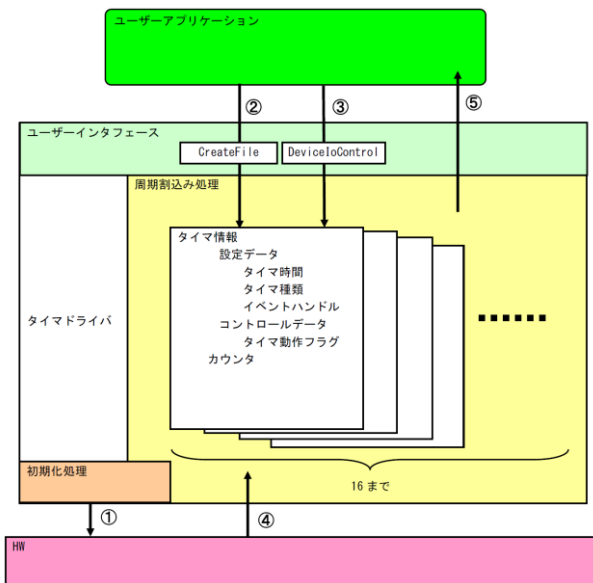


## ● タイマ割り込みとは

- ・ソフトウェアで一定周期に動作を行わせる機能です。ユーザーアプリケーションから直接制御するためのドライバを提供しています。
  - ・リアルタイムOSを使うほどシビアな定期処理でない場合に最適。温度センサや定期的な読取りなどに利用できます。
    - ・Windowsの場合・・・10msecごとに設定
    - ・Linuxの場合・・・・・・10msecから1msec～65535msec間隔で設定できます。
- ※ 正確な一定周期処理/リアルタイム処理が必要な場合は、リアルタイムOS INtimeを搭載しているオールインワンコントローラをお勧めします。

## ● タイマドライバの操作

- ① 起動時に10msec(レジストリ設定で変更可能)の周期割り込み設定を行います。
- ② オープンされたデバイスハンドル毎に、タイマ情報を作成しタイマ情報テーブルへ追加します。オープンできるハンドルはシステム全体で16までとなります。タイマ情報テーブルへの追加はオープンした順番で追加されます。
- ③ ユーザーアプリケーションからの設定をタイマ情報テーブルへ反映させます。
- ④ 周期割り込みが発生したらタイマ情報テーブルを参照し、各タイマ情報のカウント値を加算します。
- ⑤ カウント値が設定値に達したものは、イベントハンドルでタイマ通知を行います。カウント加算、イベント通知処理はタイマ情報テーブルの順番で処理されます。



## ● APIやサンプルプログラムも用意しています

```

IOCTL_FPGATIMER_START
機能
タイマ処理を開始します。
パラメータ
IOCTL_FPGATIMER_SETCONFIG
機能
タイマの設定を行います。
パラメータ
FPGATIMER_CONFIGを接続するためのポインタ
戻り値
処理が完了したときに返される値
説明
タイマ割り込み制御サンプルソース
#include <windows.h>
#include <winioctl.h>
#include <stdio.h>
#include <stdlib.h>
#include <mmsystem.h>
#include <conio.h>
#include ".\Common\FpgaTimerDD.h"
#define TIMERDRIVER_FILENAME "YYYY.YYFpgaTimer"
#define MAX_TIMEREVENT 10
typedef struct {
    int No;
    HANDLE hEvent;
    HANDLE hThread;
    volatile BOOL fStart;
    volatile BOOL fFinish;
    HANDLE hTimer;
    FPGATIMER_CONFIG Config;
} TIMEREVENT_INFO, *PTIMEREVENT_INFO;
/*
 * 割り込みハンドラ
 */
DWORD WINAPI TimerEventProc(void *pData)
{
    PTIMEREVENT_INFO info = (PTIMEREVENT_INFO)pData;
    DWORD ret;
    printf("TimerEventProc: Timer%02d: Start\n", info->No);
    info->fFinish = FALSE;
    while(1){
        if(WaitForSingleObject(info->hEvent, INFINITE) != WAIT_OBJECT_0){
            break;
        }
        if(!info->fStart){
            break;
        }
    }
}
    
```

※詳細については、マニュアルをご参照ください。

## ● 対象製品

7A IoTシリーズ	NP7A・EC7A
4A IoTシリーズ	AP4A・APS4A・EC4A・AS4A
4B IoTシリーズ	AP4B・APS4B・EC4B
4A UPSシリーズ	AP4A・APS4A・APL4A・EC4A・AS4A
1A IoTシリーズ	APS1A・EC1A・AS1A

このカタログに記載された製品は、予告なしに仕様・機能・デザイン等を変更する場合がありますので、ご採用の際には最新の情報を弊社及び弊社製品取扱販売店までお問い合わせください。掲載している会社名・製品名・ネットワーク名称等はそれぞれ各社・各団体の商標または登録商標です。 3110\_2022年1月a版