ABSTRACT
Using SAS® software to construct randomization schedules for clinical trails can be time consuming and complicated. Schedules are used at the sites to assign patients to treatment groups. This paper shows one approach for creating randomization schedules that are unbiased and utilizes SAS® random functions to assign random block sizes and then randomly assign patients at a site to treatment groups.

INTRODUCTION
To ensure that patients were assigned without bias, it is ideal to create randomization schedules using multiple block sizes (see Table 1). This posed a challenge to our biostatisticians since they normally used a single block size (see Table 1).

<table>
<thead>
<tr>
<th>SINGLE BLOCK SIZE OF 2</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Trial</td>
<td>Trial</td>
<td>Trial</td>
<td>Trial</td>
<td></td>
</tr>
<tr>
<td>Control</td>
<td>Control</td>
<td>Control</td>
<td>Control</td>
<td></td>
</tr>
<tr>
<td>Trial</td>
<td>Control</td>
<td>Control</td>
<td>Trial</td>
<td></td>
</tr>
<tr>
<td>Control</td>
<td>Control</td>
<td>Trial</td>
<td>Control</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MULTIPLE BLOCK SIZES</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Trial</td>
<td>Trial</td>
<td>Trial</td>
<td>Control</td>
<td></td>
</tr>
<tr>
<td>Control</td>
<td>Control</td>
<td>Control</td>
<td>Control</td>
<td></td>
</tr>
<tr>
<td>Trial</td>
<td>Control</td>
<td>Trial</td>
<td>Trial</td>
<td></td>
</tr>
<tr>
<td>Control</td>
<td>Control</td>
<td>Trial</td>
<td>Control</td>
<td></td>
</tr>
</tbody>
</table>

Table 1 – Schedules using single and multiple block sizes

This led to some problems:
1. Sites can determine how the schedule was generated by looking for patterns in the randomization schedule.
2. SAS® procedures can be difficult to understand and use.
3. Few schedules are created each year.
4. No formal process in place for creating tables.

The initial request to data management was to produce 1:1 randomization and randomly assign different block sizes until all patients for the site were assigned. The program is now able to produce several types of randomizations (i.e. 1:1, 1:2, 2:1, 1:3 and 1:4).

The biostatisticians can store the parameters passed to the macro in the calling program, and then re-produce an identical table at any time, if necessary.

SAS® PROCEDURES
Base SAS® and the macro facility were used along with SAS® Procedures PROC PLAN®, PROC REPORT® and PROC PRINT®. This paper will briefly examine the procedures used in the macro.

PROC PLAN
This statistical procedure was used to randomly assign subjects to treatments and assign treatments to blocks. This procedure has many options and this paper only shows the options used by this macro. The syntax is

```
PROC PLAN SEED=<seed value>;
   FACTORS UNIT=<Number of patients>;
   OUTPUT DATA=<Input Dataset> OUT=<Output Dataset>;
```
Table 2 shows how the PROC PLAN assigns treatments to patients with a UNIT of six. Patients 2, 4 and 6 are assigned to treatment 1 (Trial) and patients 1, 3 and 5 are assigned to treatment 2 (Control).

<table>
<thead>
<tr>
<th>UNIT</th>
<th>TREAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>

**Table 2 – Sample output from PROC PLAN**

PROC REPORT
The report procedure and Output Deliver System (ODS®) are used to format the randomization schedule(s). Output is stored in RTF format.

PROC PRINT
The print procedure is used to print the detail used to construct the randomization schedule. The detail was very helpful in debugging and testing the macro.

**PROBLEM / REQUIREMENTS**
The statisticians needed a way to create randomization schedules that was flexible and easy to use. Multiple block sizes had to be used to help eliminate any potential bias. Treatments had to be randomly assigned in each block. Lastly, the randomization schedule needed to be reproducible. These requirements led to the creation of the SAS® Randomization Schedule Macro.

**MACRO**
The macro, SAS® Randomization Schedule, randomly selects blocks sizes and assigns a treatment to each patient.

Output files and the SAS® Log are saved in the current working directory.

For a block size of four and a 1:1 randomization, there would be two patients assigned to the Control group and two to the Trial group.

Multiple block sizes may be used and must be a multiple of the sum of factors (ie. 1:1 can use 2, 4, 6, 8, 10 and 1:2 can use 3, 6, 9).

A seed value must be set. The program will add 100 to the seed for each site (center) processed and then add 5 to the seed value for each block processed. This will ensure that each site (center) and block have different seed values. Using this approach makes it possible to rebuild the randomization tables.

Error checking is done to validate parameters passed to the macro.

**PARAMETERS**
The RandomizationTables macro has seven named parameters.

- **Random_Seed** - seed value used by proc plan to randomize values
- **Block_Sizes_Needed** - List of desired blocks sizes. Default block sizes for 1:1 is 2 4 6 8 10 12; 1:2 is 3 6 9 12; 1:3 is 4 8 12 and 2:3 is 5 10.
- **Subjects_Needed** - Number of Subjects need for each site
- **Number_Of_Sets** - Number of rows in randomization schedule table
- **Number_Of_Centers** - Number of centers (sites) being processed
- **Ratio** - Ratio of patients assigned to Trial and Control groups
- **Inv_Device** - Device or Drug being investigated. Default equals ‘Trial’.
- **Cnt_Device** - Device or Drug in the control group. Default is ‘Control’.
- **PrintDetail** – Print tables showing how each block was randomized. Default is ‘N’.
MACRO CALL
A randomization schedule and detail listings can be produced for each site. The following call to the randomization table macro produces the table to the right (Table 3).

```sas
%RandomizationTables(
   Random_Seed=2007,
   Block_Sizes_Needed= 2 4 6,
   Subjects_Needed=20,
   Number_Of_Sets=4,
   Number_Of_Centers=1,
   Ratio=1:1,
   Inv_Device = Trial,
   Cnt_Device = Control,
   PrintDetail = N
);
```

<table>
<thead>
<tr>
<th>First Set</th>
<th>Second Set</th>
<th>Third Set</th>
<th>Fourth Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trial</td>
<td>Trial</td>
<td>Trial</td>
<td>Control</td>
</tr>
<tr>
<td>Control</td>
<td>Control</td>
<td>Control</td>
<td>Control</td>
</tr>
<tr>
<td>Trial</td>
<td>Control</td>
<td>Trial</td>
<td>Trial</td>
</tr>
<tr>
<td>Control</td>
<td>Control</td>
<td>Trial</td>
<td>Trial</td>
</tr>
<tr>
<td>Trial</td>
<td>Trial</td>
<td>Control</td>
<td>Control</td>
</tr>
</tbody>
</table>

Table 3 - Randomization Schedule

CONSIDERATIONS
Make a copy of the program. Although this macro is stored in our SAS® macro library is could change over time. If the randomization tables would need to be generated again, the original version of the program will need to be used. Otherwise, the results could be different depending on the change that was made.

CONCLUSION
The statisticians are quite happy with this robust randomization macro because it is easy to use and saves them a lot of time. It greatly simplifies the processes and removes potential bias in the clinical trial.

REFERENCES

ACKNOWLEDGMENTS
Special thanks to John Tescula who added some enhancements to the macro and Carleton Southworth, statistician, for making the request and then helping to test and verify the results.

CONTACT INFORMATION
Your comments and questions are valued and encouraged. Contact the author at:

Michael Stout
Team Lead, SAS® Programming
DePuy Orthopaedics, Inc.
PO Box 988
700 Orthopaedic Drive
Warsaw, Indiana 46581-0988
Work Phone: (574) 372 7379
Fax: (574) 371 4950
E-mail: Mstout2@dpyus.jnj.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies.
RANDOMIZATION SCHEDULE MACRO CODE

Program: Randomization Tables
Path: \Programs\DM\RandomizationTables.sas
Created By: Mike Stout
Date: 26Jul2006

Description:
Randomize patients in blocks with different block sizes (Block_Sizes_Needed) and put them into sets (Number_Of_Sets). Block sizes must be even numbers and separated by blank spaces. For example, to use five block sizes use: 2 4 6 8 10.

SPECIAL INSTRUCTIONS
Copy this program to the directory where you want to store your randomization tables. Make changes to parameters listed below and submit the job. All of the randomization tables will be put in the same directory where you run this program from. It is important to keep a copy of this program with the randomization tables, since the program could change and produce different results.

LIMITATIONS
Max of 10 blocks sizes may be used.
Max of 99 blocks
Max of 20 sets per center
Block Sizes must be a multiple of the ratio 1:1 = Multiples of 2 1:2 Multiples of 3 ect..

HOW THE SEED WORKS
Specify only one seed. The program will add 100 to the seed for each center processed and add 5 to that seed value for each block processed. This will ensure that each center has a different seed value and each block has a different seed value. Using this approach, makes it possible to rebuild the randomization tables.

******************************************************************************
%macro RandomizationTables(Random_Seed=,         /* Random seed */
Block_Sizes_Needed=,   /* Blocks sizes to be used for randomization */
Subjects_Needed=,      /* Number of subjects per Center */
Number_Of_Sets=,       /* Must be a factor of Subjects_Needed */
Number_Of_Centers=,    /* Number of centers */
Ratio=,                /* 1:1, 1:2, 1:3 .... */
Inv_Device = Trial,    /* Name of Trial Group */
Cnt_Device = Control,   /* Name Control Device */
PrintDetail=N
);

******************************************************************************
Default and check parameters passed to the program.
******************************************************************************
data _null_;
  length ratio $ 3;
  Ratio = "&ratio";
  part = scan(ratio,1,':');
  one = input(part,1.0);
  part = scan(ratio,2,':');
  two = input(part,1.0);
  factor = sum(one,two);
call symput('factor',factor);
run;
%if &Block_Sizes_Needed ne %then
  %do;
  data _null_;
    length Blocks $ 20;
    Blocks = "&Block_Sizes_Needed";
  %end;
temp = 'xx';
OK = 'Y';
call symput('Warning', ' ');
do loop = 1 to 10 until (temp = ' '); temp = scan(blocks, loop, ' '); if temp ne ' ' then do;
  Num = input(temp, 2.0);
  fact = &factor;
  if mod(num, fact) ne 0 then OK = 'N';
end;
end;
if OK = 'N' then do;
  put "WARNING: Blocks Needed was invalid as input (&Block_Sizes_Needed) used default.";
  put " Blocks Needed should be a multiple of &factor.";
  call symput('Block_Sizes_Needed', ' ');
  call symput('Warning', "**WARNING** Blocks Needed was invalid as input (&Block_Sizes_Needed) used default.");
end;
Subjects = &Subjects_Needed;
if mod(subjects, fact) ne 0 then do;
  call symput('Warning', "**WARNING** Subjects needed is not a factor of the Ratio.");
end;
run;
%end;
%if &Block_Sizes_Needed = %then
 %do;
   data _null_; if "&factor" = '2' then call symput('Block_Sizes_Needed', '2 4 6 8 10 12');
   if "&factor" = '3' then call symput('Block_Sizes_Needed', '3 6 9 12');
   if "&factor" = '4' then call symput('Block_Sizes_Needed', '4 8 12');
   if "&factor" = '5' then call symput('Block_Sizes_Needed', '5 10');
   run;
 %end;

/**************************************
Assign Labels for Treatment Groups
**************************************/
proc format ;
 value $ Treatmt
'01' = "&Inv_Device        "   /* Investigational Device */
'02' = "&Cnt_Device        ";  /* Control Device */
run;

/**************************************
Count number of Blocks requested and
assign to No_Of_Blocks
**************************************/
data _null_; do until (Xstr = ''); yLoop + 1;
  Xstr= scan("&Block_Sizes_Needed", yLoop);
end;
yLoop = yLoop - 1;
call symput('No_Of_Blocks', left(put(yLoop, 8))); run;

/**************************************
Determine Number of Sets (Columns in table)
data _null_;  
Number_Per_Set = &Subjects_Needed / &Number_Of_Sets;  
call symput('Number_Per_Set', Number_Per_Set);  
run;
/*****************************************************************************/
Create blocks and then use PROC PLAN to randomly select order of treatments in
the block
*****************************************************************************/
%macro SetBlock(CntBlks,Blks,Seed,Grp);
data A&CntBlks;
do UNIT=1 TO &Blks;
   if (UNIT <= &Grp ) then TREAT = 1;
   else TREAT = 2;
%if &Ratio = 2:3 or &ratio = 2:1 %then
  %do;
   if mod(unit+&grp,&factor) = 1 or mod(unit+&grp,&factor) = 2 then treat = 1;
   else TREAT = 2;
  %end;
output;
end;
proc PLAN SEED=&Seed;
   FACTORS UNIT=&Blks;
   output DATA=A&CntBlks OUT=c&CntBlks;
run;
proc SORT;
   by UNIT;
run;
* Mix them up some more *
%let SeedNum = %Eval(&Seed + 8500);
proc PLAN SEED=&SeedNum;
   FACTORS UNIT=&Blks;
   OUTPUT DATA=c&CntBlks OUT=c&CntBlks;
run;
proc sort;
   by UNIT;
run;
* Mix them up some more *
%let SeedNum = %Eval(&Seed + 457);
proc PLAN SEED=&SeedNum;
   FACTORS UNIT=&Blks;
   output DATA=c&CntBlks OUT=c&CntBlks;
run;
proc sort;
   by UNIT;
run;
%mend;
Assign value to a specific set.
*****************************************************************************/
%macro Assign_To_Set;
%do iLoop = 1 %to &Number_Of_Sets;
  %if &iLoop = 1 %then First_Set = put(RandTbl{Row_No,1},$Treatmt.);
  %if &iLoop = 2 %then Second_Set = put(RandTbl{Row_No,2},$Treatmt.);
  %if &iLoop = 3 %then Third_Set = put(RandTbl{Row_No,3},$Treatmt.);
  %if &iLoop = 4 %then Fourth_Set = put(RandTbl{Row_No,4},$Treatmt.);
  %if &iLoop = 5 %then Fifth_Set = put(RandTbl{Row_No,5},$Treatmt.);
  %if &iLoop = 6 %then Sixth_Set = put(RandTbl{Row_No,6},$Treatmt.);
  %if &iLoop = 7 %then Seventh_Set = put(RandTbl{Row_No,7},$Treatmt.);
  %if &iLoop = 8 %then Eigth_Set = put(RandTbl{Row_No,8},$Treatmt.);
  %if &iLoop = 9 %then Ninth_Set = put(RandTbl{Row_No,9},$Treatmt.);
  %if &iLoop = 10 %then Tenth_Set = put(RandTbl{Row_No,10},$Treatmt.);
  %if &iLoop = 11 %then Eleventh_Set = put(RandTbl{Row_No,11},$Treatmt.);
  %if &iLoop = 12 %then Twelfth_Set = put(RandTbl{Row_No,12},$Treatmt.);
  %if &iLoop = 13 %then Thirteenth_Set = put(RandTbl{Row_No,13},$Treatmt.);
  %if &iLoop = 14 %then Fourteenth_Set = put(RandTbl{Row_No,14},$Treatmt.);
  %if &iLoop = 15 %then Fithteenth_Set = put(RandTbl{Row_No,15},$Treatmt.);
  %if &iLoop = 16 %then Sixteenth_Set = put(RandTbl{Row_No,16},$Treatmt.);
  %if &iLoop = 17 %then Seventeenth_Set = put(RandTbl{Row_No,17},$Treatmt.);
  %if &iLoop = 18 %then Eighteenth_Set = put(RandTbl{Row_No,18},$Treatmt.);
  %if &iLoop = 19 %then Nineteenth_Set = put(RandTbl{Row_No,19},$Treatmt.);
  %if &iLoop = 20 %then Twentieth_Set = put(RandTbl{Row_No,20},$Treatmt.);
* More sets can be added if needed *;
%end;
%mend;
*****************************************************************************/
Print Detail for each randomized block
*****************************************************************************/
%macro PrintDetail(PCnt);
%do iloop = 100 %to &PCnt;
  proc print data=c&iloop;
  run;
%end;
%mend;
*****************************************************************************/
Determine which set the record will be assigned to.
*****************************************************************************/
%macro Get_Set_Number;
%do kLoop = 1 %to &Number_Of_Sets;
  %if &kLoop > 1 %then else;
    if cntr / &Number_Per_Set le &kLoop then
      Set_No = &kLoop;
  %end;
%mend;
*****************************************************************************/
Main program used to randomly select block sizes for number of subjects requested. Continue looping until all subject have been assigned to a block.
*****************************************************************************/
%macro RandList(Site= , RandSeed=, PrtDetail=N);
  data groups (drop=blk: drop=j);
  array blks (&No_Of_Blocks) (&Block_Sizes_Needed);
  if _n_ =1 then call symput('blks',dim(blks));
  do UNIT = 1 to 5040;
  proc print data=c&iloop;
  run;
%end;
if &No_of_Blocks = 1 then
do;
   TREAT = blks(1);
end;
if &No_of_Blocks > 1 then
do;
   %do blkcnt = &No_of_Blocks %to 1 %by -1;
      %if &No_of_Blocks = &blkcnt %then
         %do;
         if UNIT > (5040*((&blkcnt-1)/&No_Of_Blocks)) then TREAT = blks(&blkcnt);
         %end;
      %else %if &No_of_Blocks > 1 %then
         %do;
         else if UNIT > (5040*((&blkcnt-1)/&No_Of_Blocks)) then TREAT = blks(&blkcnt);
         %end;
      %else %if &No_of_Blocks = 1 %then
         %do;
         else TREAT = blks(1);
         %end;
   %end;
end;
output groups;
end;
run;
proc PLAN SEED=&RandSeed;
   FACTORS UNIT=5040;
   output DATA=Groups OUT=Agrps;
proc SORT;
   by UNIT;
   run;
quit;
data work.Treatment_Groups;
set Agrps;
   if TotTreat + Treat > &Subjects_Needed then delete;
   TotTreat + Treat;
   if Treat <= &Subjects_Needed then output;
run;
data _null_;
   retain RandomSeed CntBlks;
   if _N_ = 1 then RandomSeed = &RandSeed;
   set work.Treatment_Groups;
   if _N_ = 1 then CntBlks = 100;
   RandomSeed + 100;
   Treatment_Grps = Treat / &factor;
   call execute('%SetBlock('||CntBlks||','||Treat||','||RandomSeed||','||Treatment_Grps||')');
call symput('CntBlks',CntBlks);
run;
*****************************************************************************/
Build a list of blocks used for site
*******************************************************************************/
data _null_;   retain Str;
length Str $500.;
do i = 100 to &CntBlks;
  put i=;
  Str = trim(Str) || ' C' || put(i,3.);
end;
call symput('RanDSN',Str);
run;

/******************************************************************************
Copy blocks into a single dataset
******************************************************************************/
data ALL;
  set &RanDSN;
run;

/******************************************************************************
Assign Treatments, stating with first set (column of table0, second set and 
so on until all sets are filled up with a treatment.
******************************************************************************/
data RanData;
  retain ObsNo;
  retain;
  CntBlks + 1;
array RandTbl (&Number_Per_Set,&Number_Of_Sets) $ _Temporary_; * ;
  set ALL end=alldone;
cntr+1;
  If Row_No < &Number_Per_Set then Row_No+1;
  else Row_No = 1;
%Get_Set_Number
  RandTbl{Row_No,Set_No}= put(Treat,z2.);
  if alldone then
    do;
      do Row_No = 1 to &Number_Per_Set;
        %Assign_To_Set; /* Assign subjects to Set */
        ObsNo+1;
        output RanData;
      end;
    end;
  run;

/******************************************************************************
Print Randomization Table for Site
******************************************************************************/
ods rtf file="&Inv_Device Randomization Table Site &Site..rtf";
title1    "&Inv_Device SUBJECT GROUP ASSIGNMENT";
title2    "&Inv_Device Study";
title3    "Randomization of Ratio &Ratio - Block Sizes &Block_Sizes_Needed";
title4    "Center &Site";
title5    "Warning";
Footnote1 "Random Seed Number = &RandSeed";
Footnote2 "Sasuers\obs\&Inv_Device\Randomization - &Sysdate , &Systime";
proc report data=RanData(drop= Unit Treat cntr Row_No No Set_No CntBlks ObsNo) nowd split='_';
  %If &PrtDetail = Y %then
    %do;
      %printdetail(&CntBlks);
    %end;
run;

/* Close output destination */
ods rtf close;
run;
%mend;

/***************************************************************
Create Report for number of sites specified in Number_Of_Centers
******************************************************************************/
%macro CreateTable;
  %let SeedCnt = %Eval(&Random_Seed);
  %do zLoop = 1 %to &Number_Of_Centers;
    %RandList(Site=zLoop, RandSeed=&SeedCnt, PrtDetail=&PrintDetail );
  %let SeedCnt = %Eval(&SeedCnt + 5);
  %end;
%mend;

%CreateTable;
ods rtf close;
run;
%mend;

option mprint mlogic orientation=portrait
sasautos=('.', SASAUTOS);

%RandomizationTables(
  Random_Seed=2007,
  Block_Sizes_Needed= 2 4 6,
  Subjects_Needed=20,
  Number_Of_Sets=4,
  Number_Of_Centers=1,
  Ratio=1:1,
  Inv_Device = Trial,
  Cnt_Device = Control,
  PrintDetail=Y
);
run;