

以下に各関数の仕様・詳細を記します。

【1】初期化

書式	int ad_open_adsys (int address, int intr_no)
引数	address : ADボードのベースアドレス / 標準設定 : 0x01d0 (1-3項参照) intr_no : 割り込み番号 / 選択肢 : 3, 5, 7, 9, 10, 11, 12, 15 (3-6項参照)
戻り値	正常終了時 : ボードのID / ADM-652AT : 52H ADM-656AT : 56H エラー時 : エラーコード / 負の値 (エラーコード表)
機能・動作	ADボード上で設定したI/Oアドレスを本ハンドラが認識すると共に、 ADボードのリセット、ハンドラ内部の参照テーブルやデータバッファを初期化する。 また、ADボード上のFIFOメモリ容量をチェックする。

【2】サンプリング実行チャンネル関連の設定

書式	int ad_set_sampch (int no_ch, int scan_order[], int range[])
引数	no_ch : サンプリングを実行するチャンネル数 scan_order[] : 各チャンネルのスキャン順 / (未使用 : 本ボードでは固定) range[] : 各チャンネルの入力レンジ番号 / (未使用 : 本ボードでは固定)
戻り値	正常終了時 : = 0 エラー時 : エラーコード / 負の値 (エラーコード表)
機能・動作	サンプリングを実行するチャンネル数、各チャンネルのスキャン順、入力レンジを設定する。 例えばno_ch : 5ならば、スキャン順番0~4のチャンネル群がサンプリング実行対象となる。 本ボードではスキャン順固定 (0, 1, 2, …… 15)、入力レンジはボード上のスイッチ設定。

【3】サンプリング動作モードの設定

書式	int ad_set_sampmode (int trs_mode[], int buf_area, int intr_sw, int dma[])
引数	trs_mode[0] : ADデータ転送方法 / 0 : I/O (ポーリングまたは割り込み)、1 : DMA trs_mode[1] : 割り込み発信フラグ / 0 : EMPTY解消、1 : HALF-FULL buf_area : ADデータ転送先 / 0 : 標準メモリ、1 : EMSメモリ、2 : XMSメモリ intr_sw : I/Oデータ転送で割り込み使用の有無 / 0 : 不使用、1 : 使用する dma[0] : DMAチャンネル指定 / 0 : ch0、3 : ch3 dma[1] : DMA単位データ長指定 / 無視 (本ボードではバイトに固定)
戻り値	正常終了時 : = 0 エラー時 : エラーコード / 負の値 (エラーコード表)
機能・動作	ADボードからパソコン本体メモリへのデータ転送方法、転送先メモリ、割り込み使用の有無、 DMAチャンネル、DMA単位データ長を設定する。割り込みを使用しないI/Oデータ転送の ときはtrs_mode[1]は無視され、ブロックI/O転送を含む最適化されたアルゴリズムで サンプリング&データ転送が実行される。

【4】トリガ関連の設定A（レンジトリガ以外の場合）

書式	int ad_set_trigger(int trg_mode, int trg_source, int trg_pol, int trg_level)
引数	<trg_mode 0="" :="" トリガ動作モード="" ポストトリガ<br="" 即トリガ、1=""></trg_mode> trg_source : トリガ源 / 0 : ソフト、1 : 内部（アナログ）、2 : 外部 trg_pol : トリガ極性 / 0 : 負エッジ、1 : 正エッジ 2 : 負レベル、3 : 正レベル trg_level : トリガレベル（アナログ） / 対応するADデータコードで指定する。
戻り値	正常終了時 : = 0 エラー時 : エラーコード / 負の値（エラーコード表）
機能・動作	トリガ動作モード、源、極性、レベル等の設定。 なお、トリガ動作モードをポストトリガ、トリガ源を外部（デジタル）で極性をレベルに指定したときは帯域サンプリングとなる。（3-1項参照）

【4】' トリガ関連の設定B（レンジトリガの場合）

書式	int ad_set_rangetrigger(int trg_sel, int trg_level_hi, int trg_level_lo)
引数	trg_sel : トリガ動作モード / 0 : アウトレンジ、1 : インレンジ 2 : デュアルスロープ（+）、3 : デュアルスロープ（-） trg_level_hi : トリガレベル上限値 trg_level_lo : トリガレベル下限値
戻り値	正常終了時 : = 0 エラー時 : エラーコード / 負の値（エラーコード表）
機能・動作	レンジトリガの極性を含む動作モード、レベルの設定。（レンジトリガ以外は前記【4】を使用）

【5】オプション、または外部クロック源周波数値の設定

書式	int ad_set_exclk (ULONG exclk_freq)
引数	exclk_freq : オプション、または外部クロック源の周波数値（Hz 単位）
戻り値	正常終了時 : = 0 エラー時 : エラーコード / 負の値（エラーコード表）
機能・動作	オプション、または外部クロック源を使用するとき、その周波数値を設定する。 後記【6】ad_set_clockで分周比によるサンプリングクロック指定、またはボードに標準搭載の内部クロック源を使用するときは必要ない。

【6】サンプリング・クロック値の設定

書式	int ad_set_clock(int clk_source, int set_mode, int *time_unit, ULONG *clk_period)
引数	<p>clock__source: クロック源 / 0: 内部クロック源 0 (1.000 MHz) 1: 内部クロック源 1 (1.024 MHz) 2: 内部クロック源 2 (オプション) 3: 内部クロック源 3 (未使用) 4: 外部クロック源 (有効極性 =) 5: 外部クロック源 (有効極性 =)</p> <p>set__mode : クロック値の指定方法 / 0: クロック周期の値、 1: 分周比 time__unit : クロック周期の単位 / 0: s、 1: ms、 2: μs、 3: ns clk__period: クロック周期の値、または分周比</p>
戻り値	<p>正常終了時 : = 0 エラー時 : エラーコード / 負の値 (エラーコード表)</p>
機能・動作	<p>サンプリングクロック値を設定する。 なお、設定できないクロック周期の値を指定すると設定可能な長い方の近似値が設定される。 また、クロック源に 2, 4, 5 を指定し、クロック周期の値で設定するときは前記【5】ad__set__exclckでクロック源周波数値を定義しておく。</p>

【7】サンプリング・スタート (ADデータバッファが32K語以内のとき)

書式	int ad_start_samp (ULONG no_samp, WORD *bufptr, WORD bufsize)
引数	<p>no__samp : サンプリング・データ点数 (1チャンネル当りの点数) bufptr : ADデータ・バッファのポインタ bufsize : ADデータ・バッファの大きさ</p>
戻り値	<p>正常終了時 : = 0 エラー時 : エラーコード / 負の値 (エラーコード表)</p>
機能・動作	<p>サンプリングを実行する。 前記【4】トリガ設定でトリガ動作モードを《即トリガ》としてあるときは即スタート、また《ポストトリガ》が選択されているときはトリガ待ちスタートとなる。 前記【4】' で設定するレンジトリガのときは全てトリガ待ちスタートとなる。 データ転送先が拡張メモリEMS, XMSの場合、bufptrとbufsizeは無視される。 なお前回サンプリング実行時にセットされたフラグはここでクリアされるが、《エラー》フラグだけはクリアされない。 《エラー》フラグをクリアするには 前記【1】初期化を実行する。</p>

【7】' サンプリング・スタート (ADデータバッファが32K語を超えるとき)

書式	int ad_start_samp_h (ULONG no_samp, WORD _huge *bufptr, ULONG bufsize)
引数	<p>no__samp : サンプリング・データ点数 (1チャンネル当りの点数) bufptr : ADデータ・バッファのポインタ bufsize : ADデータ・バッファの大きさ</p>
戻り値	<p>正常終了時 : = 0 エラー時 : エラーコード / 負の値 (エラーコード表)</p>
機能・動作	<p>サンプリングを実行する。 前記【4】トリガ設定でトリガ動作モードを《即トリガ》としてあるときは即スタート、また《ポストトリガ》が選択されているときはトリガ待ちスタートとなる。 前記【4】' で設定するレンジトリガのときは全てトリガ待ちスタートとなる。 データ転送先が拡張メモリEMS, XMSの場合、bufptrとbufsizeは無視される。 なお前回サンプリング実行時にセットされたフラグはここでクリアされるが、《エラー》フラグだけはクリアされない。 《エラー》フラグをクリアするには 前記【1】初期化を実行する。</p>

【8】ステータス取得

書式	int ad_get_status (ULONG &sampld, int status[])
引数	sampld : サンプルング済みデータ点数 (1チャンネル当り) status[0] : ADボードのステータス1 / (3-13項) status[1] : ADボードのステータス2 / (未使用: 本ボードでは無い)
戻り値	正常終了時 : 連続サンプルング実行中 = 0、 停止中 = 1 エラー時 : エラーコード / 負の値 (エラーコード表)
機能・動作	当時点でのサンプルング済みデータ点数、およびADボードのステータス生データを得る。

【9】本ハンドラの終了

書式	int ad_close_adsys (void)
引数	なし
戻り値	正常終了時 : = 0 エラー時 : エラーコード / 負の値 (エラーコード表)
機能・動作	本ハンドラの終了。 ADボードのリセット、確保したメモリ領域の開放等を行う。

【10】拡張メモリ内のサンプルング・データ読み込み

書式	int ad_read_exmem(ULONG no_data, ULONG data_pos, WORD *bufptr, WORD bufsize)
引数	no_data : 拡張メモリから読み出すデータ点数 (1チャンネル当り) data_pos : 拡張メモリから読み出すデータの先頭位置 (サンプルング順番号) bufptr : 転送先ADデータバッファのポインタ bufsize : 転送先ADデータバッファの大きさ
戻り値	正常終了時 : = 0 エラー時 : エラーコード / 負の値 (エラーコード表)
機能・動作	サンプルングされたADデータが拡張メモリ (EMS, XMS) に格納されているとき、任意の部分を標準メモリ上のADデータバッファに転送する。

【11】マニュアル (1回) サンプルング・スキャン

書式	int ad_get_onescan (WORD *bufptr)
引数	bufptr : ADデータバッファのポインタ
戻り値	正常終了時 : = 0 エラー時 : エラーコード / 負の値 (エラーコード表)
機能・動作	前記【2】で指定したチャンネル群に対して各1回だけサンプルングを実行する。

【12】汎用デジタル（ラッチ）出力

書式	void ad_out_aux (int out_data)
引数	out_data : 汎用8bitデジタル（ラッチ）出力データ
戻り値	なし
機能・動作	汎用8bit（ラッチ）出力データの更新。 当出力ポートは電源投入（ハードウェア・リセット）時：00H となるが、ボードの制御部リセット（ソフト的リセット）では変化しない。

【13】汎用デジタル（現在値）入力

書式	int ad_inp_aux (void)
引数	なし
戻り値	汎用8bitデジタル（現在値）入力データ
機能・動作	汎用8bitデジタル（現在値）入力データの読み込み。

【14】データバッファをリング状／エンドレスに設定

書式	int ad_set_samplloop (void)
引数	なし
戻り値	正常終了時 : = 0 エラー時 : エラーコード / 負の値（エラーコード表）
機能・動作	ADデータバッファをリング状・エンドレスで使用するかどうかの設定。（トグル） リング状・エンドレスに設定した場合、ADデータバッファが満杯になると次は同バッファ先頭を上書きする形となる。この時、ステータス取得関数【8】で得るサンプリング済みデータ点数の値も先頭値（ゼロ）に戻る。 本設定によりADボードは無限サンプリング・モードとなる。 長時間の監視システム等に利用できる。

【15】ADデータ・コードの設定

書式	int ad_set_datacode (int data_code)
引数	data_code : ADデータ・コード指定 / 0:バイナリ、 1:2の補数
戻り値	正常終了時 : = 0 エラー時 : エラーコード / 負の値（エラーコード表）
機能・動作	ADボードに対して変換出力データ・コード（符号化の型）を指定する。

【16】サンプリング動作の強制停止

書式	int ad_stop_samp (void)
引数	なし
戻り値	正常終了時 : = 0 エラー時 : エラーコード / 負の値 (エラーコード表)
機能・動作	サンプリングを強制的に中止する。

【17】残り A D データの読み込み

書式	long ad_read_restdata ()
引数	なし
戻り値	正常終了時 : = 読み込んだ (残り) データ数 エラー時 : エラーコード / 負の値 (エラーコード表)
機能・動作	データロス・エラー発生 (検出するとサンプリングを強制停止させる)、したとき、A D ボード上の F I F O バッファメモリに残りデータがあれば、これを当関数で読み込むことができる。

【18】(B R E A K キー押下による) トリガ待ち、またはサンプリング中止の設定 / 解除

書式	int ad_onkey_quit (int set_mode)
引数	set_mode : 動作の指定 / 0 : 解除、 1 : 設定
戻り値	正常終了時 : = 0 エラー時 : エラーコード / 負の値 (エラーコード表)
機能・動作	【 B R E A K = C T R L + P A U S E 】キーの押下によりトリガ待ち (サンプリング動作中のときはサンプリング) を強制的に中止する。 前記【7】ad_start_sampの実行によってトリガ待ち状態となったにもかかわらず何かの理由でトリガが発生しない場合、人為的に待ちループから抜ける手段となる。

【19】(キー押下による) 即トリガ動作の設定 / 解除

書式	int ad_onkey_trg (int act_key)
引数	act_key : 動作の指定 (動作 key の選択) / 0 : 解除、 1 : E S C、 2 : S P A C E、 3 : E N T R
戻り値	正常終了時 : = 0 エラー時 : エラーコード / 負の値 (エラーコード表)
機能・動作	選択したキーの押下により強制的に即トリガ動作する。 前記【7】ad_start_sampの実行によってトリガ待ち状態となったにもかかわらず何かの理由でトリガが発生しない場合、人為的に待ちループから抜ける手段となる。