IPR-430RTOSのご紹介

IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430
430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS

2012年1月27日

株式会社インフォメーションプロセシングリサーチ

IPR

株式会社　インフォメーションプロセシングリサーチ

IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430
430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS

1981年7月　　神奈川県川崎市　創業

1982年6月　　世田谷区世田谷　移転

1984年8月　　世田谷区野沢　会社設立　資本金　350万

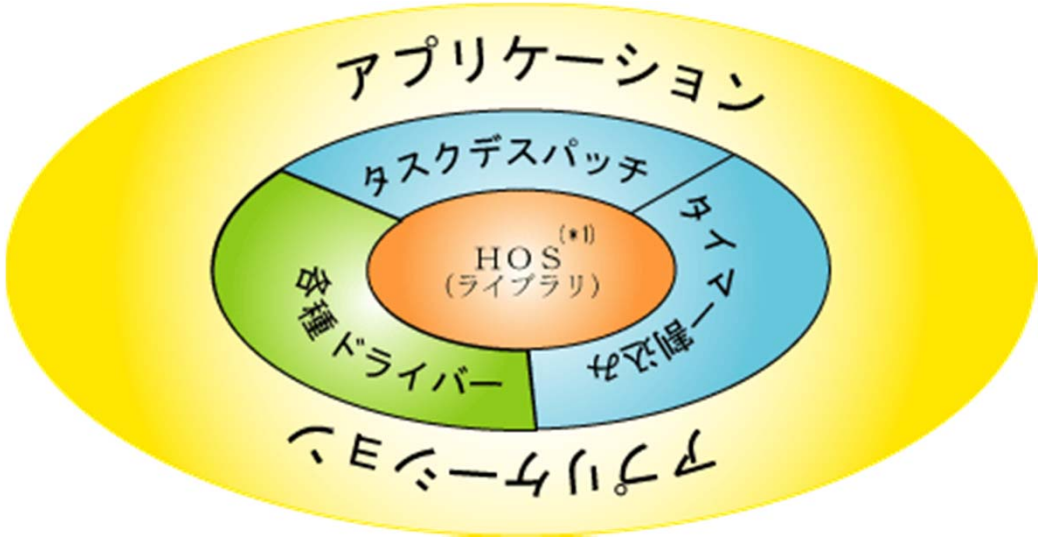
1990年7月　　資本金　1,000万に増資

2005年12月　吉祥寺に移転

マイクロコンピュータ　4004のソフト・ハード開発をスタートに、4040・8008・8080・z80・6800等を実施、8ビット・16ビットのリアルタイムOSを開発。
物流システムの受託開発では、自動倉庫・ソーター・WMSパッケージ等の開発。
2009年にMSP430のリアルタイムOS　IPR-430RTOSを販売開始。

IPR-430RTOS の概要

IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430
430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS



超低消費電力の16ビットCPU MSP430用のリアルタイムOSです。

IPR-430RTOSは、MSP430Xアーキテクチャを前提とした iTRON4.0仕様に基づいたリアルタイムのOSで、RTOSを使用しても低消費電力をサポートします。

HOSのカーネルはC言語で作成されていますが、機種に依存する部分はアセンブラで記述されています。

機能

基本的な機能は左記のように、
μiTRON4.0に準拠します。
HOS以外では、下記があります。

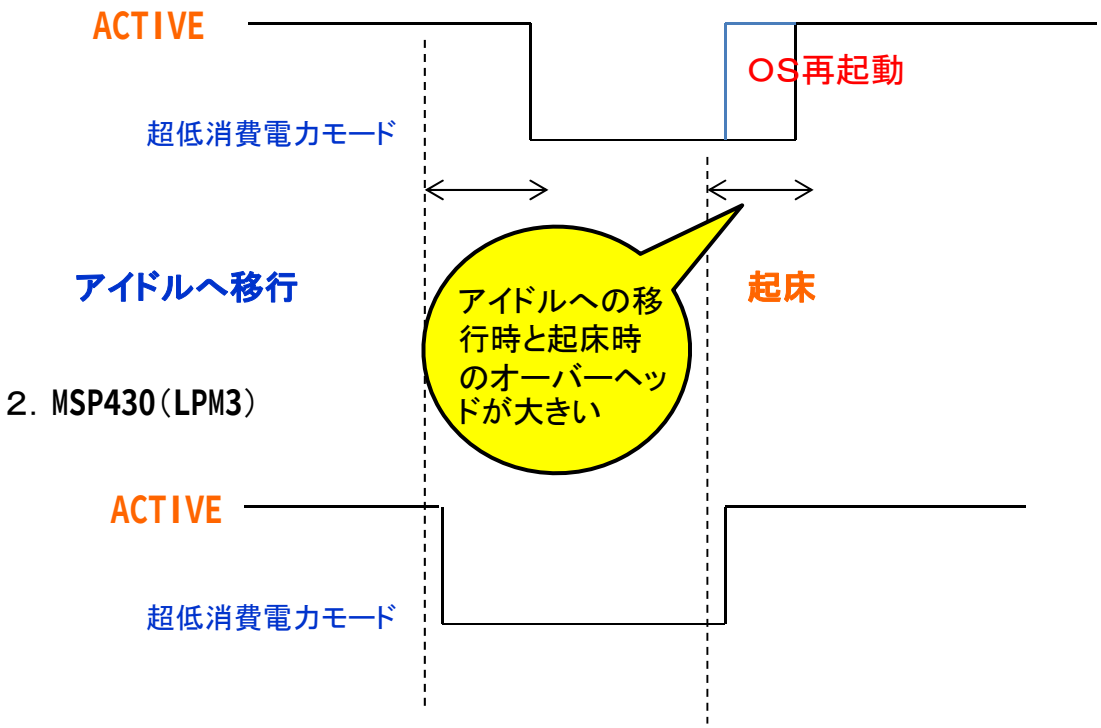
タスクディスパッチ部・スタートアップルーチン部・ベクターアドレス部・割り込み処理部・OSタイマー部。これらは、必要に応じて変更出来るようにサンプルプログラムが提供されます。

特徴

●MSP430の特徴を最大限生かしたRTOS

■ MPUのモードとRTOSの関係

1．他社製MPU（シャットダウンモード、ソフトウエアスタンバイモード）



例えば他社のMPUの場合、超消費電力を実現させるモードにしようした場合、SRAMのデータが消滅してしまうようです。

ここでRTOSを採用した場合は、リセットと同様のイニシャライズから開始することになります。

その点、MSP430のLPM3を使用すれば、下図のように違和感なく通常のタスクの起床と同様の処理が開始されます。

特徴

●μiTRON4.0仕様に準拠

μiTRON4.0仕様に準拠したHOS(Hyper Operating System)をMSP430用にポーティングし全機能を提供しています。

●RTOSを使用しての低消費電力化

アイドルタスクに省電力モードを実装し、省電力化を実現。割り込み処理では、所定の低速クロックに切り替え、終了時にLPM3モードからアクティブモードに切り替えることで、高速クロックに切り替わり、タスクがディスパッチされます。

特徴

省電力化のマクロを提供

IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430
430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS

IAR Embedded Workbench IDE

File Edit View Project Simulator Tools Window Help

Workspace: Debug

Files: SAMPLE - ... / ASM / C / H / IPR-ITRON / Output

mrmv_que.c | set_flg.c | exe_int.c | Lowinit.c | pacint.asm | OPTION.C | OsTimer.c | midl_lop.c | pacctx.asm | lpw_ref_sem.c | wai_sem.c | dly_tsk.c | taskmain.c | mexe_dsp.c | msrh_top.c | ini_tsk.c | Main.c

```
/* サンプルタスク */
void    Task1( VP_INT exinf )
{
        MESSAGE     *Msg;
        FLGPTN      p_flgptn;
        ER          ErCode;

        RUN                             // 高速クロック維持
                                        // リアルタイムクロックON

        DI
        OsTimer_Start();                // 1分割り込み開始

        RTCCTL1     =   RTCBCD_H | RTCMODE_H;
        RTCCTL0     =   RTCTEVIE;
        EI


        while( FOREVER ){
                                        // 1分割り込み待ち
            SLEEP                       // 高速クロック解放
            wai_flg( EVENT_FLG1, 0x01, TWF_ORW, &p_flgptn );
            RUN                         // 高速クロック維持
            clr_flg(  EVENT_FLG1, ~ 0x01 );
            ErCode  = get_mpf( MPF64, ( VP ) &Msg );
            if ( ErCode == E_OK ){
                memset( Msg -> buff, 0,sizeof( Msg -> buff ) );
                strcpy( Msg -> buff, "1分割り込みイベント" );
                snd_mbx( MBX_1, ( T_MSG * )Msg );
            }
            set_flg(  EVENT_FLG2, 0x01 );
        }

}
void    Task2( VP_INT exinf )
{
        MESSAGE     *Msg;
        char        Tmp[ 64 ];
```

Ready | Ln 31, Col 21

【よくある質問】
イベントやセマフォ、メッセージ等のオーソドックスな関数だけなら約5Kバイト。
アプリケーションはC言語なので使用するAUTO変数にもよるが256バイト以上は必要。

特徴 割込み時のオーバーヘッド

IPR-430RTOS IPR-430
430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS

```
;-----------------------------------------------
; 割り込みハンドラ
;-----------------------------------------------
_hos_vector041:                    // RTC
        ADD     &RTCIV_,PC
                                                ;
        RETI                       ;
        JMP     _hos_vector041_1   ; Ready
        PUSHX.A R12
        MOV.W   #41, R12           ; 割り込みIDセット
        BIC.B   #4,&RTCCTL0        ; 1分
        JMP     int_handler
_hos_vector041_1:
        PUSHX.A     R12
        MOV.W   #76, R12           ; 割り込みIDセット
        BIC.B   #0x10,&RTCCTL0
        JMP     int_handler
```

```
int_handler:
; ---- レジスタ保存
        PUSHX.A     R13
        PUSHX.A     R14
        PUSHX.A     R15

        MOV.W   UCSCTL6,R13          ;
        BIT.W   #XT2OFF,R13          ; HIGH CLOCK ON?
        JZ      int_handler_skip     ; THEN
        CALLA   #HighClockOnProc     ;
int_handler_skip:
; -------- 多重割り込みの判別
        MOV.W   kernel_int_cnt,R13
        BIT.W   R13,R13
        JNE     mul_int              ; 多重割り込みへ

; ======== 最初の割り込み ========
; ---- 割り込みネストカウンタ設定
        MOV.W   #1, R13
        MOV.W   R13,kernel_int_cnt

; ---- スタックポインタの入れ替え
        MOVX.W  SP, kernel_int_ssp
        MOVX.W  kernel_int_sp,SP     ; SPをOSレベルへ移行

; ---- 割り込み処理の開始
        PUSHX.A R12
        CALLA   #kernel_sta_int      ; ステータス変更
        POPX.A  R12
```

クロックドライバー

特徴

IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430
430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS

- **ライセンスフリー**
  ライセンスは、製品単位・開発部署単位となり、製品の台数制限はありません。
- **オープンソースでのご提供**
  ソースコードを添付して納品致します。
- **サンプルドライバーの提供**
  低消費電力を実現するために、サンプルドライバーを提供しています。これらを割込み処理で呼び出すことでクロックの切替が実現出来ます。
- **サポートの充実**
  無償サポート期間として10インシデント(1年間限定)のサービスを提供しています。

価格・納品形態

IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430
430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS IPR-430RTOS

● 価格（希望小売価格）

IPR-430RTOS-IAR　¥350,000-/ライセンス

IPR-430RTOS-CCR　¥550,000-/ライセンス

●納品形態

使用ライセンス契約締結（製品単位・部署単位）後の納品となります。

納品物は　CD-ROM　1枚（マニュアルは電子ファイルとして収納）・マニュアルをコピーしたものを添付となります。

価格は代理店又は当社へお問合せ下さい。