バンチ振動モニター（20M メモリーボード）のデータ構造

飛山　真理

☆メモリーボードには 20Mbyte =(20,971,520 bytes)のメモリーがありますが、アドレス管理としては２０ビットのアドレス（1,048,576 バイト）がついた下位メモリーと 0 から 19 までの上位バンク切り替えからなっています。下位アドレスは２０ビットありますが、実際の記録は FDMUX で３２チャンネルにデマルチプレクスされるため、３２個単位で記録しています。メモリーボードにストップシグナルが入りますと、その時点でデータ記録を停止し、その時点でのアドレス(下位アドレス/32 及びバンク)を記録します。

☆メモリーボードのスタートはリングの revolution と同期していますので、通常はたとえば address0 がバンチ ID0 なら、データ取得を繰り返してもこの関係は変わりません。

☆メモリーボードのデバイスサポートはアドレス 0 からバンチ順に 0,1,2,3...5119,0,1,2...とデータをディスク上に書き、ストップ時間に関する並べ替えはしません。時間順に再構成するためには、データの先頭に記録してあるストップアドレス情報を元にデータの並べ替えを行う必要があります。

☆ディスクに記録してあるデータの構造は以下のようになります。

b1 b2 b3 b4 [0,0] [0,1] [0,2] ...[0,5119][1,0][1,1][1,2]..[1,5119][2,0]...[4095,5118][4095,5119]

ここで b1 b2 b3 b4 等はストップアドレス(byte)、[0,0]等はメモリーデータ(byte)です。
このうち b1 と b2 が下位アドレス/32、b3(常に 0)と b4 がストップバンクアドレスです。

☆実際にメモリーボードが止まったアドレスは以下のように計算します。

stop1(下位アドレス):=b1*256+b2;
stop2(バンク):=b3*256+b4;

stop address :=　1048576*stop2 + stop1*32;

☆上で述べたように、データのストップは 32 バンチ単位ですから、このアドレスは一般にある周データの途中になります。つまり、ストップがかかった周回ではあるアドレスまでは最新の（最後の）データで、それ以降は一番古いデータということになります。この 1turn の違いを無視して、たとえばストップ後半の一番古いデータをストップ周の後ろにつけてしまう処理をすると、バンチごとの振動データ bunch[i,j],i=0..5119, j=0..4095 は以下のようなコードで得られます。

(pascal)

```
  j := ((stop address) div 5120 + 1)* 5120;
  for i := 0 to 20971519 do
   begin
     if (j>20971519) then j := j-20971520;
     mem_temp[i] := memory.mem[j];
     inc(j);
   end;
  for i:=0 to 20971519 do memory.mem[i]:=mem_temp[i];
  for k:= 0 to  4095 do for j := 0 to 5119 do bunch[j,k]:=
       memory.mem[j+k*5120];
```

(当てにならないC)

```
#include <stdio.h>
#include <stdlib.h>
#include <strings.h>
unsigned char *memory_data,*memory_temp,
                    (*bunch_data)[4096];
char file_name[255];
FILE *FHAND,*FW1,*FW2;
long bunch;
div_t a;

int main()
{
long i,j,k,kk,size;
unsigned short s1,s2;;
 memory_data = (unsigned char* )calloc(20971520,
                       sizeof(char));
 if (memory_data == NULL){
      printf("cannot alloc memory_data¥n");
      exit(0);}
 memory_temp = (unsigned char* )calloc(20971520,
                       sizeof(char));
 if (memory_temp == NULL){
      printf("cannot alloc memory_temp¥n");
      exit(0);}
bunch_data = (unsigned char(*)[4096])calloc(5120*4096,
                  sizeof(char));
if (bunch_data == NULL){
      printf("cannot alloc bunch_data¥n");
      exit(0);}
printf("input file name¥n");
scanf("%s",file_name);
FHAND=fopen(file_name,"r");
if (FHAND==NULL) exit(0);
fread(&s1,2,1,FHAND);
fread(&s2,2,1,FHAND);
size=fread(memory_temp,1,20971520,FHAND);
fclose(FHAND);
printf("Bunch %d s1 %d s2 %d size %d¥n",bunch,s1,s2,size
  );
k=1048576*s2+s1*32;
a = div(k,5120);
k=(a.quot+1)*5120;
for (i=0;i<20971520;i++){
      if (k>20971519) k= k-20971520;
      memory_data[i] =  memory_temp[k];
      k++;}
for (j=0;j<4096;j++){
 for (i=0;i<5120;i++){
   bunch_data[i][j] = memory_data[j*5120+i];}}

FW1=fopen("FULL.DAT","w+");
FW2=fopen("ONE.DAT","w+");
printf("position¥n");
scanf("%d",&bunch);
 for (i=0;i<5120;i++){
   fprintf(FW1,"%d %d¥n",i,memory_data[i]);}
for (i=0;i<4096;i++){
      fprintf(FW2,"%d %d¥n",i,bunch_data[bunch][i]);}
exit(0) ; }
```