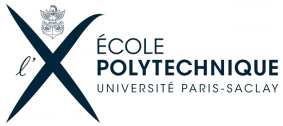
Version 11.2

Geometry

John Apostolakis and Gabriele Cosmo (CERN)

Geant4 Advanced Course

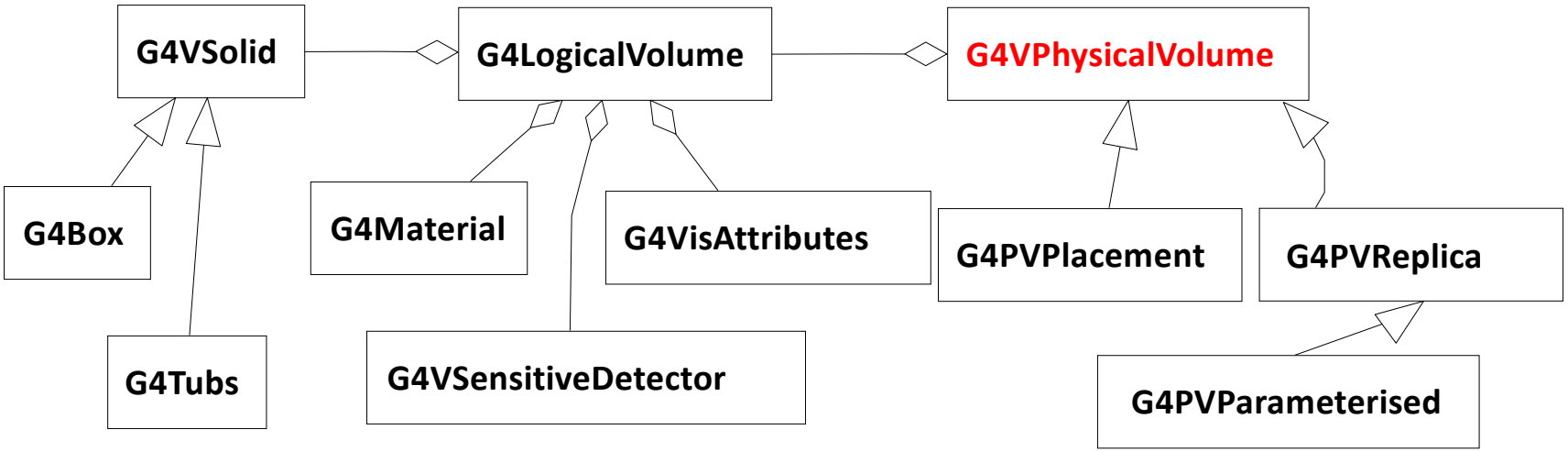


Contents – Part 1

- Placements, Replicated and Parameterised volumes - Introduction
- Touchables and Nested parameterisations
- Divided volumes
- Geometrical regions
- Assembly volumes
- Reflected volumes
- Geometry optimisation
- Parallel geometries
- Moving geometries
- CAD interface

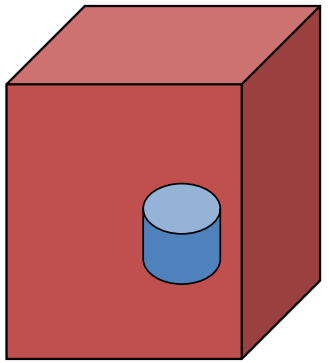
Detector geometry

- Three conceptual layers
  - G4VSolid -- *shape, size*
  - G4LogicalVolume -- *daughter physical volumes,*
    material, sensitivity, user limits, etc.
  - G4VPhysicalVolume -- *position, rotation*

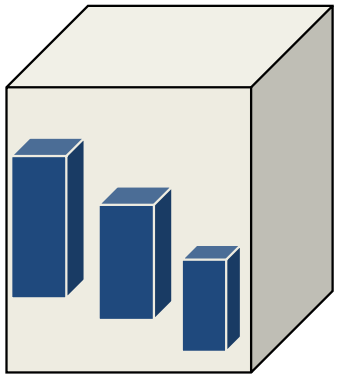


Physical Volumes – placements/repeated

- Placement volume - it is one positioned volume
  - One physical volume object represents one “real” volume
- Repeated volume - a volume placed many times
  - One physical volume object represents any number of “real” volumes
  - reduces use of memory
- Why have ‘repeated’ volumes – what is the reason and impact ?
  - In many setups thousands (or millions) of volumes share (nearly) all characteristics (shape/solid, material, ..) except their location
  - Simple repeated volumes allow us to have one represent many, by just ‘shifting’ its location
  - For each repeated volume that represents 100,000 copies, you will
    - save on the order of 100 MB
    - avoid accesses to that memory, reusing one object of ~1 KB size



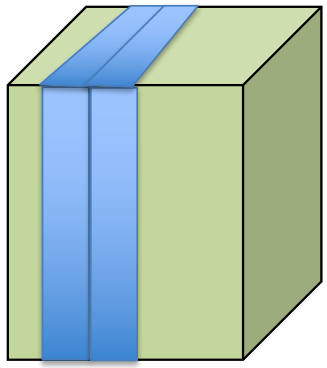
placement



repeated

Types of Repeated Physical Volumes

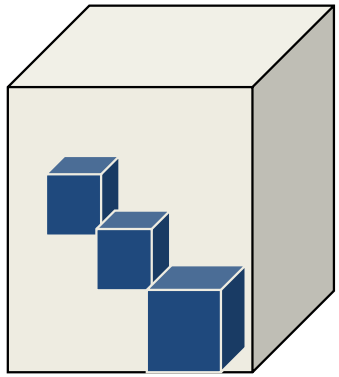
- Repeated volume - a volume placed many times can be
  - *Replica or Division*
    - simple repetition along one axis
  - *Parameterised*
    - The shape, size, material, sensitivity, vis attributes, position and rotation can be parameterised by the copy number



Replica/division

IMPORTANT note:

- A mother volume can contain either
  - Any number of placement volumes, or,
  - One repeated volume



parameterised

Physical Volumes

- G4PVReplica 1 Replica = Many Repeated Volumes
  - Daughters of same shape are aligned along one axis
  - Daughters fill the mother completely without gap in between
- G4PVDivision 1 Division = Many Repeated Volumes
  - Daughters of same shape are aligned along one axis and fill the mother
  - There can be gaps between mother wall and outmost daughters
  - No gap in between daughters

Physical Volumes

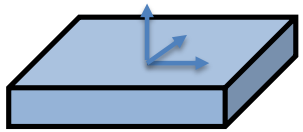
- G4PVReplica — 1 Replica = Many Repeated Volumes
  - Daughters of same shape are aligned along one axis
  - Daughters fill the mother completely without gap in between
- G4PVDivision — 1 Division = Many Repeated Volumes
  - Daughters of same shape are aligned along one axis and fill the mother
  - There can be gaps between mother wall and outmost daughters
  - No gap in between daughters
- G4ReflectionFactory — 1 Placement = a pair of Placement volumes
  - generating placements of a volume and its reflected volume
  - Useful typically for end-cap calorimeter
- G4AssemblyVolume — 1 Placement = a set of Placement volumes
  - Position a group of volumes (leave an imprint one or many times)

Structures

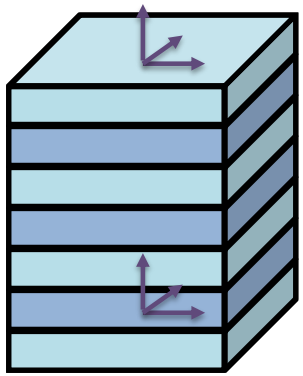
Physical Volumes: Replicas

Replicated Volumes

- The replica daughter volumes must completely fill their mother volume.
- All replicas are the same size (width) and shape
- Replication may occur along:
  - Cartesian axes (X, Y, Z) – slices are perpendicular to this axis
    - Daughter's **origin** of coordinate system is the **center** of a replica
  - Radial axis (Rho) – cons/tubs sections centered on the origin and un-rotated
    - Origin of its coordinate system is the **same** as the mother
  - Phi axis (Phi) – phi sections or wedges, of cons/tubs form
    - Coordinate system rotated such as that the X axis bisects the angle made by each wedge



a daughter logical volume to be replicated



mother volume

G4PVReplica

```
G4PVReplica(const G4String& pName,
            G4LogicalVolume* pLogical,
            G4LogicalVolume* pMother,
            const EAxis pAxis,
            const G4int nReplicas,
            const G4double width,
            const G4double offset=0.);
```

- `offset` may be used only for tube/cone segment
- Features and restrictions:
  - Replicas can be placed inside other replicas
  - Normal placement volumes can be placed inside replicas, assuming no intersection/overlaps with the mother volume or with other replicas
  - No volume can be placed inside a radial replication
  - Simple parameterised volumes cannot be placed inside a replica

Replica - axis, width, offset

- Cartesian axes - **`kXAxis, kYAxis, kZAxis`**
  - Center of n-th daughter is given as

```
-width*(nReplicas-1)*0.5+n*width
```

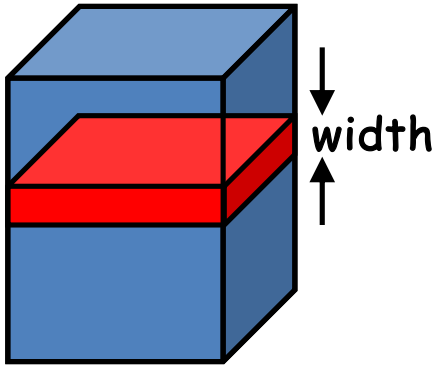
  - Offset shall not be used
- Radial axis - **`kRho`**
  - Center of n-th daughter is given as

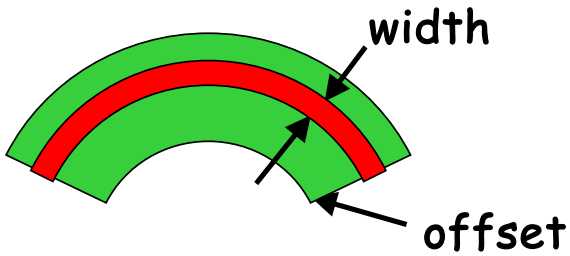
```
width*(n+0.5)+offset
```

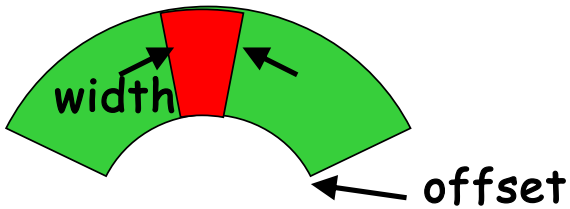
  - Offset must be the inner radius of the mother
- Phi axis - **`kPhi`**
  - Center of n-th daughter is given as

```
width*(n+0.5)+offset
```

  - Offset must be the starting angle of the mother







Physical Volumes: Parameterisations

Physical Volumes

- G4PVPlacement 1 Placement = One Placement Volume
  - A volume instance positioned once in its mother volume
- G4PVParameterised 1 Parameterized = Many Repeated Volumes
  - Generalising the concept of repeated volume, it 'parameterizes' the properties
  - The values of chosen attributes must be calculated by the user, using as input the **copy number**.
  - One or more attributes can be parameterized from
    - Size, shape(type of solid), material, sensitivity, vis attributes, position and rotation
  - How? The user (you) must implement a concrete class of G4VPVParameterisation
  - Reduction of memory consumption

Parameterised Physical Volumes

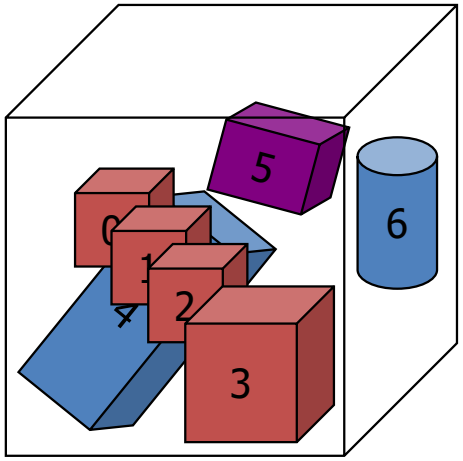
- G4PVParameterised 1 Parameterized = Many Repeated Volumes
  - One or more attributes can be parameterized from
    - Size, shape(type of solid), material, sensitivity, vis attributes, position and rotation

Restrictions:

- A parameterised volume can only have daughter volumes only if each of its instance (the volumes it creates) is identical in size & shape to the rest
  - This allows the grand-daughter volumes a chance to safely fit inside that identical daughter volumes – as required.
- Parameterisations of composed solids like Boolean, Reflected or Displaced solids are not recommended/supported
- A special sub-type called G4PVNestedParameterisation (of G4VPVParameterisation), also has access to the copy numbers of the ancestor volumes (grand-mother etc), which you can use to change (parameterize) the material, sensitivity and vis attributes.

Parameterised Physical Volumes: how to create one?

- You implement a class derived from G4VPVParameterisation abstract base class and define the following as a function of copy number:
  - where that one is positioned (translation, rotation)
  - [optionally] the size of the solid (dimensions)
  - [optionally] the type of the solid, material, sensitivity, vis attributes
- All daughters must be fully **contained** in the mother
- Daughters must **not overlap** to each other
- Limitations/suggestions:
  - Applies to a limited set of solids only
    - Box, Tube, Trd, Simple Trapezoid, Cone, Sphere, Orb, Ellipsoid, Torus, Parallelepiped, Polycone, Polyhedron, Tetrahedron, Hyperboloid
  - Consider parameterised volumes as “leaf” volumes – don’t put another volume inside !
- Typical use-cases
  - Complex detectors with large repetition of volumes, regular or irregular
  - Medical applications: the material in tissue as cubes with varying material as measured in a scan



G4PVParameterised

```
G4PVParameterised(const G4String& pName,
                  G4LogicalVolume* pLogical,
                  G4LogicalVolume* pMother,
                  const EAxis pAxis,
                  const G4int nReplicas,
                  G4VPVParameterisation* pParam
                  G4bool pSurfChk=false);
```

- Replicates the volume **nReplicas** times using the parameterisation **pParam**, within the mother volume **pMother**
- **pAxis** is a "suggestion" to the navigator along which Cartesian axis replication of parameterized volumes dominates
  - kXAxis, kYAxis, kZAxis : one-dimensional optimization
  - kUndefined : three-dimensional optimization

Touchables & Nested Parameterisations

Step Point & Touchable

- A track in Geant4 is composed of steps; **`G4Step`** has two **`G4StepPoint`** objects as its starting and ending points. All the geometrical information of the *current* step should be taken from the "PreStepPoint"
  - The geometrical information associated with G4Track is identical to the "PostStepPoint"
- Each **`G4StepPoint`** object provides:
  - Position in world coordinate system
  - Global and local time
  - Material
  - **`G4TouchableHistory`** for geometrical information
- The **`G4TouchableHistory`** object is a vector of information for each geometrical hierarchy, including:
  - copy number
  - transformation / rotation to its mother
- *Handles* (or *smart-pointers*) to touchables are used intrinsically in Geant4
  - Touchables are reference counted objects

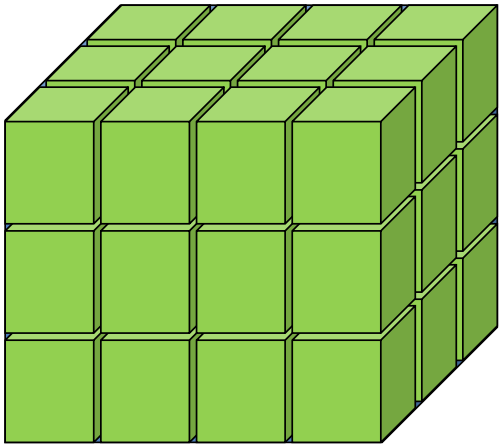
How to get information from a touchable?

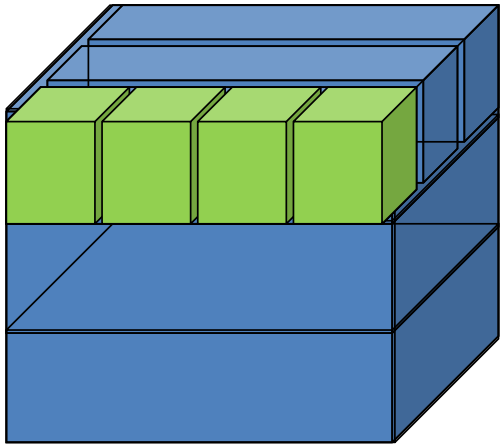
G4TouchableHistory has information on the geometrical hierarchy of the point

```
G4Step* aStep;
G4StepPoint* preStepPoint = aStep->GetPreStepPoint();
G4TouchableHistory* theTouchable =
    (G4TouchableHistory*)(preStepPoint->GetTouchable());
G4int copyNo = theTouchable->GetVolume()->GetCopyNo();
G4int motherCopyNo
           = theTouchable->GetVolume(1)->GetCopyNo();
G4int grandMotherCopyNo
           = theTouchable->GetVolume(2)->GetCopyNo();
G4ThreeVector worldPos = preStepPoint->GetPosition();
G4ThreeVector localPos = theTouchable->GetHistory()
    ->GetTopTransform().TransformPoint(worldPos);
```

Nested Parameterisation

- Suppose a geometry with three-dimensional regular repetition of the same shape and size of volumes without gap between volumes. Material of such volumes are changing according to the position
  - E.g. voxels made by CT Scan data (DICOM)
- Instead of a full three-dimensional parameterised volume:
  - Use replicas for the first and second axes sequentially, and then use one-dimensional parameterisation along the third axis

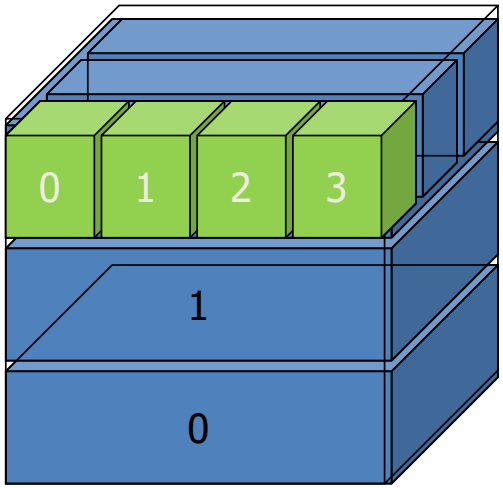




Much less memory required for geometry optimization and much faster navigation for ultra-large number of voxels

Nested Parameterisation

- Given a geometry defined as two sequential replicas and then one-dimensional parameterisation,
  - Material of a single voxel must be parameterised not only by the copy number of the voxel, but also by the copy numbers of the ancestors
  - Material is indexed by three indices



- G4VNestedParameterisation is a special parameterisation class derived from the base class G4VPVParameterisation
  - `ComputeMaterial()` method of G4VNestedParameterisation has a touchable object of the parent physical volume, in addition to the copy number of the voxel
    - Index of first axis = theTouchable->GetCopyNumber(1);
    - Index of second axis = theTouchable->GetCopyNumber(0);
    - Index of third axis = copy number

G4VNestedParameterisation

- G4VNestedParameterisation is a class derived from G4VPVParameterisation
- G4VNestedParameterisation class has three pure virtual methods to be implemented by the user in addition to **`ComputeTransformation()`**, which is a mandatory method for all G4VPVParameterisation classes:

```
virtual G4Material* ComputeMaterial(G4VPhysicalVolume* currentVol,
                                    const G4int repNo,
                                    const G4VTouchable* parentTouch) = 0;
```

  - Returns a material pointer using copy numbers of itself and its ancestors (from the parent touchable)

```
virtual G4int GetNumberOfMaterials() const = 0;
```

  - Returns total number of materials that appear as a return value of **`ComputeMaterial()`** method

```
virtual G4Material* GetMaterial(G4int idx) const = 0;
```

  - Return **`idx`**-th material
  - **`idx`** is not a copy number: **`idx=[0,nMaterial-1]`**

- *These last two methods are needed to inform Geant4 quickly about all materials that show up!*

G4VNestedParameterisation - notes

- As a sub-type of the G4VPVParameterisation class:
  - G4VNestedParameterisation can be used as an argument of G4PVParameterised
  - All other arguments of G4PVParameterised are unaffected
- Nested parameterisations of a placement volume are not supported
  - All levels (of the touchable) used as an index to calculate material must be a repeated volume
  - Also a level of placement volume cannot exist in between

Divided Volumes

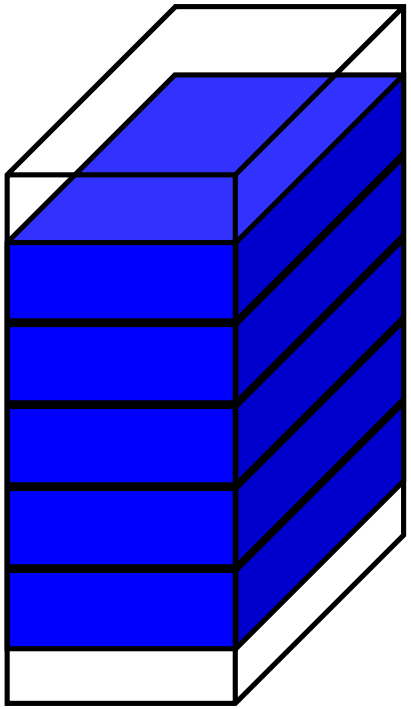
G4PVDivision

- G4PVDivision is similar to G4PVReplica except
  - It allows gaps in between the mother and daughter volumes
- The geometrical shape of all daughter volumes must be the same as for the mother volume.
  - G4VSolid (to be assigned to the daughter logical volume) must be a different object but of the same type
- Replication must be aligned along one axis

➢ For setups with no gaps, should use G4PVReplica instead

  - For identical geometry, navigation in G4PVReplica is faster

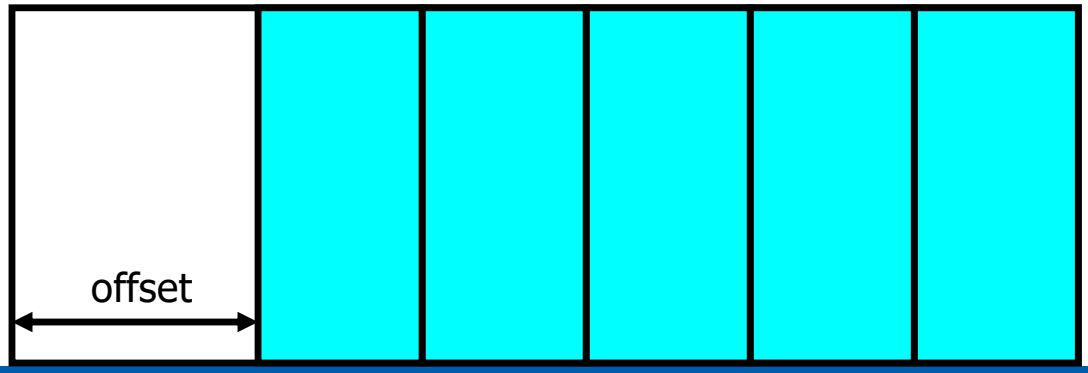
- *Note: A "division" in Geant4 is implemented as a specialized kind of parameterised volume*
  - *G4VPVParameterisation is automatically generated according to the parameters given in G4PVDivision*



G4PVDivision - # of divisions provided

```
G4PVDivision(const G4String& pName,
             G4LogicalVolume* pDaughterLogical,
             G4LogicalVolume* pMotherLogical,
             const EAxis pAxis,
             const G4int nDivisions,
             const G4double offset);
```

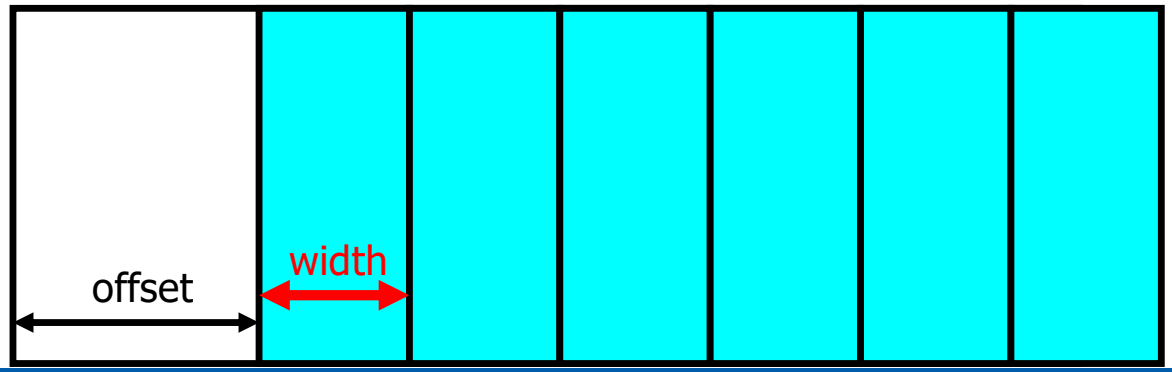
- The size (width) of the daughter volume is calculated as
  ((size of mother) - offset) / nDivisions



G4PVDivision – width of daughter slice provided

```
G4PVDivision(const G4String& pName,
             G4LogicalVolume* pDaughterLogical,
             G4LogicalVolume* pMotherLogical,
             const EAxis pAxis,
             const G4double width,
             const G4double offset);
```

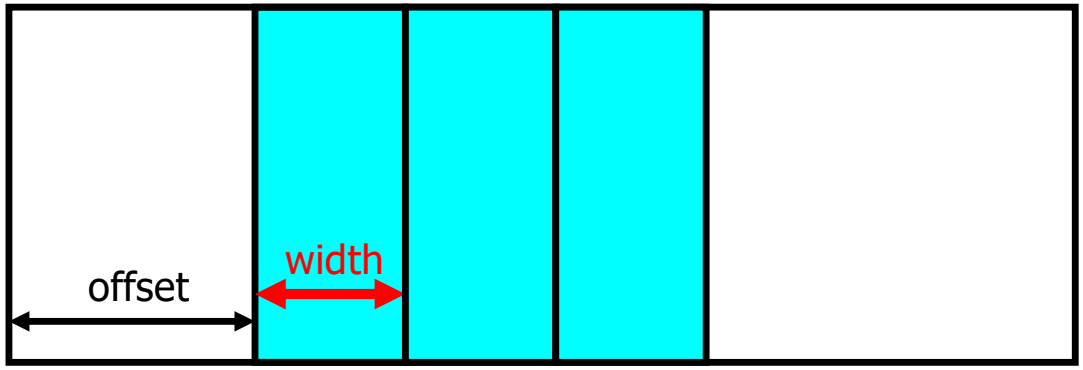
- The number of daughter volumes is calculated as
  `int( ( (size of mother) - offset ) / width )`
  As many daughters as width and offset allow



G4PVDivision – both # of divisions and width of slice provided

```
G4PVDivision(const G4String& pName,
             G4LogicalVolume* pDaughterLogical,
             G4LogicalVolume* pMotherLogical,
             const EAxis pAxis,
             const G4int nDivisions,
             const G4double width,
             const G4double offset);
```

- *nDivisions* daughters of *width* thickness



G4PVDivision – Supported cases

G4PVDivision currently supports following shapes / axes:

- G4Box : *kXAxis, kYAxis, kZAxis*
- G4Tubs : *kRho, kPhi, kZAxis*
- G4Cons : *kRho, kPhi, kZAxis*
- G4Trd : *kXAxis, kYAxis, kZAxis*
- G4Para : *kXAxis, kYAxis, kZAxis*
- G4Polycone : *kRho, kPhi, kZAxis*
  - *kZAxis* - the number of divisions must be the same as solid sections (i.e. numZPlanes-1), the width will not be taken into account
- G4Polyhedra : *kRho, kPhi, kZAxis*
  - *kPhi* - the number of divisions must be the same as solid sides (i.e. numSides), the width will not be taken into account
  - *kZAxis* - the number of divisions must be the same as solid sections (i.e. numZPlanes-1), the width will not be taken into account

G4ReplicatedSlice

- Extension of G4PVDivision allowing gaps in between divided volumes:

```
G4ReplicatedSlice(const G4String& pName, G4LogicalVolume* pDaughterLogical,
                  G4LogicalVolume* pMotherLogical,
                  const EAxis pAxis, const G4int nDivisions,
                  const G4double half_gap, const G4double offset);
G4ReplicatedSlice(const G4String& pName,  G4LogicalVolume* pDaughterLogical,
                  G4LogicalVolume* pMotherLogical,
                  const EAxis pAxis, const G4double width,
                  const G4double half_gap,  const G4double offset);
G4ReplicatedSlice(const G4String& pName, G4LogicalVolume* pDaughterLogical,
                  G4LogicalVolume* pMotherLogical,
                  const EAxis pAxis, const G4int nDivisions, const G4double width,
                  const G4double half_gap, const G4double offset);
```