reset==1 or i_valid==0

row!=1 and column!=15 and flag = 0

compute_flag==1 and column==15 and row!=15

column!=15 and row!=15 and compute_flag==0

RST

row=0
column=0
o_done=0
o_data=0
p = 0
flag = 0
compute_flag=0

minuend (0 to 15)=0
subtrahend (0 to 15)=0
addend (0 to 15)=0
stage =001
if(i_valid==1) {
  minuend[column]=i_data
  column++
}

reset==0 and i_valid==1

Init

if(row==0 and i_valid==1) {
  minuend[column]=i_data
  if(column==15) {
    column=0
    row=1
  } else {
    column++
  }
} else if(row!=0 and i_valid==1) {
  subtrahend[column]=i_data
  if(column==15) {
    flag=1
  } else {
    column++
  }
  flag=0
}

flag==1 and row==1 and column==15

Slt

column=0
row++
switch(stage) {
  case 001:
    stage=010

  case 010:
    stage=010
  minuend =subtrahend
  subtrahend =addend
  default:

}

Compute

if(i_valid==1) {
  addend[column]=i_data

  p = minuend[column] - subtrahend[column]
    + addend[column]

  if p≥0 {
    o_data++;
  }
  if(column==15) {
    compute_flag=1
  } else {
    column++
    compute_flag=0
  }
}

complete

o_done=1

i_valid==1 or reset==1

i_valid!=1 and reset!=1

column==15 and row==15 and compute_flag==1

— Delete signals a, b and c, introduce new signals flag and compute_flag

— In Rst, when data is valid, store it into minuend

— In Init, perform actions only when data is valid, and switch rows when flag is 1

— In Slt, change stage number when first time entering the Slt; otherwise, set previous substrahend to current minuend, set pervious addend to current substrahend
   (only)

— In compute, perform actions only when data is valid, and set compute_flag to 1 when column is 15