Using the Gen2 Template

This is an explanation of the [vb.users.template]GEN2.com template followed by examples. Copy this template to your directory and modify it for your needs. DO NOT edit anything in the template directory or its subdirectories.

To use the template you will need to assign certain variables. Here is a list.

******************************

NOTE:
DO NOT CHANGE the assignment operator ie. =, :=, ==, or :=. Single quotes around a name use the value stored in the variable rather than the name. For example
G_Align := 'true' assigns G_Align the value of the variable true which is “1”. False = “0”

******************************
**G_PAT**
The .FRE file you wish to expose. Use the full path, for example:

```
[VB.USERS.USERNAME.CAD]\TEST.FRE
```
or

```
VB$PATS\GCA_KEY.FRE
```

**G_VRU**
The variable resolution unit (VRU) is a power of 2 (1 to 32) which multiplies the pixel size (typically 5 nm or 8 nm) to give the beam step size. For example, if

G_VRU := 4 and pixel size is 5 nm, beam step size is 20 nm.

**G_NUMROWS**

**G_NUMCOLS**
The number of rows and columns in the arrayed exposure of G_PAT

**G_STEPX**

**G_STEPY**
The increment in x and y (in mm) between elements of the array
G_DoseMatrix  If true, the dose used for exposure will be increased from G_DOSE to G_EndDose by an equal percentage across the elements of the arrayed exposures. Whether shooting a dose matrix or not, all relative doses will be shot proportional to G_Dose. G_Dose gets incremented when shooting a matrix. Absolute doses are not affected by the value of G_Dose.

Examples:
SDSE 0 1.2  The dose for clock 0 is set to 1.2 times G_Dose.
SDSE 1 2 /rel  Clock 1 = 2 × G_Dose.
SDSE 3 300 /abs  Clock 3 = 300 µC/cm² regardless of other settings

If exposing a dose matrix, doses 1 and 2 will get incremented and dose 3 will not.

The dose increases with each exposure. The exposure pattern (x,y) for a 2 x 2 array at (0,0) with the step in x and y equal to 1 would be (0,0) (1,0) (0,1) (1,1).

G_Function  If G_Function is true then G_FunctionValue will be executed in the array rather than the exposure. This is useful for doing a dry run without exposing a pattern, by setting G_FunctionValue := @[VB.USERS.TEMPLATE.G2]DoNothing.com
G_ORIGIN This is used to define the global origin and is used for both aligned and unaligned exposures. It can take three values HERE, ARG, or CURRENT. If set to HERE the origin will be set to the stored position HERE. To store a position as HERE go to the desired location and type

SP HERE

If G_ORIGIN is set to ARG you will supply the x and y location as an argument when running your job.

Ex.
@myjob.com 25 25 !Global origin will be set to 25 25 in the current map.

If G_ORIGIN is set to CURRENT the global origin will be set to the current stage location.

G_Align If false, the lower left corner of the cad will be exposed at the global origin (see G_ORIGIN) plus the offsets g_g2d (distance from global origin to lower left die in mm) and G_d2p (distance from die to corner pattern in mm) in the lower left cell of the array and stepped from there.

If true, you will need to define some additional variables.
The template is set up so that you can use any combination of three alignments. Typically you would first prealign.

G_prealign := 'true'

The prealignment will shift the global origin and rotate the map based on the displacement of two marks, one at a distance G_g1 from the origin and another at a distance G_g2. Set these variables to the location of your marks. You can manually align to these marks.

G_manAlPre := 'true'

The script will give you a chance to align with the joystick. For a more accurate pre-alignment you could have the tool locate the marks.

G_AutoAlPre := 'true'

This will require you to set G_GlobalMark to the file that defines the mark type and G_GlobalMarkName to the marks name in that file. Often each die has a set of marks and a pair of these marks on adjacent dies is used as the global marks.

If you have multiple rows and columns you may want to do a wafer alignment

G_waferAlign := 'true'
This will create a coordinate system based on the location of a mark in the dies in the four corners of the array. It uses the first die mark which is defined by G_m1d (the distance from the die origin to the first die mark in mm). If prealignment is used it should not be necessary to manually align these marks.

\[
\begin{align*}
G\_ManAlWafer & := 'false' \\
G\_AutoAlWafer & := 'true'
\end{align*}
\]

For better alignment you may also want to align each die of the array.

\[
G\_DIEALIGN := 'true'
\]

To use die alignment you will also need to define the location of a number of die marks. The variable G_NumDieMarks determines the number of marks that will be used. Each mark is defined by the variable G_m#d , # being a number from 1 to 4. The value assigned to each variable is the distance from the die origin to the mark.
Examples:

EXAMPLE 1
Unaligned exposure

In this example we will shoot a 3 x 3 array of a pattern on a wafer, so that the lower left corner of the central exposure is at the center of the wafer. The spacing between exposures will be 10 mm in x and y. All clocks (from layers) will be exposed at 500 μC/cm².

To center the array on the wafer it is necessary to move the starting location to -10 -10. You may want to move an additional amount to center the pattern rather than the corner of the central die.

To run the following job type:
MVSP CENTRE !note British spelling
MVRL -10 -10
SP HERE
@MYJOB.COM

$! MYJOB - Username  09/20/05
$!!!! DO NOT EDIT ANY JOB IN THE TEMPLATE DIR OR ITS SUB DIRECTORIES !!!!!
$! *************************************************
$ G_pat := [VB.users.username.cad].cad.FRE
$ G_VRU := 1
$ G_NUMROWS := 3
$ G_NUMCOLS := 3
$ G_STEPX := 10
$ G_STEPY := 10
$! *************************************************
$ G_DoseMatrix := 'false'
$ G_DOSE := 500  ! Start dose if DoseMatrix
$ G_EndDose := 1000  ! End dose if DoseMatrix
$!
$ SDSE 0-31 1 ! rel
$ SDSE 0 1 /abs ! rel doses get incremented, abs doses do not.
$ SDSE 1 1
$ SDSE 2 1
$ SDSE 3 1
$! *************************************************
$ G_Function = 'false' ! false => expo pattern true=> do G_FunctionValue
$ G_FunctionValue := @[VB.USERS.TEMPLATE.G2]DoNothing.com !
$! *************************************************
$ G_logging = 'false'
$ G_logdir := [vb.users.your.log]expo.log
$! *************************************************
$ G_ORIGIN := HERE  !Global origin at "HERE", "ARG"(ument Xmm,Ymm ), or "CURRENT" location
$ G_Align := 'false' !0= false 1 = true
$ LL CORNER OF PATTERN WILL BE POSITIONED AT HERE (OR ARG) PLUS G_D2P AND G_G2D
$! IF NO ALIGNMENT IS USED
$!
$!
$ To position at HERE set G_ORIGIN := "HERE"
$!
$! For Alignment you will need mark def files
$! that define a mark of type GlobalName and DieMarkName
$!
EXAMPLE 2  
Aligned example

In this example the pattern will be aligned to 10 \(\mu\)m square marks on the wafer. The CAD was drawn with a bounding box that encloses all shapes. When the CAD was converted to FRE, the bounds were used to set the origin so that patterns taken from the cad had the same extent. The lower left corner of the bounding box will be called the die origin. The four that will be used are located, relative to the origin, at

\[
\begin{align*}
X, Y & \text{ in mm} \\
M1 & = 1, 1 \\
M2 & = 9, 1 \\
M3 & = 9, 9 \\
M4 & = 1, 9
\end{align*}
\]

Before exposing it is necessary to find the location of some feature on the wafer with the prealignment microscope. It is also important to make the x or y axis of the wafer line up with that of the microscope stage. Let’s say the location of M1 in die (col 0, row 0) is measured as
–40 mm, 60 mm relative to CIRCLE1 (a stored position), and this will be the global origin. To run the following job type:

MVSP CIRCLE1
MVRL -40 60 !As measured
SP HERE
@MYJOB.COM

The modifications to the previous job are:

$ G_Align := 'true'
!0= false 1 = true
$! LL CORNER OF PATTERN WILL BE POSITIONED AT HERE (OR ARG) PLUS G_D2P AND G_G2D
$! IF NO ALIGNMENT IS USED
$!
$! To position at HERE set G_ORIGIN := "HERE"
$!
$!
$! For Alignment you will need mark def files
$!
$! that define a mark of type GlobalName and DieMarkName
$!
$!
$ G_GlobalMark := [VB.USERS.USERNAME.MARKS]MARK_PIT.COM
$ G_GlobalMarkName := pit
$ G_DieMark := 'G_GlobalMark' !Same as global in this case
$ G_DieMarkName := pit!
$ G_g1 := 0.0 0.0 !Global mark 1 from global Origin
$ G_g2 := 'g_stepx' 0.0 !Global mark 2
$ G_g2d := -1.0 -1.0 !Offset from global origin to die(0,0) origin
$ G_d2p := 0.0 0.0 !Offset from die to pattern corner
$ G_NumDieMarks = 4 !Number of diemarks 4 is the max
$ G_m1d := 1 1 !Offset diemark 1 from die
$ G_m2d := 9 1 ! " 2
$ G_m3d := 9 9 ! " 3
$ G_m4d := 1 9 ! " 4
$ G_prealign := 'true' ! if false HERE or p1,p2 is mapped to G_g1
$ G_ManAlPre := 'true' ! Find marks manually for pre-alignment
$ G_AutoAlPre := 'true' ! locate marks with tool for "
$ G_waferAlign := 'true'
$ G_ManAlWafer := 'false' ! Find marks manually for wafer alignment
$ G_AutoAlWafer := 'true' ! locate marks with tool for "
$ G_DieAlign := 'true' ! false(wafer only)
$ G_ManAlDie := 'false' ! Find marks manually for die alignment
$ G_AutoAlDie := 'true' ! locate marks with tool for "

This job sets G_GlobalMark to the file [VB.USERS.USERNAME.MARKS]MARK_PIT.COM shown below. The file tries to delete four marks from the mark database. If any one of these four marks is in the database their removal will create room in the database for the new mark PIT created and defined in the last four lines.

$qmark del pit
$qmark del cross
$qmark del inner
$qmark del outer
$qmark def pit/desc= 10um_square_pit
$qmark def pit/sign=(rt=0.5,ct=0.1,filter=8,type=dark,locate=pit)
$qmark def pit/geom=(h=10,w=10,mh=3,mrkt=0.2)
$qmark def pit/scan=(ovsam=8,lines=1,cslim=50,para=4,mlen=1.0,sres=1)
EXAMPLE 3
Jobcal after each die

This job is the same as in Ex. 2 except that the time of exposure of each pattern is long enough to require a jobcal between them.

$! *************************************************
$ G_Function = 'true'  ! false => expo pattern  true=> do G_FunctionValue
$ G_FunctionValue := @[VB. USERS.USERNAME]JcExpo.com !

Now instead of exposing the pattern, the file [VB.USERS.USERNAME]JcExpo.com is run. That file looks like:

$ epat /prog /nopostmove
$ jobcal

Note: this only works perfectly for non-aligned exposures. Because jobcal changes the beam deflection lookup tables from their previous values, the aligned coordinates may no longer be in the same place after the jobcal. Stored global mark positions will be affected. However, jobcal probably won't change the global alignment enough to prevent finding die marks, so it will most likely still be able to do die alignment correctly. The die marks will be found and new, correct die coordinates will be defined after the jobcal.

You could also use a master job file that repeatedly calls jobcal and multiple gen2-based job files, for example:

$ @jobfile1.com
$ jobcal
$ mvpo x2 y2
$ sp here
$ @jobfile2.com

EXAMPLE 4
Different patterns on different rows

This example shows a way to expose different pattern files on each row. If all patterns have the same extent, as they would if all patterns are drawn in the same bounding box and converted as in EXAMPLE 2, this could be done by changing the job file as follows

$ G_FunctionValue := @[VB. USERS.USERNAME]DiffPatExpo.com !

DiffPatExpo.com could be written as:

$ if row .eq. 0 then SPAT [VB.users.username.cad]pat0.FRE
$ if row .eq. 1 then SPAT [VB.users.username.cad]pat1.FRE
$ if row .eq. 2 then SPAT [VB.users.username.cad]pat2.FRE
$ EPAT

Or if the pattern files were named as above

$ SPAT [VB.users.username.cad]pat'row'.FRE
$ EPAT
Again, this will only place the pattern where you expect it to go if all patterns have the same extent.

**EXAMPLE 5**  
**Different patterns, and extents, on different rows using parameters**

This example uses a parameterized job file to expose a different pattern on each row. These patterns may have a different extent. This job file is then called by another (BATCH.COM) with the correct arguments.

The batch file is then run. The necessary changes to MyJob.com are shown below

```
$ G_pat := [VB.users.username.cad]'p1'.FRE
$ G_NUMROWS = 1
$ G_Function = 'false' ! false => expo pattern true=> do G_FunctionValue
$ rowOffset = step_y * p2
$ G_g2d := 0 'rowOffset' !Offset from global origin to die(0,0) origin
$ G_d2p := 'p3' 'p4' !Offset from die to pattern corner
```

Now Batch.com could be written as follows

```
$ MVSP CIRCLE1
$ MVRL -40 60
$ SP HERE
$ ! Pat Row Offset from die origin to pattern corner
$ @MyJob.com pat1 0 0 0
$ @MyJob.com pat2 1 .1 -.1
$ @MyJob.com pat3 2 2.71828 3.14159
```

**EXAMPLE 6**  
**Arrays of arrays**

This example uses nested GEN2 templates to align and expose a 3x3 array of 18x18 arrays. The nested template has been named dieArray.com in this example.

DieArray.com looks like Example 2 with the following changes:

```
$ G_NUMROWS = 18
$ G_NUMCOLS = 18
$ G_STEPX := .5
$ G_STEPY := .5

$ G_ORIGIN := ARG !"HERE", "ARG", or "CURRENT"
```
This set up will do one alignment for all 18 by 18 cells. Turning on The DieAlign would align for each. It will look for the m1 mark at the location of the argument because G_ORIGIN := ARG and G_g2d and G_m1d both equal 0.0.

Myjob.com will be similar to Example 2 with the following changes:

$ G\_g1 := 0.0 0.0 \!{\text{Global mark 1 from global Origin}}$
$ G\_g2 := 'g\_stepx' 0.0 \!{\text{Global mark 2}}$
$ G\_g2d := 0.0 0.0 \!{\text{Offset from global origin to die(0,0) origin}}$
$ G\_d2p := 0.0 0.0 \!{\text{Offset from die to pattern corner}}$
$ G\_NumDieMarks = 4 \!{\text{Number of diemarks 4 is the max}}$
$ G\_m1d := 0 0 \!{\text{Offset diemark 1 from die}}$
$ G\_m2d := .3 0 \!{\text{Offset diemark 2 from die}}$
$ G\_m3d := .3 .3 \!{\text{Offset diemark 3 from die}}$
$ G\_m4d := 0 .3 \!{\text{Offset diemark 4 from die}}$
$ G\_prealign := 'false' \!{\text{if false HERE or p1,p2 is mapped to G\_g1}}$
$ G\_ManAlPre := 'true' \!{\text{Find marks manually for pre-alignment}}$
$ G\_AutoAlPre := 'true' \!{\text{locate marks with tool for "}}$
$ G\_waferAlign := 'true' \!{\text{locate marks with tool for wafer alignment}}$
$ G\_AutoAlWafer := 'true' \!{\text{locate marks with tool for die alignment}}$
$ G\_AutoAlDie := 'true' \!{\text{locate marks with tool for "}}$

Example 7
Array of arrays using “current” location
This example is similar to 6 but uses

$ G\_ORIGIN := CURRENT$

Before the gen2 template exposes or does G\_FunctionValue it moves the stage so that the pattern if exposed will be located at the expected position. It does this with the command

$ mvpo 'tx' 'ty' /corner$

This positions the stage _ field (typically 150u) in x and y from (tx, ty). If you write myJob.com so that g\_p2d is set to the location of m1 of the subarray then do a

$ mvpo 'tx' 'ty'$