点復帰完了条件
12. API関数
13. コード例

# 1.用語

No.	PPD	CPD	備考
1	送り量	移動量	位置決め移動量または目標位置
2	FL速度, R1速度	ベース速度	加減速を伴う動作(台形駆動など)の初速度
3	FH速度, R2速度	動作速度	位置決め, 補間, 原点復帰, 連続送り等の際の指示速度
4	倍率	速度倍率	速度倍率 = $300 / (\text{速度倍率レジスタ値} + 1)$ 速度(pps) = 速度レジスタ設定値 × 速度倍率
5	減速点	減速開始点	減速を開始する位置. 残移動量 = 減速開始点で減速を開始
6	CTRCL	SVCTRCL	サーボ偏差カウンタ(エラーカウンタ)クリア出力
7	プリセット動作	位置決め動作	
8	高速スタート	加速スタート	スタートコマンド書込みで加速を開始し停止時は減速する
9	浮動原点復帰	0点復帰	カウンタの0点に復帰する動作
10	手動パルス送り ハンドル手動送り	手動パルサ送り	パルサ入力に同期して動作するモード
11	デバイスID	デバイスハンドル	ボードを識別するための値

## 2.運用概略

PPD, CPDともに運用概略は同様です.



# 3.初期化・終了処理手順

手順はPPD,CPDともに同様です.

但し関数名, 関数仕様が若干異なります.

## ●初期化

(1) 使用する全ボードのデバイス情報の取得

(2) ボード毎にデバイスオープン

デバイスオープンに成功すると

→デバイスハンドル取得→以降, このハンドルでボードにアクセス

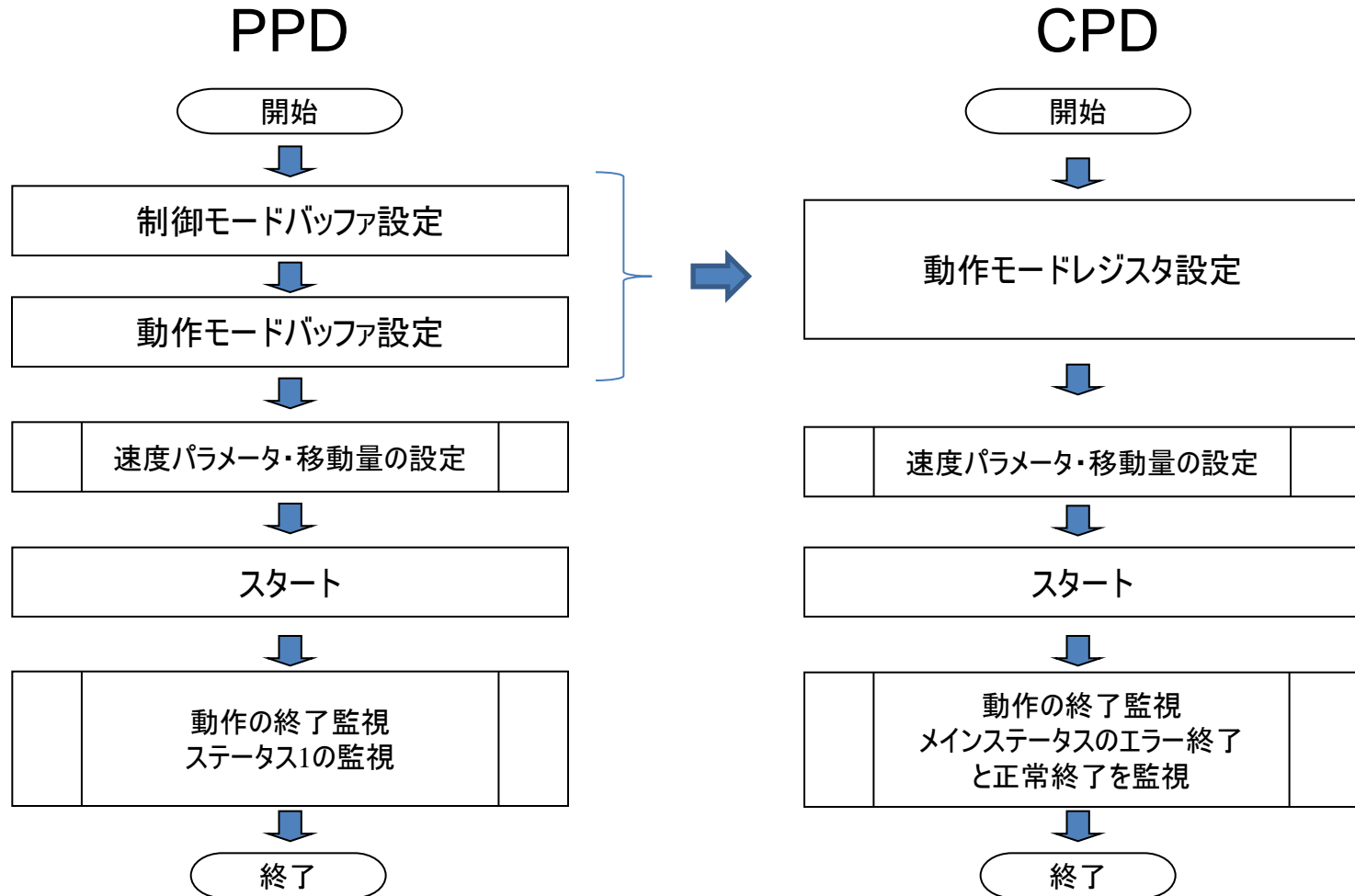
→ボードの信号処理方法の初期設定及び各軸の初期化

## ●終了処理

(1)パルス停止などのアプリケーション終了処理を行います.

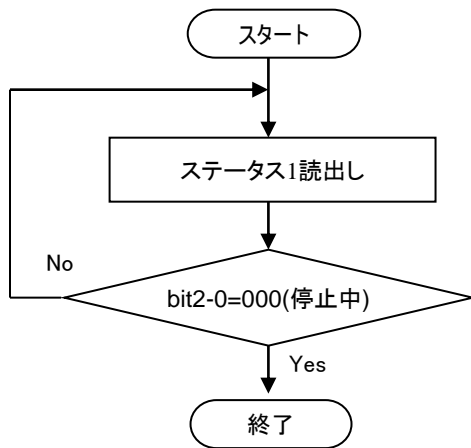
(2)ボード毎にデバイスクローズします.

# 4.動作手順



# 5.動作の終了監視

PPD

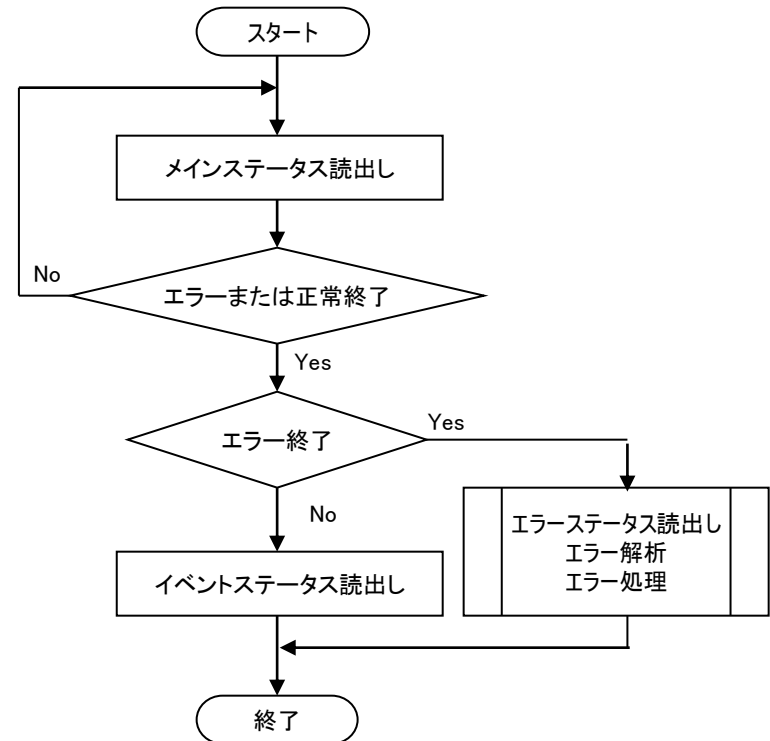


<備考>

●スタート書込みでステータス1が停止中から停止中以外に変化する時間は約350nsec

●割込みステータスを使用している場合は割込みステータスを読み出し、ステータス内容に応じた処理を行う。

CPD



# 6.ステータス

No.	内容	PPD	CPD
1	動作状態, マシンI/F	ステータス1	メインステータス(MSTS), サブステータス(SSTS), 拡張ステータス(RSTS)
2	コンパレータ結果, サーボI/F	ステータス2	メインステータス(MSTS), サブステータス(SSTS), 拡張ステータス(RSTS)
3	割込みステータス	割込みステータス	イベントステータス(RIST), エラーステータス(REST)
4	指令位置	UP/DOWNカウンタ(R9)	指令カウンタ(RCTR1)
5	機械位置	UP/DOWNカウンタ(R9)	機械カウンタ(RCTR2)
6	指令速度	カウンタモニタ(R12)	現在速度(RSPD)
7	Z相カウンタ	カウンタモニタ(R12)	現在速度(RSPD)
8	アイドル制御カウンタ	カウンタモニタ(R12)	現在速度(RSPD)
9	脱調検出カウンタ	カウンタモニタ(R12)	偏差カウンタ(RCTR3)
10	同時スタート/ストップ端子状態	コマンドモニタ1(R13)	拡張ステータス(RSTS)
11	動作モード	コマンドモニタ1(R13)	動作モードレジスタ(RMD)
12	制御モード	コマンドモニタ1(R13)	動作モードレジスタ(RMD)
13	スタート保留状態	コマンドモニタ2(R14)	拡張ステータス(RSTS)
14	Z相信号	コマンドモニタ2(R14)	拡張ステータス(RSTS)

# 6.ステータス

- ステータス1

bit	内容	CPD	備考
2-0	000: 停止中	MSTS.bit3=1	
	010: スタート保留中	RSTS.bit3-0=0010	
	101: 加速中	SSTS.bit8=1	
	110: 定速動作中	SSTS.bit10=1	
	111: 減速中	SSTS.bit9=1	
3	+ELS入力	SSTS.bit12	
4	-ELS入力	SSTS.bit13	
6,5	DLS入力	SSTS.bit15	
7	OLS入力	SSTS.bit14	

# 6.ステータス

- ステータス2

bit	内容	CPD	備考
0	コンパレート条件成立	MSTS.bit8(CMP1) MSTS.bit9(CMP2) MSTS.bit10(CMP3) MSTS.bit11(CMP4) MSTS.bit12(CMP5)	
1	SVALM入力	SSTS.bit11	
2	CTRCL出力	RSTS.bit9	
3	INPOS入力	RSTS.bit16	
4	不使用		
5	SVON出力	SSTS.bit0	
6	SVRST出力	SSTS.bit1	

# 6.ステータス

## ● 割込みステータス

値(HEX)	内容	CPD	備考
01	減速停止コマンド(0Ah)書込みによる停止	RIST.bit0	CPDは停止コマンド書込み後のイベントステータスで確認
02	位置決め動作完了による停止	RIST.bit0	CPDは位置決め動作スタート後のイベントステータスで確認
03	原点復帰(原点サーチ)動作完了による停止	RIST.bit0	CPDは原点復帰(サーチ)動作スタート後のイベントステータスで確認
04	原点抜け出し動作完了による停止	RIST.bit0	CPDは原点抜け出し動作スタート後のイベントステータスで確認
05	即停止コマンド(09h)書込みによる停止	RIST.bit0	CPDは停止コマンド書込み後のイベントステータスで確認
06	同時停止コマンドによる停止	REST.bit8	
07	-ELS信号ONによる停止	REST.bit6	
08	+ELS信号ONによる停止	REST.bit5	
09, 0A	DLS信号ONによる減速停止	REST.bit10	
0B	SVALM信号ONによる停止	REST.bit7	
0C	脱調検出による停止	REST.bit2	CPDでは偏差カウンタとRCMP3を比較し指令パルスを停止させる。
11	現在位置エンコーダ入力エラー	REST.bit16	
12	手動パルスエンコーダ入力エラー	REST.bit17	
13	脱調検出エンコーダ入力エラー	REST.bit16	
14	次動作スタート(プリレジスタ変更可)	RIST.bit1	
15	減速開始	RIST.bit6	
16	コンパレート条件が偽→真に変化	RIST.bit12-8	CPDではコンパレータが軸当たり5個

# 6.ステータス

- カウンタモニタ(R12)

bit	内 容	CPD	備 考
14-0	現在指令速度	RSPD.bit15-0	PPDでは停止時はFL設定値が読みだされる. CPDでは停止時"0"が読みだされる.
19-16	Z相カウンタ	RSPD.bit19-16	
22-20	アイドルパルス制御カウンタ	RSPD.bit22-20	
29-24	脱調検出カウンタ	RCTR3	

- コマンドモニタ1(R13)

bit	内 容	CPD	備 考
5-0	コマンド	なし	
6	同時スタート端子	RSTS.bit5	
7	同時ストップ端子	RSTS.bit6	
15-8	動作モード	RMD	
23-16	制御モード	RMD	

# 6.ステータス

## ● コマンドモニタ2(R14)

bit	内容	CPD	備考
5-0	プリコマンド	なし	
6	スタート保留状態	RSTS.bit3-0=0010	
7	Z相信号	RSTS.bit10	
15-8	プリ動作モード	PRMD	
16	PR0書込済み	MSTS.bit14	CPDではMSTS.bit14のみで動作プリレジスタを運用
17	PR1書込済み	MSTS.bit14	CPDではMSTS.bit14のみで動作プリレジスタを運用
18	PR2書込済み	MSTS.bit14	CPDではMSTS.bit14のみで動作プリレジスタを運用
19	PR3書込済み	MSTS.bit14	CPDではMSTS.bit14のみで動作プリレジスタを運用
20	PR4書込済み	MSTS.bit14	CPDではMSTS.bit14のみで動作プリレジスタを運用
21	プリ動作モード書込済み	MSTS.bit14	CPDではMSTS.bit14のみで動作プリレジスタを運用
22	プリレジスタ確定状態	RSTS.bit21,20	CPDはプリレジスタが2段ある。
23	スタートコマンド書込状態	MSTS.bit0	
24	PR5書込済み	MSTS.bit14	CPDではMSTS.bit14のみで動作プリレジスタを運用
25	PR15書込済み	MSTS.bit14	CPDではMSTS.bit14のみで動作プリレジスタを運用
26	PR16書込済み	MSTS.bit14	CPDではMSTS.bit14のみで動作プリレジスタを運用
31-27	不使用		

# 7.環境レジスタ

## PPD環境レジスタ1(R6)とCPDレジスタ

- bit15-0

bit	内容	CPD	備考
0	指令パルス出力	RENV1.bit0	
1	共通指令方向論理	RENV1.bit1	
2	指令パルス出力形式	RENV1.bit2	
3	INPOS	RENV1.bit22	
4	DLS	RENV1.bit6	
5	OLS	RENV1.bit7	
6	SVALM	RENV1.bit9	
7	Z相	RENV2.bit23	
8	不使用		
9	CTRCL自動出力	RENV1.bit11, RENV1.bit10	CPDでは自動出力条件を原点復帰完了時, 異常停止時と個別に設定可能
11,10	エンコーダ入力(A/B相)	RENV2.bit21,20	
13,12	パルス入力(A/B相)	RENV2.bit25,24	
15,14	脱調検出用エンコーダ入力	RENV2.bit21,20	CPDではエンコーダ入力と指令パルス出力の偏差カウンタを比較し脱調検出を行う。

# 7.環境レジスタ

## PPD環境レジスタ1(R6)とCPDレジスタ

- bit31-16

bit	内容	CPD	備考
23-16	不使用		
24	SVON出力設定	RENV2.bit1=0, RENV2.bit0=1	
25	SVRST出力設定	RENV2.bit3=0, RENV2.bit2=1	
26	不使用		
27	UP/DOWNカウンタ カウント入力選択	不要	CTR1:指令パルスカウント CTR2:エンコーダ入力カウント CTR3:偏差カウント
28	UP/DOWNカウンタ カウント数値形式	なし	CPDは符号付きカウントのみ
29	DLS検出時動作	RENV1.bit4	
30	OLS, DLSをラッチ	RENV1.bit5	CPDは原点復帰時OLSoft→onのエッジ検出 DLSはラッチ可能
31	不使用		

# 7.環境レジスタ

## PPD環境レジスタ2(R7)とCPDレジスタ

- bit31-0

bit	内容	CPD	備考
3-0	原点復帰時エンコーダZ相カウント設定	RENV3.bit7-4	
5,4	原点復帰条件	RENV3.bit3-0	
6	原点復帰時UP/DOWNカウンタクリア	RENV3.bit23-20	CPDは軸当たりカウンタ4個
7	マシンロック	PRMD.bit11	
9,8	不使用		
10	CTRCL出力後のスタートコマンド遅延有効	RENV1.bit17,16	CPDは遅延時間を4通り選択可能
11	共通指令出力形式時, 方向変化後パルス出力遅延有効	RENV1.bit28=0	CPDは0で有効, 1で無効
14-12	アイドルパルス	RENV5.bit10-8	
15	不使用		
19-16	コンパレータ条件	RENV4,RENV5	CPDは軸当たりコンパレータ5個
21,20	コンパレータ条件成立時動作	RENV4,RENV5	CPDは軸当たりコンパレータ5個
22	比較カウンタ選択	RENV4,RENV5	CPDは軸当たりコンパレータ5個
31-23	不使用		

CPDのコンパレータ設定はCPDシリーズ共通編マニュアルを参照

# 7.環境レジスタ

## PPD環境レジスタ3(R8)とCPDレジスタ

- bit15-0

bit	内容	CPD	備考
0	コマンド停止時報告	RIRQ.bit0	CPDは停止コマンド書込み後のイベントステータスで確認
1	プリセット動作完了時報告	RIRQ.bit0	CPDは位置決め動作スタート後のイベントステータスで確認
2	原点復帰完了時報告	RIRQ.bit0	CPDは原点復帰動作スタート後のイベントステータスで確認
3	減速開始時報告	RIRQ.bit6	
7-4	不使用		
12-8	脱調検出エンコーダ	RIRQ.bit10	CPDはCMP3を利用
13	コンパレータ条件成立時報告	RIRQ.bit12-8	CPDは軸当たりコンパレータ5個
14	次動作スタート時報告	RIRQ.bit1	
15	同時停止コマンドによる停止時報告	なし	PPD異常停止時の報告. CPD異常停止時はエラーステータスに変化. (エラーステータスはマスク不可)

# 7.環境レジスタ

## レジスタ名, レジスタ制御コマンド

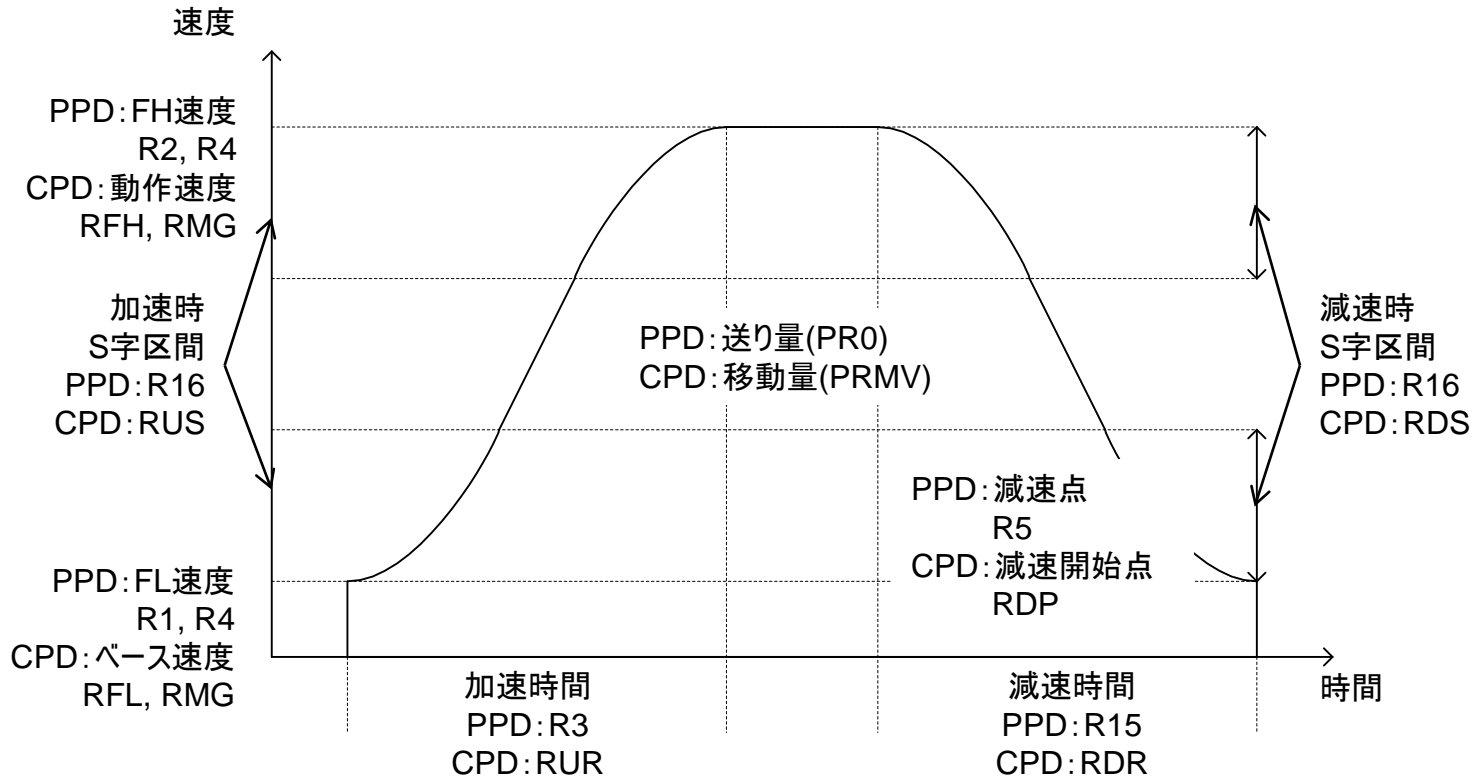
No.	内容	PPD レジスタ名	PPD 読出コマンド	PPD 書込コマンド	CPD レジスタ名	CPD 読出コマンド	CPD 書込コマンド
1	環境レジスタ1	R6	86h	C6h	RENV1, RENV2	DCh, DDh	9Ch, 9Dh
2	環境レジスタ2	R7	87h	C7h	RENV1,RENV3, RENV4,RENV5, PRMD	DCh,DEh, DFh,E0h, C7h	9Ch,9Eh, 9Fh,A0h, 87h
3	環境レジスタ3	R8	88h	C8h	RIRQ	ECh	Ach
4	アップ/ダウンカウンタ	R9	89h	C9h	RCTR1,RCTR2	E3h,E4h	A3h,A4h
5	コンパレータ1データ	R10	8Ah	CAh	RCMPx	E7h,E8h, E9h,EAh,EBh	A7h,A8h, A9h,AAh,ABh
6	コンパレータ2データ	R11	8Bh	CBh	RCMPx	E7h,E8h, E9h,Eah,EBh	
7	カウンタモニタ	R12	8Ch	---	RSPD	F5h	---
8	コマンドモニタ1	R13	8Dh	---	RMD	D7h	---
9	コマンドモニタ2	R14	8Eh	---	RSTS	F1h	---
10	プリセットカウンタ	PCTR	95h	---	RPLS	F4h	---
11	減速開始点カウンタ	SCTR	96h	---	RSDC	F6h	---

# 8.動作レジスタ

## レジスタ名, レジスタ制御コマンド

No.	内容	PPD レジスタ名	PPD 読出コマンド	PPD 書込コマンド	CPD レジスタ名	CPD 読出コマンド	CPD 書込コマンド	備考
1	送り量	R0	80h	---	RMV	D0h	90h	
2	FL(R1×速度倍率)速度	R1	81h	D1h	RFL	D1h	91h	
3	FH(R2×速度倍率)速度	R2	82h	D2h	RFH	D2h	92h	
4	加減速レート	R3	83h	D3h	RUR	D3h	93h	
5	速度倍率	R4	84h	D4h	RMG	D5h	95h	
6	減速点	R5	85h	D7h	RDP	D6h	96h	
7	減速レート	R15	9Ah	DAh	RDR	D4h	94h	
8	S字区間	R16	9Bh	DBh	RUS,RDS	D9h,DAh	99h,9Ah	
9	送り量	PR0	90h	C0h	PRMV	C0h	9Ah	プリレジスタ
10	FL(R1×速度倍率)速度	PR1	91h	C1h	PRFL	C1h	80h	プリレジスタ
11	FH(R2×速度倍率)速度	PR2	92h	C2h	PRFH	C2h	81h	プリレジスタ
12	加減速レート	PR3	93h	C3h	PRUR	C3h	82h	プリレジスタ
13	速度倍率	PR4	94h	C4h	PRMG	C5h	85h	プリレジスタ
14	減速開始点	PR5	97h	C5h	PRDP	C6h	86h	プリレジスタ
15	減速レート	PR15	98h	D8h	PRDR	C4h	84h	プリレジスタ
16	S字区間	PR16	99h	D9h	PRUS,PRDS	C9h,CAh	89h,8Ah	プリレジスタ

# 8.動作レジスタ



# 9.制御モード・動作モードバッファ

## ● 制御モードバッファ

bit	内容	CPD	備考
0	PCTR 動作 ON/OFF	なし	CPDは連続送り設定は動作モードのみで行う.
1	減速点検出方法	PRMD.bit13	
2	加減速方式	PRMD.bit10	
3	INPOSITION制御	PRMD.bit9	
4	次動作自動スタート制御	なし	CPDは動作中にスタートコマンドを書込むと次動作予約
5	同時停止信号入力	PRMD.bit24	
6	同時停止信号出力	PRMD.bit25	
7	三角駆動時速度修正機能	PRMD.bit26	

## ● 動作モードバッファ

bit	内容	CPD	
3-0	動作モード	PRMD.bit6-0	
4	送り方向	なし	CPDは動作モードに方向を含む.
5	DLS機能	PRMD.bit8	
6	UP/DOWNカウンタ動作	PRMD.bit11	
7	動作完了タイミング	PRMD.bit12	

# 9.制御モード・動作モードバッファ

## ● 動作モード

No.	内容	PPD(HEX)	CPD(HEX)	備考
1	連続モード1(手動送り)	0	00, 08	CPD 00: +連続送り, 08: -連続送り
2	連続モード2(ハンドル手動送り)	1	01	
3	原点復帰モード1	2	10, 18	CPD 10: +原点復帰, 18: -原点復帰
4	原点復帰モード2	3	なし	CPDではソフトリミットを利用
5	原点サーチ	4	15, 1D	CPD 15: +原点サーチ, 1D: -原点サーチ
6	原点抜け出し	5	12, 1A	CPD 12: +原点抜け出し, 1A: -原点抜け出し
7	1パルス出力	6	46, 4E	CPD 46: +1パルス出力, 4E: -1パルス出力
8	タイマモード (ドウェル)	7	47	
9	プリセットモード1 (方向と送り量)	8	なし	
10	プリセットモード2 (相対値送り)	9	41	
11	プリセットモード3 (絶対値送り)	A	42, 43	CPD 42: 指令位置, 43: 機械位置
12	浮動原点復帰モード	B	44, 45	CPD 44: 指令位置, 45: 機械位置
13	プリセットモード4 (ハンドル同期送り)	C	51	

# 10.コマンド

- スタートコマンド

No.	内容	PPD	CPD
1	FL定速スタート	10h	50h
2	FH定速スタート	11h	51h
3	高速スタート	13h	53h
4	残量FLスタート	14h	54h
5	残量FHスタート	15h	55h
6	残量高速スタート	17h	57h
7	同時スタート	30h	06h

- 速度レジスタ変更コマンド

No.	内容	PPD	CPD
1	瞬時にR1速度に変更	00h	40h
2	瞬時にR2速度に変更	01h	41h
3	減速してR1速度に変更	03h	43h
4	加速してR2速度に変更	04h	44h

# 10.コマンド

- 停止コマンド

No.	内容	PPD	CPD
1	即停止	09h	49h
2	減速停止	0Ah	4Ah
3	同時停止	28h	07h

- スタート保留コマンド

No.	内容	PPD	CPD
1	FL定速スタート保留	20h	CPDではPRMD.bit19=0, bit18=1にすることで同時スタート待ちになる。
2	FH定速スタート保留	21h	
3	高速スタート保留	23h	
4	残量FL定速スタート保留	24h	
5	残量FH定速スタート保留	25h	
6	残量高速スタート保留	27h	

# 10.コマンド

- コントロールコマンド

No.	内容	PPD	CPD	備考
1	ソフトウェアリセット	60h	04h	
2	UP/DOWNカウンタリセット	61h	20h, 21h	CPDでは20h: 指令カウンタ, 21h: 機械カウンタ
3	脱調検出用偏差カウンタリセット	62h	22h	
4	非常停止	63h	05h	
5	プリレジスタ確定セット	66h	4Fh	
6	プリレジスタ確定リセット	67h	26h	
7	SVON ON	58h	18h	
8	SVON OFF	48h	10h	
9	SVRESET ON	59h	19h	
10	SVRESET OFF	49h	11h	

# 11.原点復帰完了条件

No.	R2.bit5, 4	内容	CPD
1	00	OLS入力ONで完了	RENV3.bit3-0 = 0000
2	01	OLS入力ON後のZ相カウントで完了 通常は定速またはDLSを使用する.	RENV3.bit3-0 = 0010 但し加速スタート時はOLS入力ONで減速開始
3	10	OLS入力ONで減速開始, Z相カウントで完了	RENV3.bit3-0 = 0010

## <PPDとCPDの原点復帰動作の違い>

- PPDではOLS入力ON状態で原点復帰を行うと即完了.
- CPDではOLS入力OFFからONのエッジ検出の為, OLS入力ON状態で原点復帰を行うとELSまで動作.  
(CPDでは原点サーチ動作を推奨)

## <ステータスの違い>

- PPDでは割り込みステータスにより原点復帰完了か原点復帰を停止コマンドにより停止したかの判別ができる.
- CPDでは原点復帰完了時もコマンドによる停止時もイベントステータス上は正常停止となるのでアプリケーションで判断する必要有. (エラー停止時はエラーステータスで確認可能)

# 12.API関数比較

## 関数名

No.	機能	PPD関数名	CPD関数名
1	ボード枚数の取得	hppd530_GetDeviceCount	cp530_GetDeviceCount
2	デバイス情報の取得	hppd530_GetDeviceInfo	cp530_GetDeviceInfo
3	デバイスのオープン	hppd530_OpenDeviceID hppd530_OpenDevice	cp530_OpenDevice
4	デバイスのクローズ	hppd530_CloseDevice	cp530_CloseDevice
5	コマンドバッファ書込	hppd530_CmdWrite	cp530_wCmdW
6	(レジスタ書込)	hppd530_CmdWrite	cp530_wReg
7	レジスタ読込	hppd530_regRead	cp530_rReg
8	ELS極性選択ポート読込	hppd530_ElstRead	cp530_rPortB cp530_rPortW
9	ELS極性選択ポート書込	hppd530_ElstWrite	cp530_wPortB cp530_wPortW
10	コンパレータ同時スタート読込	hppd530_CmpcRead	cp530_rPortB cp530_rPortW
11	コンパレータ同時スタート書込	hppd530_CmpcWrite	cp530_wPortB cp530_wPortW

# 12.API関数比較 仕様

No.	関数名	戻り値	引数1	引数2
1	hppd530_GetDeviceCount	ボード枚数	dummy	---
2	cp530_GetDeviceCount	0:正常, 0以外:エラー	ボード枚数	---
3	hppd530_GetDeviceInfo	1:正常, 0:エラー	ボード枚数	デバイス情報
4	cp530_GetDeviceInfo	0:正常, 0以外:エラー	ボード枚数	デバイス情報
5	hppd530_OpenDeviceID hppd530_OpenDevice	デバイスハンドル	デバイス情報	---
6	cp530_OpenDevice	0:正常, 0以外:エラー	デバイスハンドル	デバイス情報
7	hppd530_CloseDevice	1:正常, 0:エラー	デバイスハンドル	---
8	cp530_CloseDevice	0:正常, 0以外:エラー	デバイスハンドル	---

# 12.API関数比較 仕様

No.	関数名	戻り値	引数1	引数2	引数3	引数4	引数5
1	hppd530_CmdWrite	1: 正常, 0: エラー	ハンドル	軸指定	コマンドデータ	0	0
2	cp530_wCmdW	0: 正常, 0以外: エラー	ハンドル	軸指定	コマンドデータ	---	---
3	hppd530_CmdWrite	1: 正常, 0: エラー	ハンドル	軸指定	書込コマンド	1	レジスタデータ
4	cp530_wReg	0: 正常, 0以外: エラー	ハンドル	軸指定	書込コマンド	レジスタデータ	---
5	hppd530_regRead	1: 正常, 0: エラー	ハンドル	軸指定	読出コマンド	レジスタデータ	---
6	cp530_rReg	0: 正常, 0以外: エラー	ハンドル	軸指定	読出コマンド	レジスタデータ	---
7	hppd530_ElstRead	1: 正常, 0: エラー	ハンドル	極性データ	---	---	---
8	cp530_rPortB, cp530_rPortW	0: 正常, 0以外: エラー	ハンドル	読出コマンド	極性データ	---	---
9	hppd530_ElstWrite	1: 正常, 0: エラー	ハンドル	極性データ	---	---	---
10	cp530_wPortB, cp530_wPortW	0: 正常, 0以外: エラー	ハンドル	書込コマンド	極性データ	---	---

# 13.コード例

## デバイスオープン/クローズ

### PPD

```
HPCDEVICEINFO  HpciInfo[16];
DWORD          dwNum;
DWORD          hDev;

// ボード枚数取得
dwNum = hppd530_GetDeviceCount(1);

// デバイス情報取得
if (!hppd530_GetDeviceInfo(&dwNum, &HpciInfo[0])) {
    // エラー
}

// デバイスオープン
hDev = hppd530_OpenDeviceID(&HpciInfo[0]);
if (0xffffffff == hDev) {
    // 戻り値がffffffffhの時はエラー
}

// 処理 .....

// デバイスクローズ
if (!hppd530_CloseDevice(hDev)) {
    // エラー
}
```

### CPD

```
HPCDEVICEINFO  HpciInfo[16];
DWORD          dwNum;
DWORD          dwRet;
DWORD          hDev;

// ボード枚数取得
dwRet = cp530_GetDeviceCount(&dwNum);
if(dwRet) {
    //エラー
}

// デバイス情報取得
dwRet = cp530_GetDeviceInfo(&dwNum, &HpciInfo[0]);
if(dwRet) {
    //エラー
}

// デバイスオープン
dwRet = cp530_OpenDevice(&hDev, &HpciInfo[0]);
if(dwRet) {
    //エラー
}

// 処理 .....

// デバイスクローズ
dwRet = cp530_CloseDevice(hDev);
```

# 13.コード例

## 軸の初期化

### PPD

```
// ELS入力極性(全軸B接)
hppd530_ElStWrite(hDev, 0x00);

// X軸初期化
// コマンドバッファ書込(ソフトウェアリセット)
hppd530_CmdWrite(hDev, 0, 0x60, 0, 0);

// レジスタ書込(R1速度レジスタ:PR1 = 200)
hppd530_CmdWrite(hDev, 0, 0xc1, 1, 200);

// レジスタ書込(R2速度レジスタ:PR2 = 2000)
hppd530_CmdWrite(hDev, 0, 0xc2, 1, 2000);

// レジスタ書込(加減速レート:PR3)
// 基準クロック周波数= 19,660,800Hz, 加速時間= 減速時間= 0.5秒
// PR2 = 2000, PR1 = 200, PR3:加速レート
// 加速時間(秒) = (PR2 - PR1) * (PR3 + 1) * 4 / 19660800
// PR3 = 1364
hppd530_CmdWrite(hDev, 0, 0xc3, 1, 1364);

// レジスタ書込(倍率=1倍, PR4=299)
Hppd530_CmdWrite(hDev, 0, 0xc4, 1, 299);

// 以降R6, R7, R8等初期化
```

### CPD

```
// ELS入力極性(全軸B接)
cp530_wPortB(hDev, 0x80, 0x00);

// X-U軸ソフトウェアリセット
cp530_wCmdW(hDev, 0, 0x04);

// X軸の初期化
// レジスタ書込(ベース速度レジスタ:PRFL = 200)
cp530_wReg (hDev, 0, 0x81, 200);

// レジスタ書込(動作速度レジスタ:PRFH = 2000)
cp530_wReg (hDev, 0, 0x82, 2000);

// レジスタ書込(直線加速時加速レートレジスタ)
// 基準クロック周波数= 19,660,800Hz, 加速時間= 減速時間= 0.5秒
// PRFH = 2000, PRFL = 200, PRUR:加速レート
// 加速時間(秒) = (PRFH - PRFL) * (PRUR + 1) * 4 / 19660800
// PRUR = 1364
cp530_wReg (hDev, 0, 0x83, 1364);

// レジスタ書込(速度倍率= 1倍, PRMG = 299)
cp530_wReg (hDev, 0, 0x85, 299);

// 以降RENV1, RENV2, REV3等初期化
```

# 13.コード例

## 連続送り

### PPD

```
BYTE sts1;  
  
// 停止中を確認  
// ステータスのbit2-0=000ならば停止中  
hppd530_status1Read(hDev, 0, &sts1);  
if(0x00 == (sts1 & 0x07)) {  
    // FH速度:PR2 = 5000  
    hppd530_CmdWrite(hDev, 0, 0xc2, 1, 5000);  
  
    // 動作モードバッファ書込(+連続送り)  
    hppd530_DousaWrite(hDev, 0, 0x20);  
  
    // コマンドバッファ書込(高速スタート)  
    hppd530_CmdWrite(hDev, 0, 0x13, 0, 0);  
}
```

### CPD

```
WORD msts;  
  
// 停止中を確認  
// メインステータスのbit3=1ならば停止中  
cp530_rMstsW(hDev, 0, &msts);  
if(0x0008 == (msts & 0x0008)) {  
    // 動作速度レジスタ:PRFH = 5000  
    cp530_wReg(hDev, 0, 0x82, 5000);  
  
    // 動作モードレジスタ:PRMD = 0x00000000(+連続送り)  
    cp530_wReg(hDev, 0, 0x87, 0x00000000);  
  
    // 加速スタート  
    cp530_wCmdW(hDev, 0, 0x53);  
}
```

# 13.コード例

## 動作完了待ち

### PPD

```
BYTE sts1;

while(1) {
    // 停止中を確認
    // ステータスのbit2-0=000ならば停止中
    hppd530_status1Read(hDev, 0, &sts1);
    if(0x00 == (sts1 & 0x07)) {
        break;
    }
}
```

### CPD

```
// 軸の初期化時にRIRQ=0x00000001(正常停止報告)の設定をしておく
WORD msts;
DWORD rest;
DWORD rist;
while(1) {
    // メインステータスのbit4=1ならばエラー発生
    // bit5=1ならばイベント報告有(この場合正常停止)
    cp530_rMstsW(hDev, 0, &msts);
    if(msts & 0x0030) {
        if(0x0010==(msts & 0x0010)) {
            // エラー発生
            // エラーステータス(REST)読出→MSTS.bit4がリセットされる
            // RESTもリセットされる
            cp530_rReg(hDev, 0, 0xf2, &rest);
            // エラー内容に応じた処理
            break;
        }
        if(0x0020==(msts & 0x0020)) {
            // イベント報告有
            // イベントステータス(RIST)読出→MSTS.bit5がリセットされる
            // RISTもリセットされる
            cp530_rReg(hDev, 0, 0xf3, &rist);
            if(0x00000001==(rist & 0x00000001)) {
                break; // 正常停止
            }
        }
    }
}
```

# 13.コード例

## 原点サーチ

### PPD

```
// 軸の初期化時に原点復帰完了条件を設定しておく
BYTE sts1;

// 停止中を確認
// ステータスのbit2-0=000ならば停止中
hppd530_status1Read(hDev, 0, &sts1);
if(0x00 == (sts1 & 0x07)) {
    // 送り量:PR0 = 5000
    // 原点抜け出し量
    hppd530_CmdWrite(hDev, 0, 0xc0, 1, 5000);

    // FH速度:PR2 = 2000
    hppd530_CmdWrite(hDev, 0, 0xc2, 1, 2000);

    // 動作モードバッファ書込(-原点サーチ)
    hppd530_DousaWrite(hDev, 0, 0x14);

    // コマンドバッファ書込(高速スタート)
    hppd530_CmdWrite(hDev, 0, 0x13, 0, 0);

    // 原点復帰の完了待ち
}
```

※原点抜け出し量は符号なしの値を設定

### CPD

```
// 軸の初期化時に原点復帰モード(原点復帰のパターン)を設定しておく
WORD msts;

// 停止中を確認
// メインステータスのbit3=1ならば停止中
cp530_rMstsW(hDev, 0, &msts);
if(0x0008 == (msts & 0x0008)) {
    // 移動量レジスタ:PRMV = 5000
    // 原点サーチ抜け出し量を設定する
    cp530_wReg(hDev, 0, 0x80, 5000);

    // 動作速度レジスタ:PRFH = 2000
    cp530_wReg(hDev, 0, 0x82, 2000);

    // 動作モードレジスタ:PRMD = 0x0000001D(-原点サーチ)
    cp530_wReg(hDev, 0, 0x87, 0x0000001D);

    // 加速スタート
    cp530_wCmdW(hDev, 0, 0x53);

    // 原点復帰の完了待ち
}
```

※原点抜け出し量は符号なしの値を設定

# 13.コード例

## プリセットモード1位置決め

### PPD

```
BYTE sts1;

// 停止中を確認
// ステータスのbit2-0=000ならば停止中
hppd530_status1Read(hDev, 0, &sts1);
if(0x00 == (sts1 & 0x07)) {
    // 送り量:PR0 = 50000
    hppd530_CmdWrite(hDev, 0, 0xc0, 1, 50000);

    // FH速度:PR2 = 10000
    hppd530_CmdWrite(hDev, 0, 0xc2, 1, 10000);

    // 制御モードバッファ書込(INPOSITION制御)
    hppd530_SeigyoWrite(hDev, 0, 0x08);

    // 動作モードバッファ書込(プリセットモード1位置決め)
    hppd530_DousaWrite(hDev, 0, 0x08);

    // コマンドバッファ書込(高速スタート)
    hppd530_CmdWrite(hDev, 0, 0x13, 0, 0);

    // 位置決め完了待ち
}
}
```

### CPD

```
WORD msts;

// 停止中を確認
// メインステータスのbit3=1ならば停止中
cp530_rMstsW(hDev, 0, &msts);
if(0x0008 == (msts & 0x0008)) {
    // 移動量レジスタ:PRMV = 50000
    cp530_wReg(hDev, 0, 0x80, 50000);

    // 動作速度レジスタ:PRFH = 10000
    cp530_wReg(hDev, 0, 0x82, 10000);

    // 動作モードレジスタ:PRMD = 0x00000241
    // 相対位置決め(0x41) INPOS制御ON(bit9=1)
    cp530_wReg(hDev, 0, 0x87, 0x00000241);

    // 加速スタート
    cp530_wCmdW(hDev, 0, 0x53);

    // 位置決め完了待ち
}
}

※CPDの相対位置決めの動作方向は移動量の符号により決定
```

# 13.コード例

## プリセットモード3位置決め

### PPD

```
BYTE sts1;

// 停止中を確認
// ステータスのbit2-0=000ならば停止中
hppd530_status1Read(hDev, 0, &sts1);
if(0x00 == (sts1 & 0x07)) {
    // 目標位置:PR0 = 10000
    hppd530_CmdWrite(hDev, 0, 0xc0, 1, 10000);

    // FH速度:PR2 = 10000
    hppd530_CmdWrite(hDev, 0, 0xc2, 1, 10000);

    // 制御モードバッファ書込(INPOSITION制御)
    hppd530_SeigyoWrite(hDev, 0, 0x08);

    // 動作モードバッファ書込(プリセットモード3位置決め)
    hppd530_DousaWrite(hDev, 0, 0x0A);

    // コマンドバッファ書込(高速スタート)
    hppd530_CmdWrite(hDev, 0, 0x13, 0, 0);

    // 位置決め完了待ち
}
}
```

### CPD

```
WORD msts;

// 停止中を確認
// メインステータスのbit3=1ならば停止中
cp530_rMstsW(hDev, 0, &msts);
if(0x0008 == (msts & 0x0008)) {
    // 目標位置:PRMV = 10000
    cp530_wReg(hDev, 0, 0x80, 10000);

    // 動作速度レジスタ:PRFH = 10000
    cp530_wReg(hDev, 0, 0x82, 10000);

    // 動作モードレジスタ:PRMD = 0x00000242
    // 指令位置座標指定位置決め(0x42) INPOS制御ON(bit9=1)
    cp530_wReg(hDev, 0, 0x87, 0x00000242);

    // 加速スタート
    cp530_wCmdW(hDev, 0, 0x53);

    // 位置決め完了待ち
}

※この動作モードはCPDシリーズマニュアルに記載されておりません。
この動作モードはPRMV(目標位置)レジスタ設定値とRCTR1との差分だけ動作するモードです。
RCTR1の値はスタート時に記憶されるので、RMV変更による目標のオーバーライドはできますが、RCTR1の変更によるオーバーライドはできません。
(動作中のRCTR1の書き換えは禁止)
```