契約番号 2007 情財第 311 号

２００７年度
オープンソースソフトウェア活用基盤整備事業

「Linux カーネルへの組み込みシステム向け
メモリ管理方式の実現」

API 仕様書

平成２０年１月

リネオソリューションズ株式会社

目次

用語定義

用語	意味
CABI	Common Accounting Blocking Interface システム上の資源の利用をプロセスグループ単位で管理するためのインターフェイス。
AO	Accounting Object CABI で資源管理を行うためのオブジェクト。

1. 概要

本書では、IPA オープンソースソフトウェア活用基盤整備事業「Linux カーネルへの組み込みシステム向けメモリ管理方式の実現」において開発されたシステムの API 仕様を説明する。

2. データ構造

2.1 AO 構造体

AO の実体となる構造体である。

各 AO のプロパティと資源管理用関数のポインタが含まれている。

```
struct cabi_account {
   cabi_object_t               cabi_id;          /* AO ID */
   account_type_t              type;             /* 管理資源種別 */
   struct list_head   cabi_link;                 /* 次 AO へのリンク */
   struct list_head   proc_list;                 /* プロセスのリスト */
   struct proc_dir_entry *     proc_account_dir; /* procfs のエントリ */
   union {
            cpu_account_t      cpu;              /* CPU 管理用構造体 */
            mem_account_t      mem;              /* メモリ管理用構造体 */
   }                           res;
   int (* create) (cabi_account_t, cabi_uaccount_t);  /* 生成関数 */
   int (* set) (cabi_account_t, cabi_uaccount_t);     /* パラメータ設定関数 */
   int (* get) (cabi_account_t, cabi_uaccount_t);     /* パラーメタ取得関数 */
   int (* init) (cabi_account_t);                /* 初期化関数 */
   int (* exit) (cabi_account_t);                /* 破棄関数 */
   int (* start) (struct task_struct *);         /* 資源管理開始関数 */
   int (* end) (struct task_struct *);           /* 資源管理終了関数 */
   int (* proc_create) (cabi_account_t);         /* proc エントリ生成関数 */
   int (* proc_destroy) (cabi_account_t);        /* proc エントリ破棄関数 */
};
```

2.2 メモリ資源管理構造体

ユーザが指定したメモリ管理に関するパラメータが含まれている。

AO が破棄されるまで保持される。

```
struct mem_param {
   cabi_mem_op_flag_t op_flag;          /* 警告時オペレーション指定フラグ */
   cabi_mem_obj_flag_t obj_flag;        /* オブジェクトモード指定フラグ */
   unsigned long user_limit;            /* ユーザメモリ上限値*/
   unsigned long file_limit;            /* ファイルキャッシュ上限値 */
   unsigned long user_warn;             /* ユーザメモリ警告閾値 */
   unsigned long file_warn;             /* ファイルキャッシュ警告閾値 */
   unsigned long reclaim_pages;         /* 一度に回収するページ数 */
   struct {
            pid_t pid;                  /* シグナル送信先 PID */
            int sig;                    /* 送信シグナル番号 */
            int flag;                   /* 確認用フラグ（未使用） */
   } signal;
};
```

2.3 ユーザ AO 構造体

アカウンティング API でユーザとのパラメータのやり取り（set、get）を行うための構造体である。

```
struct cabi_uaccount {
   cabi_object_t               cabi_id;          /* AO ID */
   account_type_t              type;             /* 管理資源種別 */
   union {
            cpu_uaccount_t     cpu;              /* CPU 管理用パラメータ */
            mem_uaccount_t     mem;              /* メモリ管理用パラメータ */
   }                           res;
};
```

2.4 メモリ資源管理パラメータ構造体

アカウンティング API でユーザとメモリ資源管理に関するパラメータのやり取り（set、get）を行うための構造体である。

```
struct mem_uaccount {
   cabi_mem_unit_t mem_unit;           /* サイズの単位指定 */
   cabi_mem_op_flag_t op_flag;         /* 警告時オペレーション指定フラグ */
   cabi_mem_obj_flag_t obj_flag;       /* オブジェクトモード指定フラグ */
   unsigned long user_limit;           /* ユーザメモリ上限値*/
   unsigned long file_limit;           /* ファイルキャッシュ上限値*/
   unsigned long user_warn;            /* ユーザメモリ警告閾値 */
   unsigned long file_warn;            /* ファイルキャッシュ警告閾値 */
   unsigned long reclaim_pages;        /* 一度に回収するページ数 */
   struct {
            pid_t pid;                 /* シグナル送信先 PID */
            int sig;                   /* 送信シグナル番号 */
            int flag;                  /* 確認用フラグ（未使用） */
   } signal;
};
```

3. アカウンティングシステムコール

3.1 sys_cabi_account_create

[書式]

int sys_cabi_account_create(struct cabi_uaccount *ucabi)

[引数]

struct cabi_uaccount *ucabi　　ユーザ AO 構造体

[説明]

AO を新規に作成する。

引数 ucabi に以下のパラメータを設定する。

- ucabi->type 資源種別
  ２：メモリ
- ucabi->res.mem->unit　サイズの単位指定
  ０：ページフレーム数（通常 1 ページ＝4096byte）
  １：％（トータルメモリ量に対する割合）
  ２：byte
  ３：kbyte
  ４：Mbyte
  ５：Gbyte
- ucabi->res.mem->op_flag　オペレーションフラグ
  ※下記設定値は組み合わせて設定可能
  １：何もしない
  ２：シグナル送信
  ４：ドライバの select 起床
  ８：ページ回収
  ０を設定するとデフォルト値として８（ページ回収）が設定される
- ucabi->res.mem->obj_flag　オブジェクトフラグ
  １：強制バインド有効
- ucabi->res.mem->user_limit　ユーザメモリ上限値
  単位は ucabi->res.mem->unit による。
  ０より大きく搭載メモリ量より小さい値を設定すること。
- ucabi->res.mem->file_limit　ファイルキャッシュ上限値
  単位は ucabi->res.mem->unit による。
  ユーザメモリ上限値以下の値を設定すること。
  0 を設定するとデフォルト値としてユーザメモリ上限値と同じ値が設定される。

- ucabi->res.mem-> user_warn　ユーザメモリ警告値
  単位は ucabi->res.mem->unit による。
  ユーザメモリ上限値以下の値を設定すること。
  0 を設定するとデフォルト値としてユーザメモリ上限値の 80%の値が設定される。
- ucabi->res.mem-> file_warn　ファイルキャッシュ警告値
  単位は ucabi->res.mem->unit による。
  ファイルキャッシュ上限値以下の値を設定すること。
  ０を設定するとデフォルト値としてファイルキャッシュ上限値の 80%の値が設定される。
- ucabi->res.mem->reclaim_pages　ページ回収時の一回の回収数
  オペレーションフラグでページ回収を選択している場合のみ有効。
  ０を設定するとデフォルト値として３２（ページ）が設定される。
- ucabi->res.mem->signal.pid　シグナル送信先 PID
  オペレーションフラグでシグナル送信を選択している場合のみ有効。
- ucabi->res.mem->signal.sig　送信シグナル番号
  オペレーションフラグでシグナル送信を選択している場合のみ有効。
- ucabi->res.mem->signal.flag　確認用フラグ（未使用）

[返り値]

成功すると CABI_SUCCESS を返す。また引数 ucabi->cabi_id にオブジェクト ID を返す。オブジェクト ID は 2 から 4294967295 までの整数値である。

失敗した場合以下のエラーコードを返す。

CABI_CREATE_ERR　不正なパラメータ、メモリ不足、パーミッション違反等

3.2 sys_cabi_account_destroy

[書式]

int sys_cabi_account_ destroy (unsigned long cabi_id)

[引数]

unsigned long cabi_id　　オブジェクト ID

[説明]

指定された AO を削除する。

[返り値]

成功すると CABI_SUCCESS を返す。

失敗した場合以下のエラーコードを返す。

CABI_EACCESS　　パーミッション違反

CABI_ENOEXIST　　AO が存在しない

CABI_EATTACHED	まだプロセスがアタッチされている
CABI_EINVAL	不正なパラメータ

3.3 sys_cabi_account_bind_pid

[書式]

int sys_cabi_account_bind_pid (unsigned long cabi_id, pid_t pid)

[引数]

unsigned long cabi_id	オブジェクト ID
pid_t pid	プロセス ID

[説明]

指定したプロセスを AO に登録する。

[返り値]

成功すると CABI_SUCCESS を返す。

失敗した場合以下のエラーコードを返す。

CABI_EACCESS	パーミッション違反
CABI_ENOEXIST	AO が存在しない
CABI_EPNOEXIST	プロセスが存在しない
CABI_EATTACHED	まだプロセスがアタッチされている
CABI_EREGIST	プロセスが既に指定した AO に登録されている
CABI_ENOAVLE	プロセスが既に別の AO に登録されている
CABI_ENOMEM	メモリ不足
CABI_EINVAL	不正なパラメータ
CABI_ERROR	その他エラー

3.4 sys_cabi_account_bind_pgid

[書式]

int sys_cabi_account_bind_pgid (unsigned long cabi_id, pid_t pgid)

[引数]

unsigned long cabi_id	オブジェクト ID
pid_t pgid	プロセスグループ ID

[説明]

指定したプロセスグループを AO に登録する。

[返り値]

成功すると CABI_SUCCESS を返す。

失敗した場合以下のエラーコードを返す。

CABI_EACCESS	パーミッション違反
CABI_EPGNOEXIST	プロセスグループが存在しない
CABI_EREGIST	プロセスが既に指定した AO に登録されている
CABI_ENOAVLE	プロセスが既に別の AO に登録されている
CABI_ENOMEM	メモリ不足
CABI_EINVAL	不正なパラメータ
CABI_ERROR	その他エラー

3.5 sys_cabi_account_unbind

[書式]

int sys_cabi_account_unbind(pid_t pid)

[引数]

pid_t pid	プロセス ID

[説明]

指定したプロセスを AO から切り離す。

[返り値]

成功すると CABI_SUCCESS を返す。

失敗した場合以下のエラーコードを返す。

CABI_EACCESS	パーミッション違反
CABI_EINVAL	不正なパラメータ
CABI_EPNOEXIST	プロセスが存在しない
CABI_NOBIND	指定したプロセスがどの AO にも登録されていない

3.6 sys_cabi_account_get

[書式]

int sys_cabi_account_get (unsigned long cabi_id, struct cabi_uaccount *ucabi)

[引数]

unsigned long cabi_id	プロセス ID
struct cabi_uaccount *ucabi	ユーザ AO 構造体

[説明]

指定した AO のパラメータを取得する。

[返り値]

成功すると CABI_SUCCESS を返す。また引数 ucabi に現在のパラメータを返す。

ucabi には以下の値が設定されている。

- ucabi->type 資源種別
  ２：メモリ
- ucabi->res.mem->unit　サイズの単位指定（常に０）
  ０：ページフレーム数（通常１ページ＝4096byte）
- ucabi->res.mem->op_flag　オペレーションフラグ
  １：何もしない
  ２：シグナル送信
  ４：ドライバの select 起床
  ８：ページ回収
- ucabi->res.mem->obj_flag　オブジェクトフラグ
  １：強制バインド有効
- ucabi->res.mem->user_limit　ユーザメモリ上限値
  単位はページフレーム数
- ucabi->res.mem->file_limit　ファイルキャッシュ上限値
  単位はページフレーム数
- ucabi->res.mem-> user_warn　ユーザメモリ警告値
  単位はページフレーム数
- ucabi->res.mem-> file_warn　ファイルキャッシュ警告値
  単位はページフレーム数
- ucabi->res.mem->reclaim_pages　ページ回収時の一回の回収数
- ucabi->res.mem->signal.pid　シグナル送信先 PID
- ucabi->res.mem->signal.sig　送信シグナル番号
- ucabi->res.mem->signal.flag　確認用フラグ（未使用）

失敗した場合以下のエラーコードを返す。

CABI_EINVAL	不正なパラメータ
CABI_ENOEXIST	AO が存在しない
CABI_EINVAL	不正なパラメータ
CABI_ERROR	その他エラー

3.7 sys_cabi_account_set

[書式]

int sys_cabi_account_set (unsigned long cabi_id, struct cabi_uaccount *ucabi)

[引数]

unsigned long cabi_id	プロセス ID
struct cabi_uaccount *ucabi	ユーザ AO 構造体

[説明]

指定した AO のパラメータを変更する。

引数 ucabi 構造体の内容については 3.1 sys_cabi_account_create 参照。

[返り値]

成功すると CABI_SUCCESS を返す。

失敗した場合以下のエラーコードを返す。

CABI_ENOEXIST	AO が存在しない
CABI_EINVAL	不正なパラメータ
CABI_ERROR	その他エラー

4. アカウンティング API

4.1 cabi_account_create

[書式]

int cabi_account_create (struct cabi_uaccount *ucabi)

[引数]

struct cabi_uaccount *ucabi　　ユーザ AO 構造体

[説明]

AO を新規に作成するライブラリ関数。

引数 ucabi 構造体の内容については 3.1 sys_cabi_account_create 参照。

[返り値]

成功すると CABI_SUCCESS を返す。また引数 ucabi->cabi_id にオブジェクト ID を返す。オブジェクト ID は 2 から 4294967295 までの整数値である。

失敗した場合エラーコードを返す。

エラーコードは sys_cabi_account_create と同様。

4.2 cabi_account_destroy

[書式]

int cabi_account_destroy (unsigned long cabi_id)

[引数]

unsigned long cabi_id　　オブジェクト ID

[説明]

指定された AO を削除するライブラリ関数。

[返り値]

成功すると CABI_SUCCESS を返す。

失敗した場合エラーコードを返す。

エラーコードは sys_cabi_account_destroy と同様。

4.3 cabi_account_bind_pid

[書式]

int cabi_account_bind_pid (unsigned long cabi_id, pid_t pid)

[引数]

unsigned long cabi_id　　オブジェクト ID

pid_t pid　　プロセス ID

[説明]

指定したプロセスを AO に登録するライブラリ関数。

[返り値]

成功すると CABI_SUCCESS を返す。

失敗した場合以下のエラーコードを返す。

エラーコードは sys_cabi_account_bind_pid と同様。

4.4 cabi_account_bind_pgid

[書式]

int cabi_account_bind_pgid (unsigned long cabi_id, pid_t pgid)

[引数]

unsigned long cabi_id	オブジェクト ID
pid_t pid	プロセスグループ ID

[説明]

指定したプロセスグループを AO に登録するライブラリ関数。

[返り値]

成功すると CABI_SUCCESS を返す。

失敗した場合エラーコードを返す。

エラーコードは sys_cabi_account_bind_pgid と同様。

4.5 cabi_account_unbind

[書式]

int cabi_account_unbind(pid_t pid)

[引数]

pid_t pid	プロセス ID

[説明]

指定したプロセスを AO から切り離すライブラリ関数。

[返り値]

成功すると CABI_SUCCESS を返す。

失敗した場合エラーコードを返す。

エラーコードは sys_cabi_account_unbind と同様。

4.6 cabi_account_get

[書式]

int cabi_account_get (unsigned long cabi_id, struct cabi_uaccount *ucabi)

[引数]

unsigned long cabi_id	プロセス ID
struct cabi_uaccount *ucabi	ユーザ AO 構造体

[説明]

指定した AO のパラメータを取得するライブラリ関数。

[返り値]

成功すると CABI_SUCCESS を返す。また引数 ucabi に現在のパラメータを返す。

ucabi の内容については 3.6 sys_cabi_account_get を参照。

失敗した場合エラーコードを返す。

エラーコードは sys_cabi_account_get と同様。

4.7 cabi_account_set

[書式]

int cabi_account_set (unsigned long cabi_id, struct cabi_uaccount *ucabi)

[引数]

unsigned long cabi_id	プロセス ID
struct cabi_uaccount *ucabi	ユーザ AO 構造体

[説明]

指定した AO のパラメータを変更するライブラリ関数。

引数 ucabi 構造体の内容については 3.1 sys_cabi_account_create 参照。

[返り値]

成功すると CABI_SUCCESS を返す。

失敗した場合エラーコードを返す。

エラーコードは sys_cabi_account_set と同様。

5. procfs 機能

本機能は、ユーザがシステム並びにアプリの使用メモリ量を把握するのに必要な情報を提供することにより、使用メモリ量の予測を可能とすることを目的とする。メモリリソース管理情報は以下のディレクトリに作成する。

/proc/cabi/mem/

5.1 procfs のディレクトリ構成

本機能の/proc 以下のディレクトリ構成を以下に示す。

```
/proc
  |-cabi
    `-mem/
      |-info          } ①システムの情報
      |-cmaps
      `-<ao id>/      } ②AO 毎の情報
        |-status
        `-<pid>/      } ③プロセス毎の情
          `-pmaps
```

① システムの情報
- info：メモリ情報（搭載メモリ量、リニアアドレス情報）
- cmaps：全ページフレームのマッピング情報

② AO 毎の情報　各 AO 毎に ID でディレクトリが作成される。
- status：パラメータ、メモリ使用量、メモリ使用量のピーク値

③ プロセス毎の情報　各プロセスごとに PID でディレクトリが作成される。
- pmaps：プロセス空間の仮想アドレスに対するページフレームのマッピング情報をヒープ、スタック、ファイルマッピング等の各 VM 領域別に表示する。

5.1.1 info（メモリ情報）

システム全体に関するメモリサイズ情報、及びリニアアドレスの情報を表示する。以下に出力結果を示す。なお、表示されているサイズやアドレスはシステムによって異なる。

```
Memory size information:
 total memory:            229312kb            搭載メモリ量
 total page frame:         57328              総ページフレーム数
 reserved page:             1179              予約されたページフレーム数

Liner address information:
 process space:          00000000-bfffffff    プロセス空間
 kernel straight map:    c0000000-cdfeffff    ストレートマップ空間
  zone_dma:              c0000000-c0ffffff    DMA ゾーン
   kernel text:          c0100000-c02a22f8    カーネルテキスト領域
   kernel data:          c02a22f9-c0350197    カーネルデータ領域
   kernel bss:           c0350198-c03a1923    カーネル BSS 領域
  zone_nomal:            c1000000-cdfeffff    NOMAL ゾーン
 kernel virtual area:    ce800000-ffffafff    カーネル仮想空間
 fixed map area:         ffffd000-ffffefff    固定マップ空間
```

5.1.2 cmaps（全ページフレームのマッピング情報）

全ページフレームのページ構造体を走査し、各ページフレームの現在の状態を一覧表示する。以下に出力結果を一部省略して示す。

```
.=Free, -=Cache, +=Dirty, B=Buddy, A=Anonymous, R=Reserved, S=Slab, W=Swap
N:Z:PFN--++-------+-------+-------+-------+-------+-------+-------+-------
0:0:00000|RRB.B...B.......B..............B..............................
0:0:00040|B.............................................................
0:0:00080|B.............................RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
0:0:000c0|RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
0:0:00100|RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
0:0:00140|RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
0:0:00180|RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
0:0:001c0|RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
0:0:00200|RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
0:0:00240|RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
0:0:00280|RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
0:0:002c0|RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
0:0:00300|RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
0:0:00340|RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRB.......B..............
0:0:00380|B.......B...B.RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRBRBB.B...
0:0:003c0|B.............................................................
0:0:00400|B.............................................................
0:0:00440|.............................................................
0:0:00480|.............................................................
      ＜中略＞
0:0:00f80|.............................................................
0:0:00fc0|.............................................................
0:2:01000|RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
0:2:01040|RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRSSSSSSS.SSS.SSSSSSSSSS.SSSSSSS
0:2:01080|SSSSS---SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
0:2:010c0|SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSBSSSSSSSSSSSSSSSSSAASS--SSSSSS
0:2:01100|S.SSSSSSSSS..SSS.SS.SS.SSSSSSS-SSSSSSS--SS----SS.SSSSS.SSSS.SS
0:2:01140|SSSSSSSSSSSSSSSSSSSS-A----SSSS-AA-SSSSB.-SB---SSSSSSSSSSSSSSS
0:2:01180|SSSSSSSSSSSSSSS.SSSSSSS---------SSS.SS-S----SSSS----SSSSSSSSSS
0:2:011c0|SSS-SS-A----SAS.----SS..SSA-SS-S-----S..----SSSSSSSSSSS-S--A-S
0:2:01200|SSSSSSS-S---SSSS----B-SS--AASSSSSSSSSSSSSSSS--------SSSSSSSB...
0:2:01240|SS.SSSS-SSSSSSS-----AAASSSS----ASSS---SASSS----SSSSSAAASSSSSS--
0:2:01280|.---SSS.SSS-AA--SA.-SS.--S----S-S---B.SS---ASSSSSS.S.SS--SSSSSS
0:2:012c0|SSSS----SSB.SSSSSSSSSSA-------SSSS----SS-SSS--SSSS----SSB.----
0:2:01300|SSSBS------ASSSS----SSSSSSS----A-------SSSSSS-----SSSSSS.SSASS
0:2:01340|SS--SS.SSSSS-----------------------------------------------A-
```

①インデックス情報　　②ページフレーム状態一覧

① インデックス情報

左から[ノード番号：ゾーン番号：ページフレーム番号（16 進）]を表す。一行で 64 ページ分の情報を表示するため。ページフレーム番号は 0x40 ずつ増える。

② ページフレーム状態一覧

一文字が 1 ページフレームを表す。各記号の意味は下表の通りである。

記号	意味
.（ピリオド）	未使用
-（ハイフン）	ページキャッシュとして使用している
D	ダーティな状態
B	バディシステムで管理する連続空き領域の先頭
A	無名ページ
R	予約領域
S	スラブとして使用中
W	スワップキャッシュまたはスワップ処理中

5.1.3 status（AO の情報）

AO に設定されているパラメータ、アカウンティング情報等を表示する。

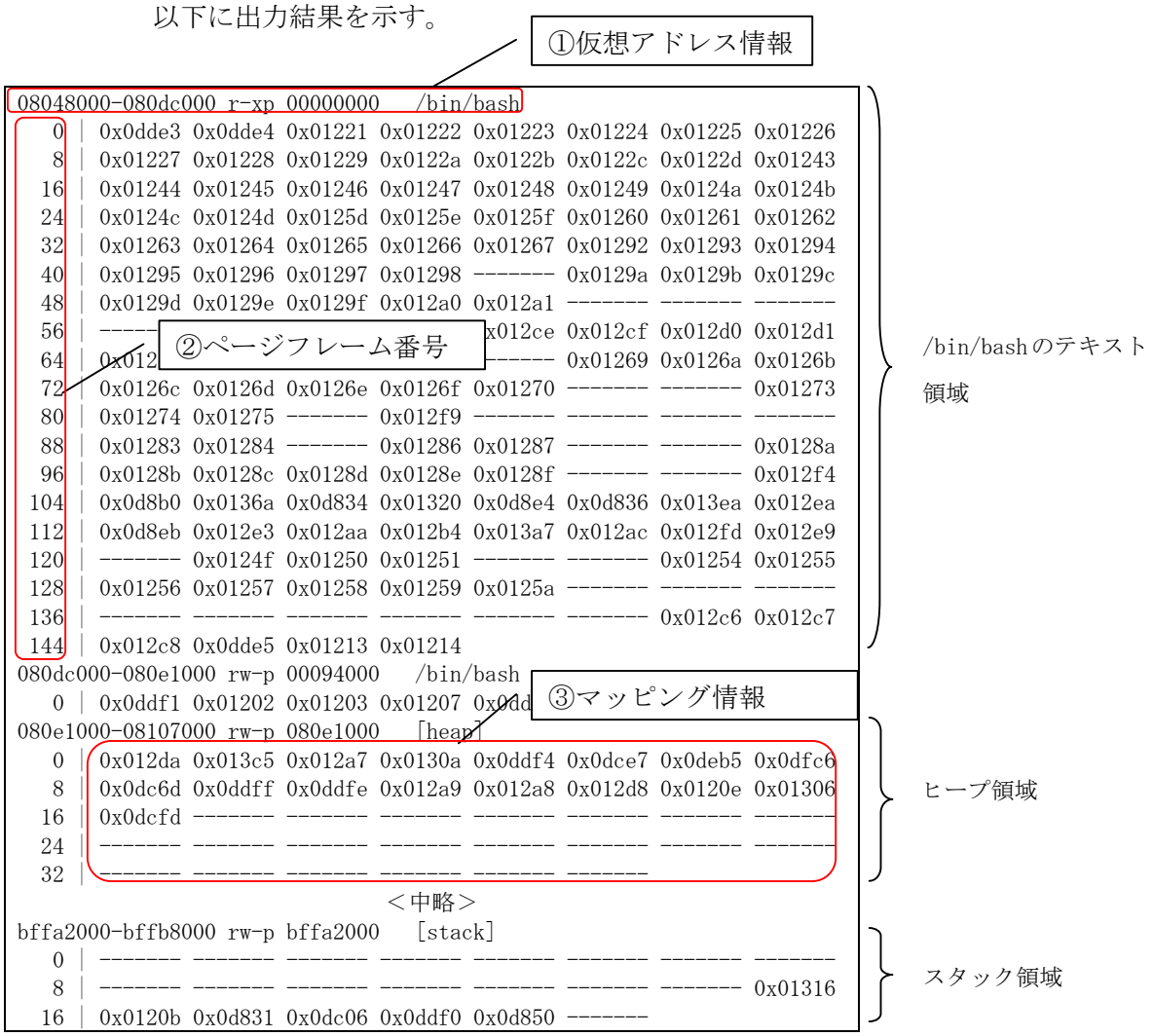
以下に出力結果を示す。

```
user_limit :    10240kb          使用可能メモリ量
user_warn  :     8192kb          警告メモリ量
user_max   :      972kb          メモリ使用量のピーク値
user_use   :      924kb          現在のメモリ使用量
file_limit :    10240kb          使用可能ページキャッシュ量
file_warn  :     8192kb          警告ページキャッシュ量
file_max   :       24kb          ページキャッシュ使用量のピーク値
file_use   :       24kb          現在のページキャッシュ使用量
signal pid :        0            シグナル送信先ＰＩＤ
signal sig :        0            送信シグナル番号
signal flag: 0x000000            シグナルフラグ（未使用）
```

5.1.4 pmaps（プロセスの情報）

ＡＯにバインドされているプロセスの仮想メモリ（ＶＭ）の情報を表示する。

以下に出力結果を示す。




```
08048000-080dc000 r-xp 00000000   /bin/bash
   0 | 0x0dde3 0x0dde4 0x01221 0x01222 0x01223 0x01224 0x01225 0x01226
   8 | 0x01227 0x01228 0x01229 0x0122a 0x0122b 0x0122c 0x0122d 0x01243
  16 | 0x01244 0x01245 0x01246 0x01247 0x01248 0x01249 0x0124a 0x0124b
  24 | 0x0124c 0x0124d 0x0125d 0x0125e 0x0125f 0x01260 0x01261 0x01262
  32 | 0x01263 0x01264 0x01265 0x01266 0x01267 0x01292 0x01293 0x01294
  40 | 0x01295 0x01296 0x01297 0x01298 ------- 0x0129a 0x0129b 0x0129c
  48 | 0x0129d 0x0129e 0x0129f 0x012a0 0x012a1 ------- ------- -------
  56 | -------                                 0x012ce 0x012cf 0x012d0 0x012d1
  64 | 0x012                                   ------- 0x01269 0x0126a 0x0126b
  72 | 0x0126c 0x0126d 0x0126e 0x0126f 0x01270 ------- ------- 0x01273
  80 | 0x01274 0x01275 ------- 0x012f9 ------- ------- ------- -------
  88 | 0x01283 0x01284 ------- 0x01286 0x01287 ------- ------- 0x0128a
  96 | 0x0128b 0x0128c 0x0128d 0x0128e 0x0128f ------- ------- 0x012f4
 104 | 0x0d8b0 0x0136a 0x0d834 0x01320 0x0d8e4 0x0d836 0x013ea 0x012ea
 112 | 0x0d8eb 0x012e3 0x012aa 0x012b4 0x013a7 0x012ac 0x012fd 0x012e9
 120 | ------- 0x0124f 0x01250 0x01251 ------- ------- 0x01254 0x01255
 128 | 0x01256 0x01257 0x01258 0x01259 0x0125a ------- ------- -------
 136 | ------- ------- ------- ------- ------- ------- 0x012c6 0x012c7
 144 | 0x012c8 0x0dde5 0x01213 0x01214
080dc000-080e1000 rw-p 00094000   /bin/bash
   0 | 0x0ddf1 0x01202 0x01203 0x01207 0x0dd
080e1000-08107000 rw-p 080e1000   [heap]
   0 | 0x012da 0x013c5 0x012a7 0x0130a 0x0ddf4 0x0dce7 0x0deb5 0x0dfc6
   8 | 0x0dc6d 0x0ddff 0x0ddfe 0x012a9 0x012a8 0x012d8 0x0120e 0x01306
  16 | 0x0dcfd ------- ------- ------- ------- ------- ------- -------
  24 | ------- ------- ------- ------- ------- ------- ------- -------
  32 | ------- ------- ------- ------- ------- -------
                          ＜中略＞
bffa2000-bffb8000 rw-p bffa2000   [stack]
   0 | ------- ------- ------- ------- ------- ------- ------- -------
   8 | ------- ------- ------- ------- ------- ------- ------- 0x01316
  16 | 0x0120b 0x0d831 0x0dc06 0x0ddf0 0x0d850 -------
```

① 仮想アドレス情報

左から仮想アドレス範囲、パーミッション、オフセット、パス名（または領域名）を表す。/proc/[pid]/maps で出力される情報と同様である。

② ページフレーム番号

ページフレーム番号を示す。1 行で 8 ページ分の情報が表示されるので 8 ずつインクリメントされる。

③ マッピング情報

各仮想アドレスにページフレームが割り当てられている場合はページフレーム番号を表示する。割り当てられていない場合ハイフンを表示する。