**SubsetCoordsMulti IP**

**Inputs**

1. S00\_AXI
   1. [IP: axi\_interconnect\_1] M05\_AXI → S00\_AXI
2. num\_of\_subsets\_0 [32-bits]
   1. [IP: ParametersMulti\_0] num\_of\_subsets\_0 → num\_of\_subsets\_0
3. subset\_centerpoint\_x\_0 [32-bits]
   1. [IP: Coords\_Interface\_0] coord\_cx\_0 → subset\_centerpoint\_x\_0
4. subset\_centerpoint\_y\_0 [32-bits]
   1. [IP: Coords\_Interface\_0] coord\_cy\_0 → subset\_centerpoint\_y\_0
5. subset\_size\_0 [32-bits]
   1. [IP: Coords\_Interface\_0] subset\_size\_0 → subset\_size\_0
6. half\_subset\_size\_0 [32-bits]
   1. [IP: Coords\_Interface\_0] half\_subset\_size\_0 → half\_subset\_size\_0
7. subset\_shape\_0 [32-bits]
   1. [IP: Coords\_Interface\_0] subset\_shape\_0 → subset\_shape\_0
8. param\_ready\_0 [1-bit]
   1. [IP: ParametersMulti\_0] param\_done\_0 → param\_ready\_0
9. coord\_interface\_done\_0 [1-bit]
   1. [IP: Coords\_Interface\_0] coord\_interface\_done\_0 → coord\_interface\_done\_0
10. s00\_axi\_aclk [1-bit]
    1. [IP: zynq\_ultra\_ps\_e\_0] pl\_clk0 → s00\_axi\_aclk
11. s00\_axi\_aresetn [1-bit]
    1. [IP: rst\_ps8\_0\_100M] peripheral\_aresetn → s00\_axi\_aresetn

**Associated IPs (inputs):**

1. zynq\_ultra\_ps\_e\_0
2. rst\_ps8\_0\_100M
3. axi\_interconnect\_1
4. ParametersMulti\_0
5. Coords\_Interface\_0

**Outputs**

1. ea\_x\_0 [1-bit]
   1. ea\_x\_0 → ena [IP: blk\_mem\_gen\_6]
2. ea\_y\_0 [1-bit]
   1. ea\_y\_0 → ena [IP: blk\_mem\_gen\_7]
3. wea\_x\_0 [1-bit]
   1. wea\_x\_0 → wea [IP: blk\_mem\_gen\_6]
4. wea\_y\_0 [1-bit]
   1. wea\_y\_0 → wea [IP: blk\_mem\_gen\_7]
5. din\_x\_0 [32-bits]
   1. din\_x\_0 → dina [IP: blk\_mem\_gen\_6]
6. din\_y\_0 [32-bits]
   1. din\_y\_0 → dina [IP: blk\_mem\_gen\_7]
7. addr\_x\_0 [16-bits]
   1. addr\_x\_0 → addra [IP: blk\_mem\_gen\_6]
8. addr\_y\_0 [16-bits]
   1. addr\_y\_0 → addra [IP: blk\_mem\_gen\_7]
9. base\_address\_0 [32-bits]
   1. base\_address\_0 → base\_address\_0 [IP: Gam\_Interface\_0]
   2. base\_address\_0 → probe\_in# [IP: VIO]
10. num\_pxl\_Int\_0 [32-bits]
    1. num\_pxl\_Int\_0 → num\_pxl\_Int\_in\_0 [IP: Gam\_Interface\_0]
11. num\_pxl\_FP\_0 [32-bits]
    1. num\_pxl\_FP\_0 → num\_pxl\_FP\_in\_0 [IP: Gam\_Interface\_0]
    2. num\_pxl\_FP\_0 → probe\_in# [IP: VIO]
12. sub\_done\_0 [1-bit]
    1. sub\_done\_0 → coord\_done\_0 [IP: Coords\_Interface\_0]
    2. sub\_done\_0 → subset\_done\_0 [IP: Gam\_Interface\_0]
    3. sub\_done\_0 → probe\_in# [IP: VIO]
13. coord\_new\_subset\_0 [1-bit]
    1. coord\_new\_subset\_0 → coord\_new\_subset\_0 [IP: Coords\_Interface\_0]
    2. coord\_new\_subset\_0 → probe\_in# [IP: VIO]
14. subset\_counter\_0 [32-bits]
    1. subset\_counter\_0 → coord\_subset\_number\_0 [IP: Coords\_Interface\_0]
    2. subset\_counter\_0 → subset\_counter\_0 [IP: Gam\_Interface\_0]
    3. subset\_counter\_0 → probe\_in# [IP: VIO]

**Associated IPs (outputs):**

1. blk\_mem\_gen\_6 [BRAM 6]
2. blk\_mem\_gen\_7 [BRAM 7]
3. Gam\_Interface\_0
4. Gamma\_Imp\_0
5. Coords\_Interface\_0
6. VIO

**IP Description**

The SubsetcoordsMulti\_0 IP is responsible for finding the pixel indexes within the subsets. The current version of this IP supports squares (‘1’ in the subsets.txt file) and circles (‘0’ in the subsets.txt file) as the shape of the subset. In general, this IP iterates over the total subsets defined as a user-defined parameter and find the pixels that are inside that subset by considering its shape and size. Then the pixel indexes would be saved into two BRAMs (BRAM 6 and BRAM 7). The Gamma\_Imp\_0 IP will read from these two BRAMs to access the subset coordinates (pixel indexes) as needed. The subset information such as subset\_centerpoint\_x\_0 , subset\_centerpoint\_y\_0, subset\_size\_0 , half\_subset\_size\_0 , and subset\_shape\_0 are fetched from BRAM 8 by Coords\_Interface\_0 IP and then is forwarded to this IP. The SubsetcoordsMulti\_0 IP needs this information to find where the pixels are within each subset. At the first state of this IP it will wait until param\_ready\_0 and coord\_interface\_done\_0 are set to “1” (meaning the parameters are saved within the BRAMs and are ready to be used). Then in the next states, the subset\_shape\_0 is checked so that if it is “1” means that the current subset is a square and the pixel index is found in two nested for loops. We start from the subset\_centerpoint\_x\_0 - half\_subset\_size\_0 for x-direction and subset\_centerpoint\_y\_0 - half\_subset\_size\_0 for the y-direction, to reach to the last column of the first row. Then x is incremented by “1” and y would be subset\_centerpoint\_y\_0 - half\_subset\_size\_0 to jump to the first column of the next row. This is continued to x becomes subset\_centerpoint\_x\_0 + half\_subset\_size\_0 and y becomes subset\_centerpoint\_y\_0 + half\_subset\_size\_0 which is the last column of the last row then we just to the second state and wait for the next subset. if it is “0” the current subset is a circle and again using two nested for loops the pixel is computed and stored in BRAM. In this case, the Euclidean distance of the pixels that are within range of subset\_centerpoint\_x\_0 - half\_subset\_size\_0 to subset\_centerpoint\_x\_0 + half\_subset\_size\_0 and subset\_centerpoint\_y\_0 - half\_subset\_size\_0 to subset\_centerpoint\_y\_0 + half\_subset\_size\_0 are compared with the radius squared. If the distance is less than the radius squared (subset\_size\_0 is set to be radius and half\_subset\_size\_0 is assumed to have the radius squared for circle subsets) it means the pixel is within the subset range and so it is stored in BRAM 6 and 7. Then again, we jump to the second state and wait for the next subset. This IP also sets the base\_address\_0 which is the start point of each new subset in BRAMs 6 and 7 so that the other IPs know that each start starts at what address of the BRAM. base\_address\_0 is forwarded to the Gam\_Interface\_0 IP. The Gam\_Interface\_0 IP holds onto the location of every subset and sends this data over to the Gam\_Imp\_0 IP as it's needed to perform the correlation on that subset. This IP is also responsible for counting the number of pixels are inside the subset in both integer and floating-point format for using in the Gam\_Imp\_0 IP. The Gamma\_Imp IP has not started when the SubsetCoordsMulti IP is finished so the Gam\_Interface IP acts as a buffer and holds onto these values for Gamma until it is ready to start. So, at the end of the process of each IP, its base\_address\_0, num\_pxl\_Int\_0, and num\_pxl\_FP\_0 are forwarded to the Gam\_Interface IP to be saved in an internal register.