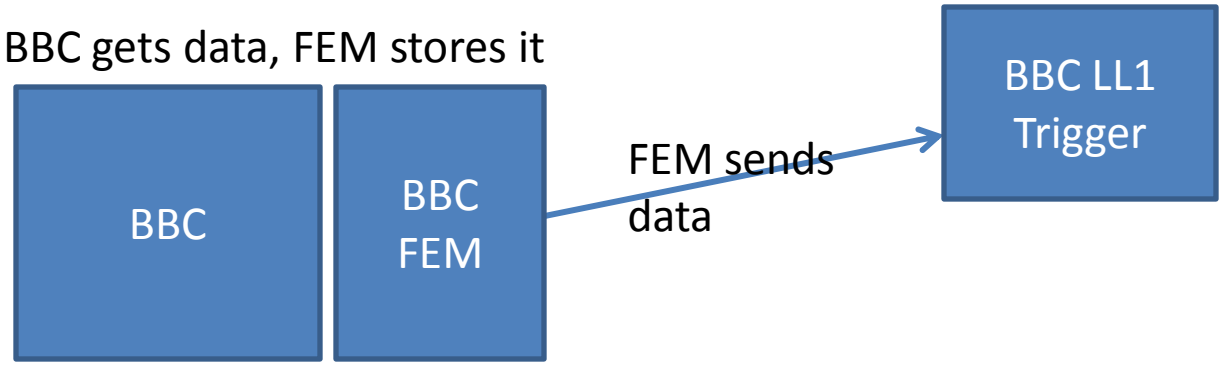
LL1 Trigger Overview

Josh Perry
ISU
Spinfest 2013

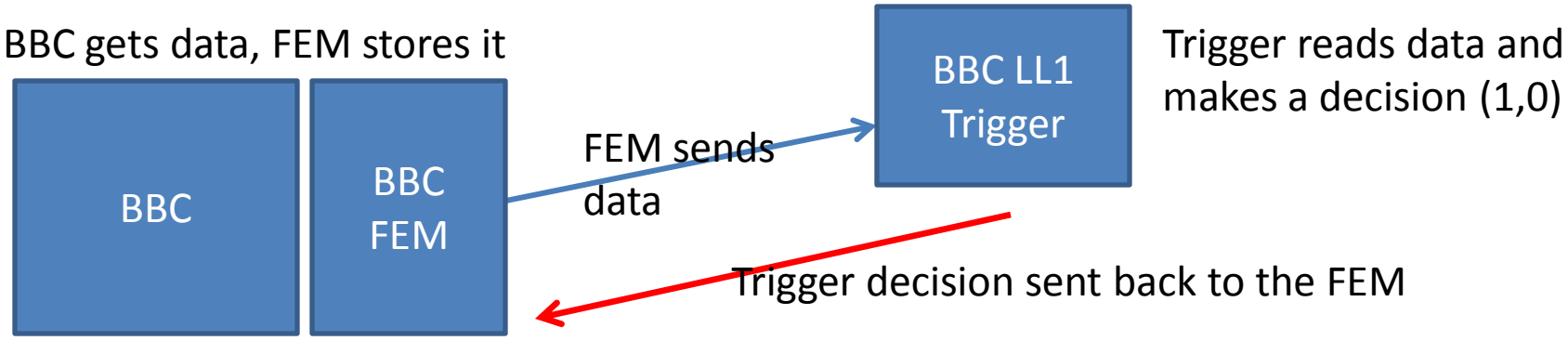
Trigger theory

- Select out rare events
  - Note: We can only select out events which MIGHT be rare/good events!
- Suppress background and uninteresting events
- We have a given DAQ rate (6kHz)
  - We want to fill this DAQ rate without missing out on good events!

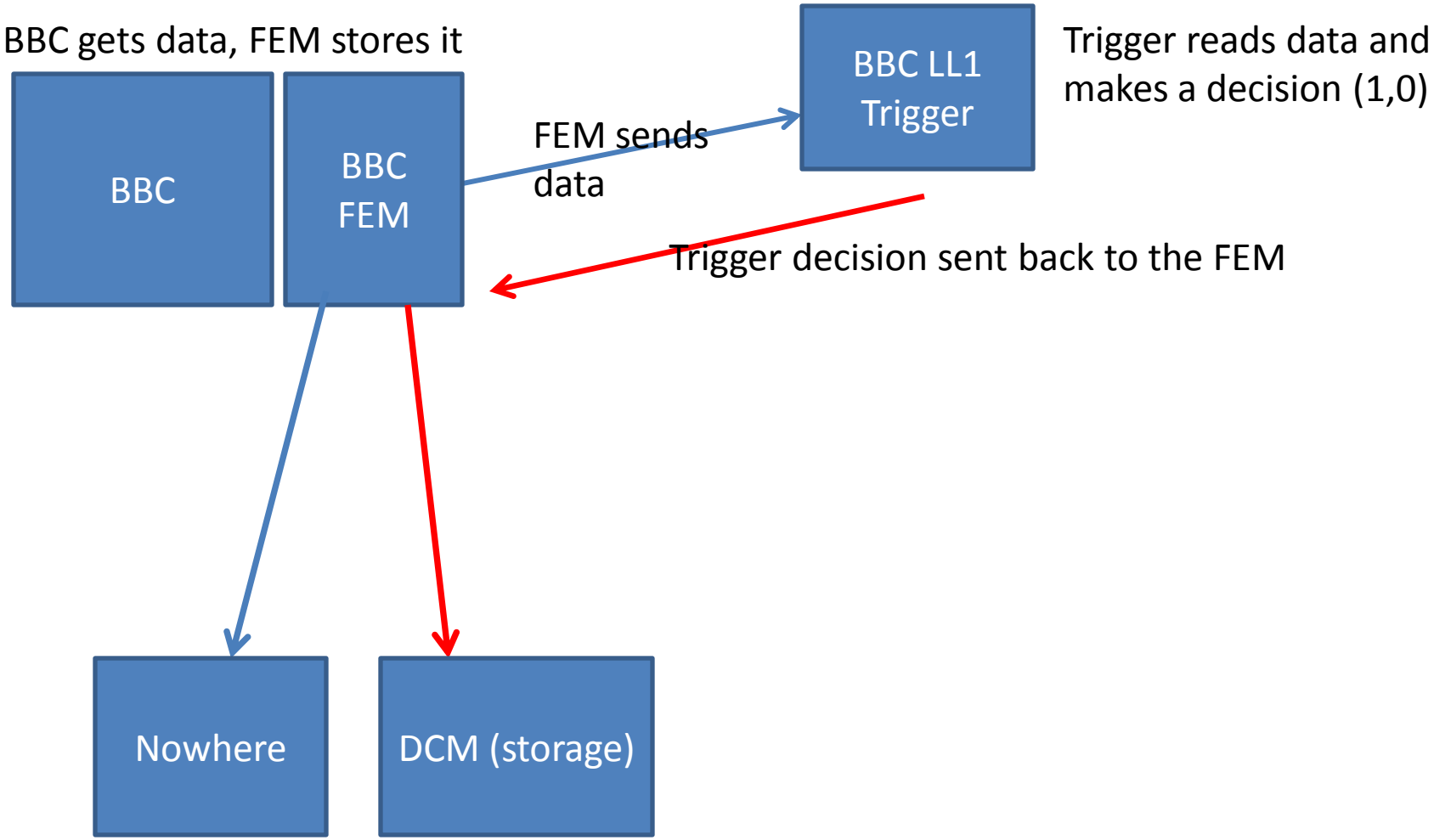
One Detector Triggering (Instant)



One Detector Triggering (Instant)

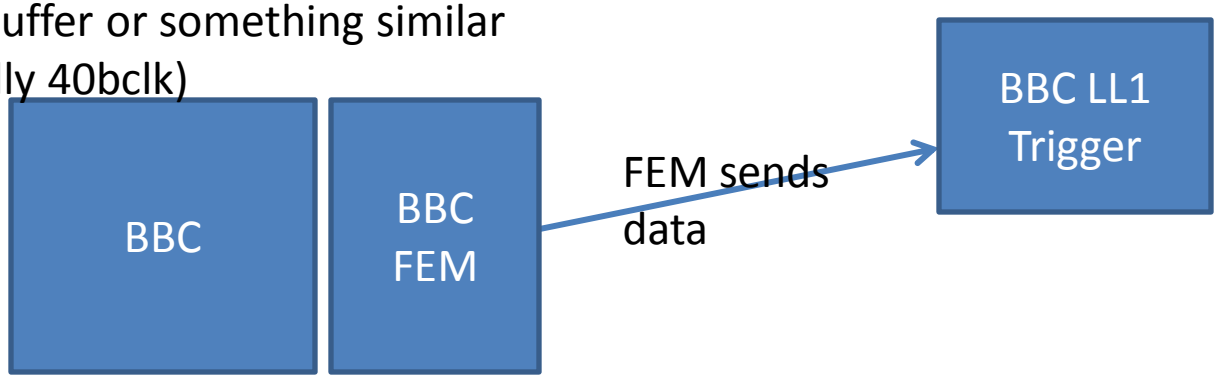


One Detector Triggering (Instant)



One Detector Triggering (Non-Instant)

BBC gets data, FEM stores it in a ring-buffer or something similar (usually 40bclk)

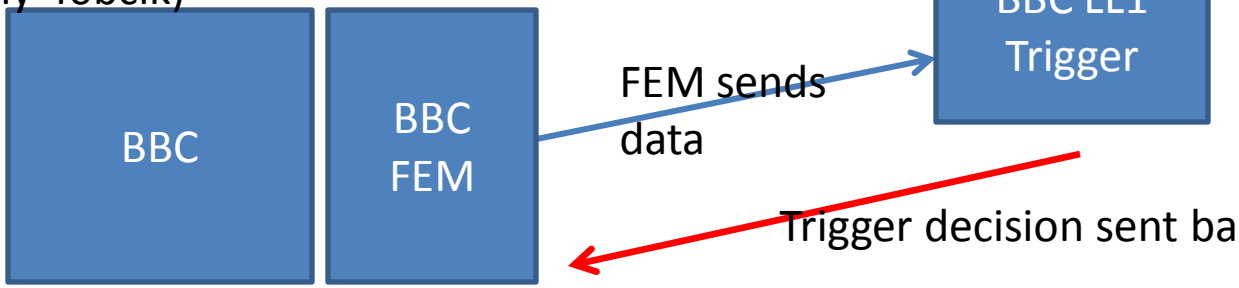


The BBC FEM does ADC on the FEM itself, so it has a long delay (many beam blocks) before the data is sent to LL1; Same with PC, TEC, DC, BBC, ZDC, MuTr (new), MuID, old MPC (using MuID FEMs)

Note: the AMU/ADC32 based FEMs digitize AFTER the accept signal from LL1: MuTr (old), EMC, RICH, ToF; this means the delay is shifted

One Detector Triggering (Non-Instant)

BBC gets data, FEM stores it in a ring-buffer or something similar (usually 40bclk)



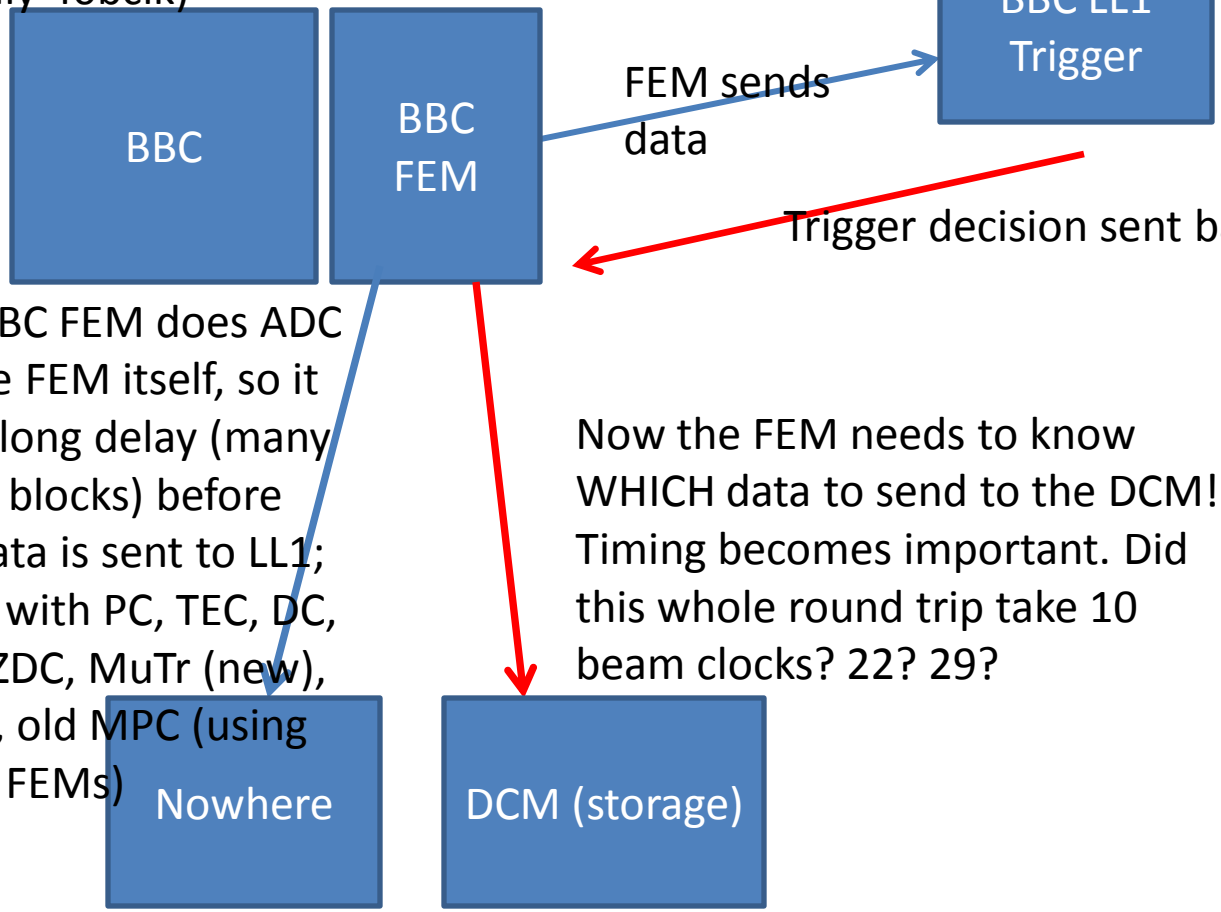
Trigger reads data and makes a decision (1,0). Note: the LL1s must be output a decision EVERY beam clock! If it doesn't, data is lost!

The BBC FEM does ADC on the FEM itself, so it has a long delay (many beam blocks) before the data is sent to LL1; Same with PC, TEC, DC, BBC, ZDC, MuTr (new), MuID, old MPC (using MuID FEMs)

Note: the AMU/ADC32 based FEMs digitize AFTER the accept signal from LL1: MuTr (old), EMC, RICH, ToF; this means the delay is shifted

One Detector Triggering (Non-Instant)

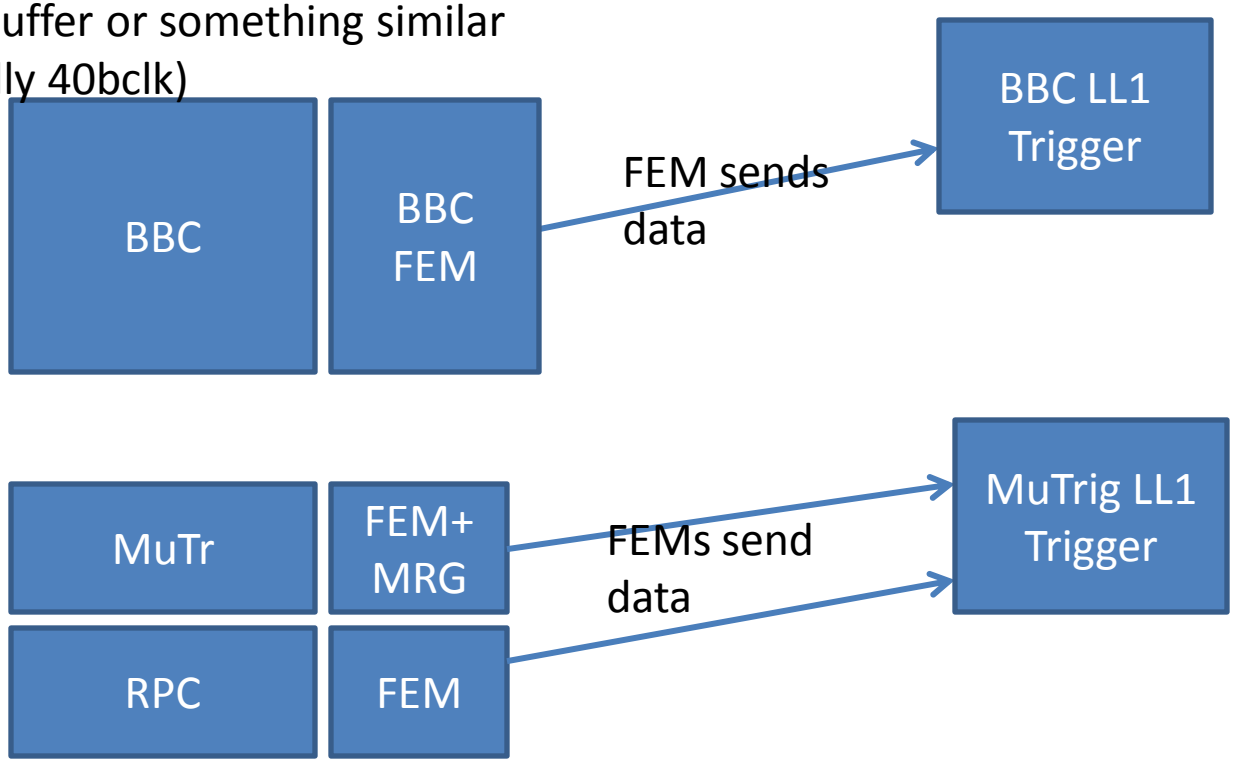
BBC gets data, FEM stores it in a ring-buffer or something similar (usually 40bclk)



Note: the AMU/ADC32 based FEMs digitize AFTER the accept signal from LL1: MuTr (old), EMC, RICH, ToF; this means the delay is shifted

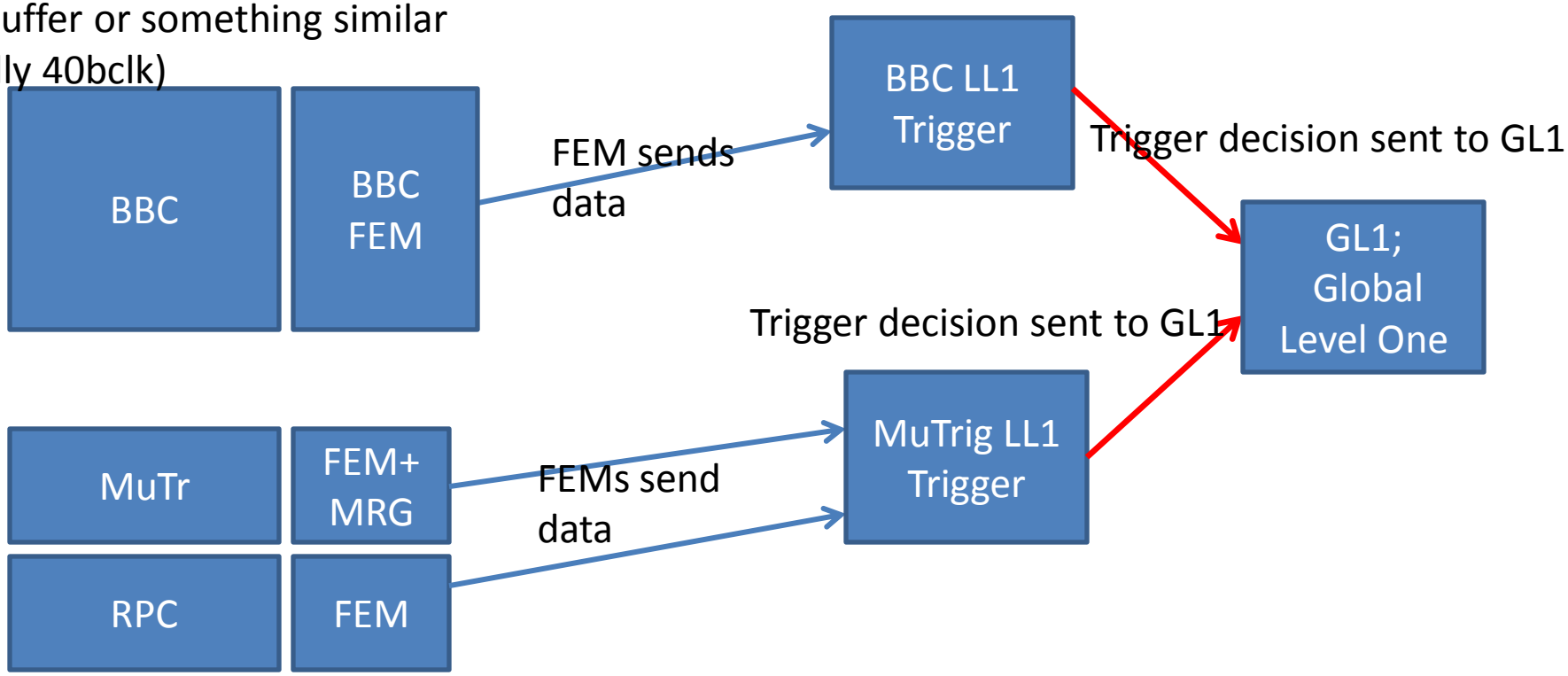
Multiple Detector Triggering

BBC gets data, FEM stores it in a ring-buffer or something similar (usually 40bclk)



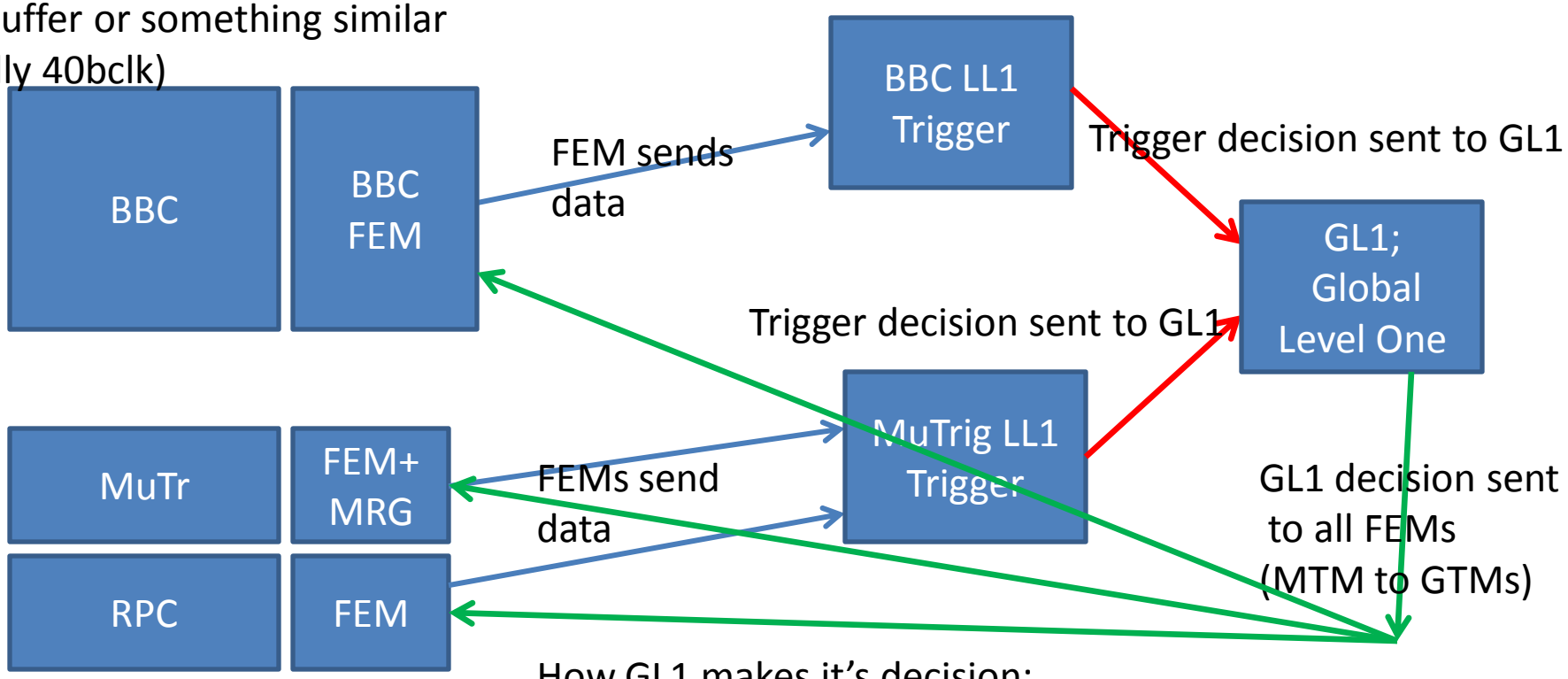
Multiple Detector Triggering

BBC gets data, FEM stores it in a ring-buffer or something similar (usually 40bclk)



Multiple Detector Triggering

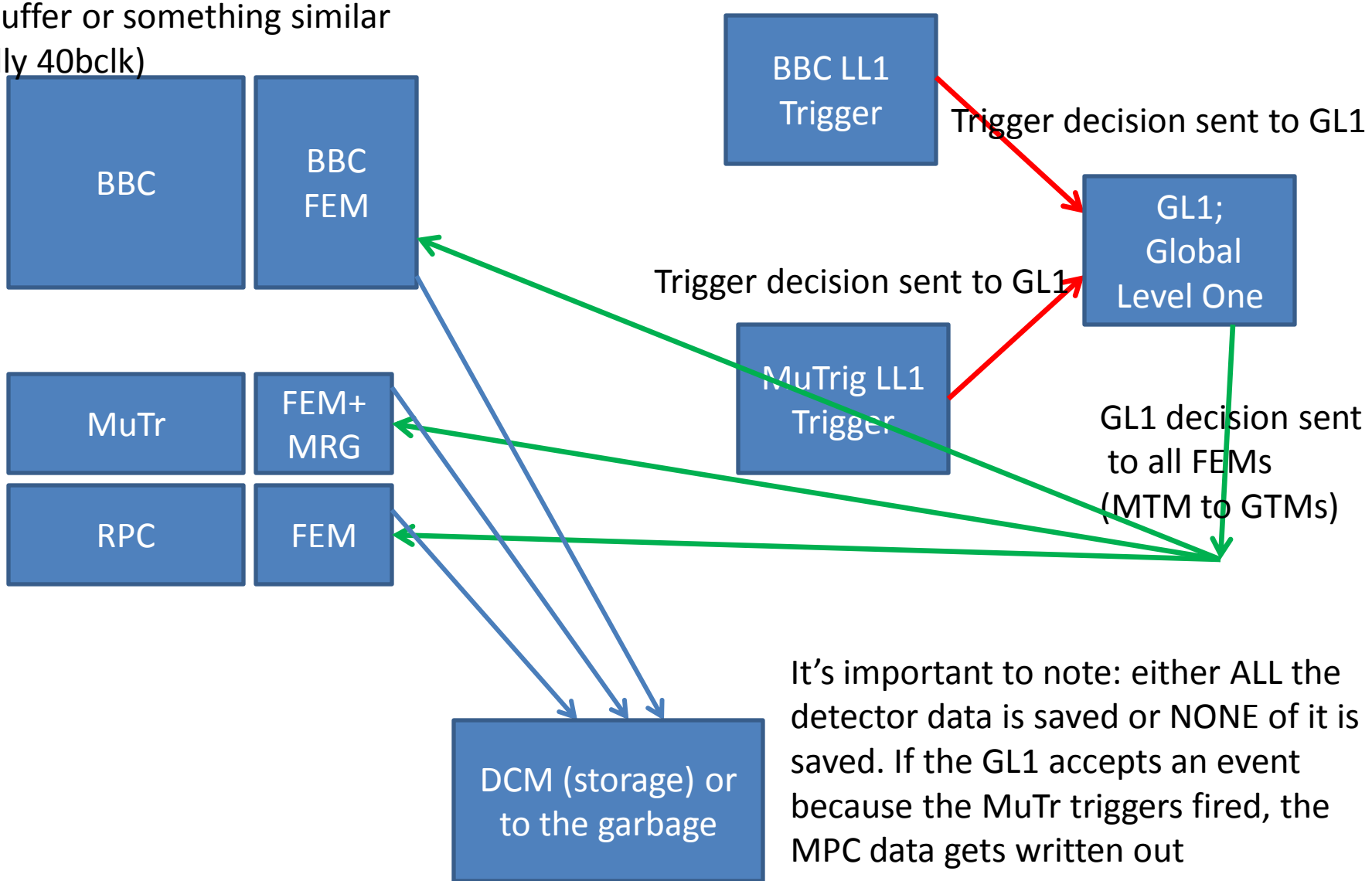
BBC gets data, FEM stores it in a ring-buffer or something similar (usually 40bclk)



How GL1 makes it's decision:
Prescales. Prescales define how much weight we put on a specific mix of trigger's decision. If a detector is triggering too much (BBC(noVtx), for example), then the GL1 will ignore its pleas for a global accept. However, something like a SG1&BBC&RPC1&RPC3 trigger will never be ignored (prescale 0).

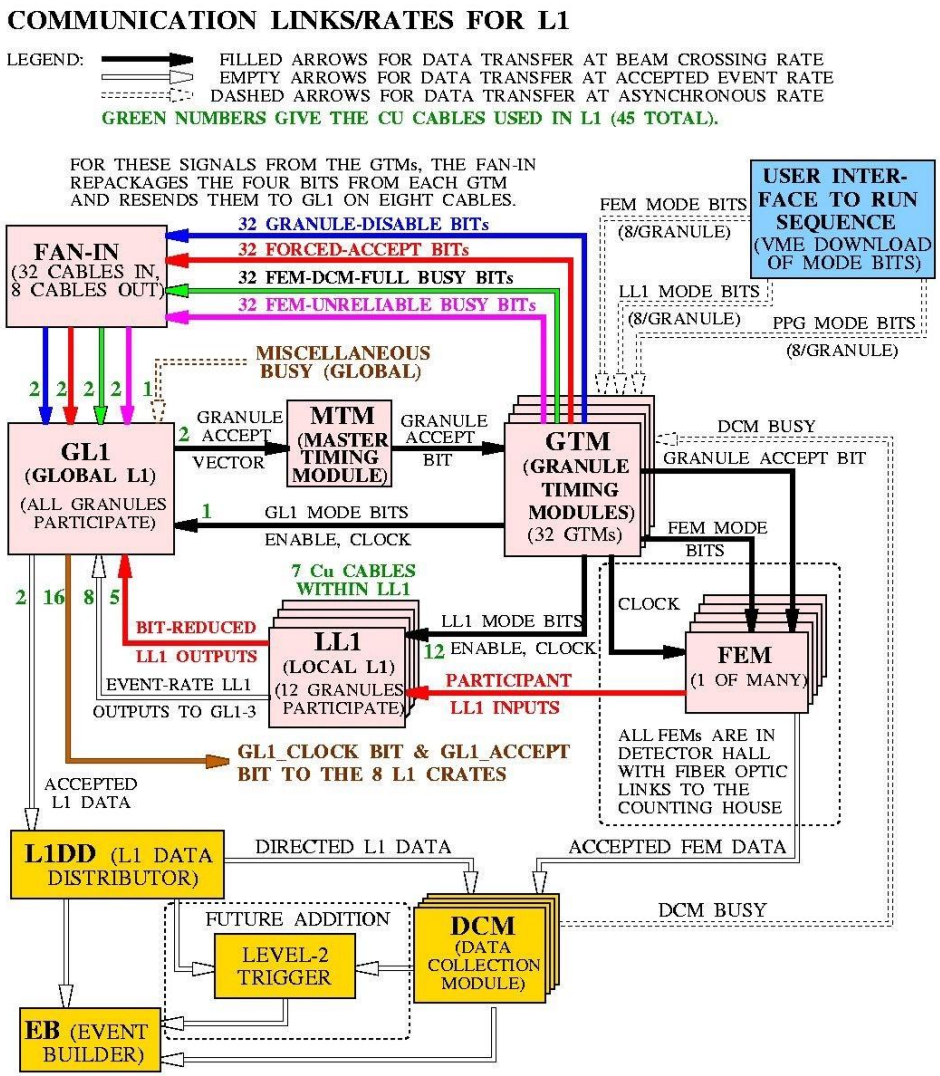
Multiple Detector Triggering

BBC gets data, FEM stores it in a ring-buffer or something similar (usually 40bclk)



It's important to note: either ALL the detector data is saved or NONE of it is saved. If the GL1 accepts an event because the MuTr triggers fired, the MPC data gets written out

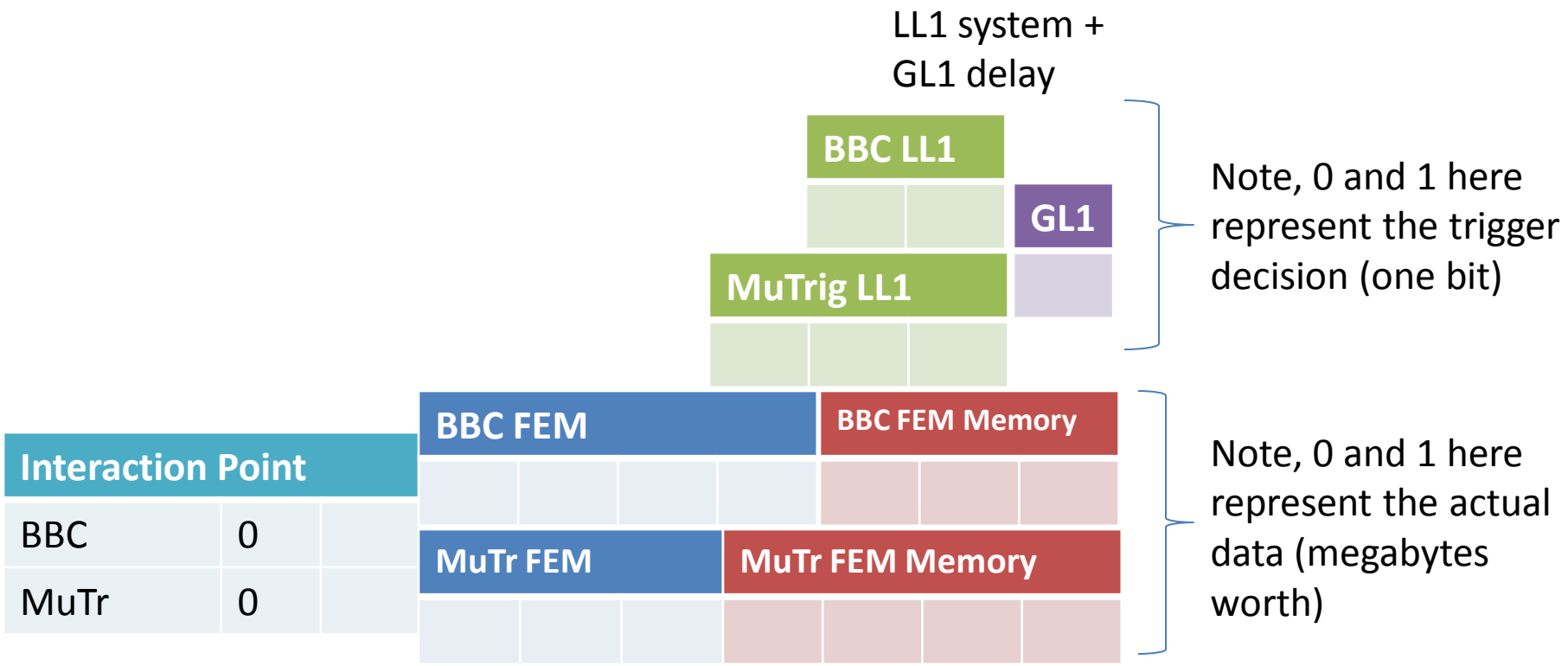
A simplified view of the system



Animation

- This is a crappy “animation” I made like three years ago and never showed anyone.

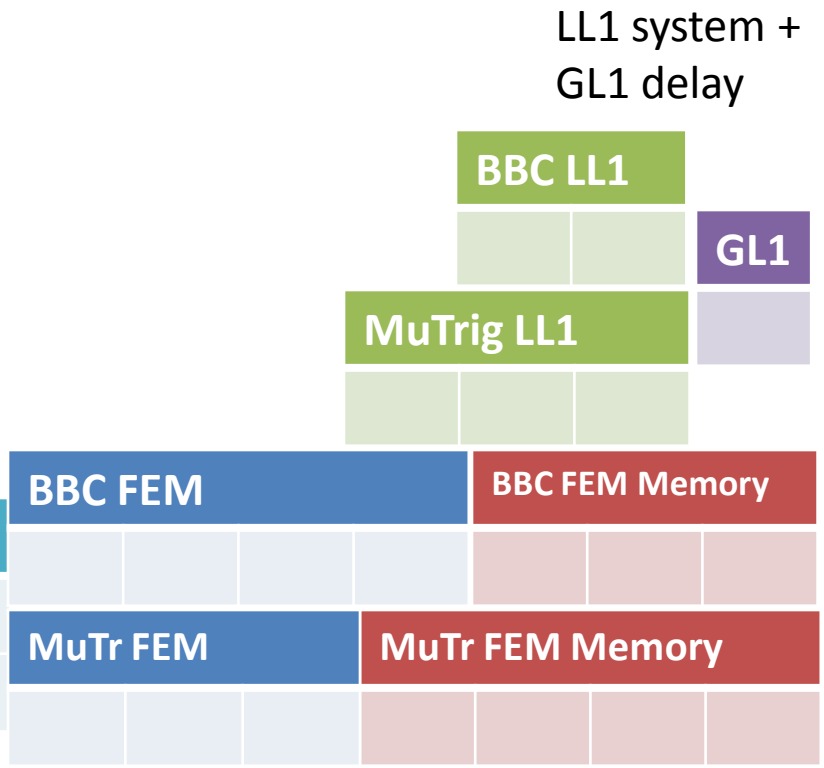
If things are in time



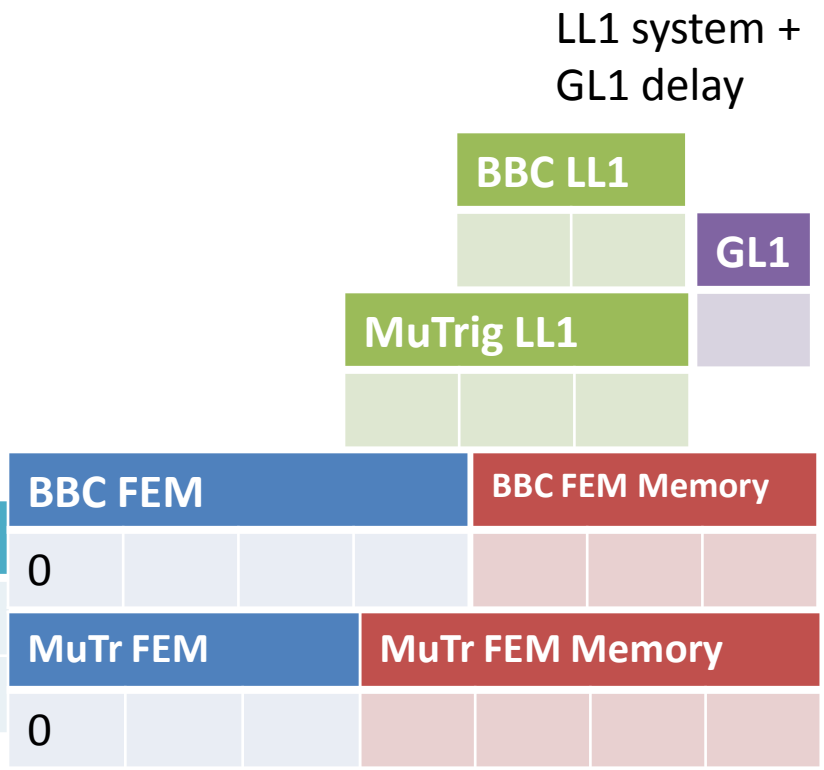
Interaction Point		
BBC	0	
MuTr	0	

If things are in time

Interaction Point		
BBC	0	0
MuTr	0	0

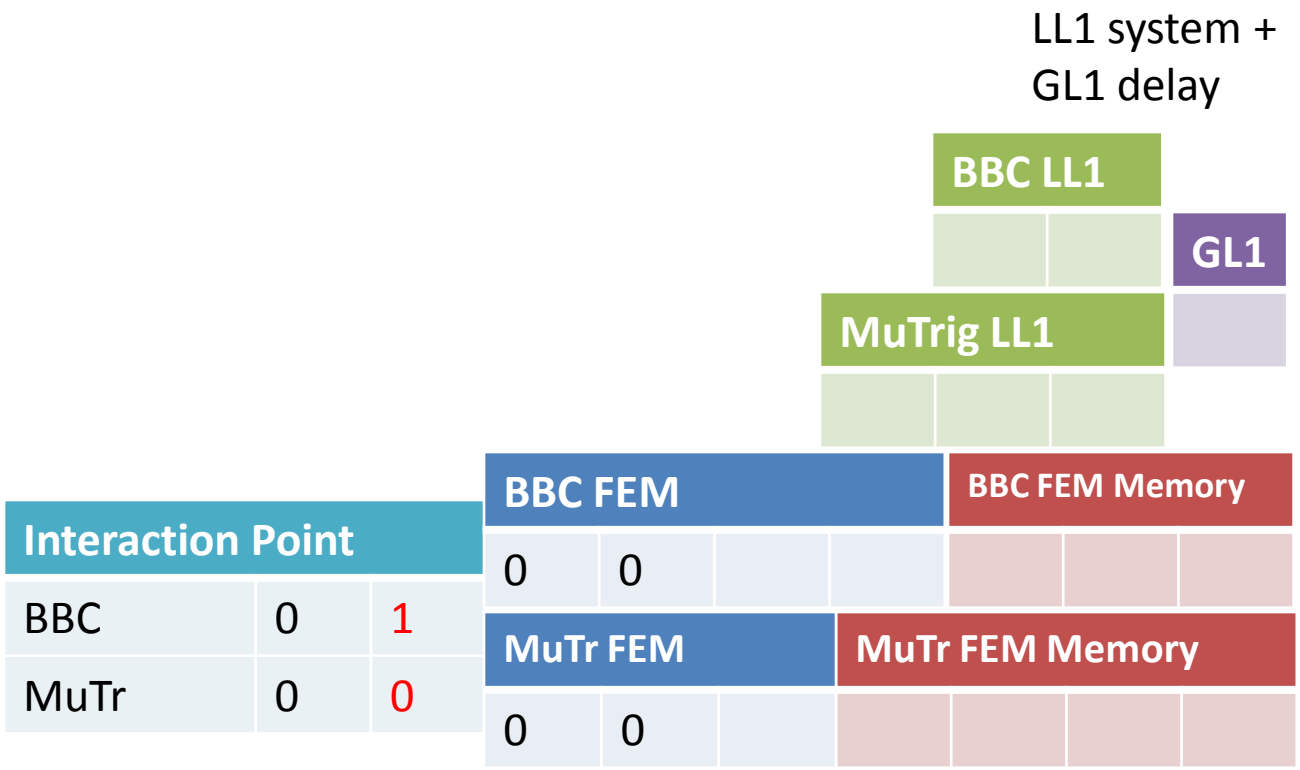


If things are in time



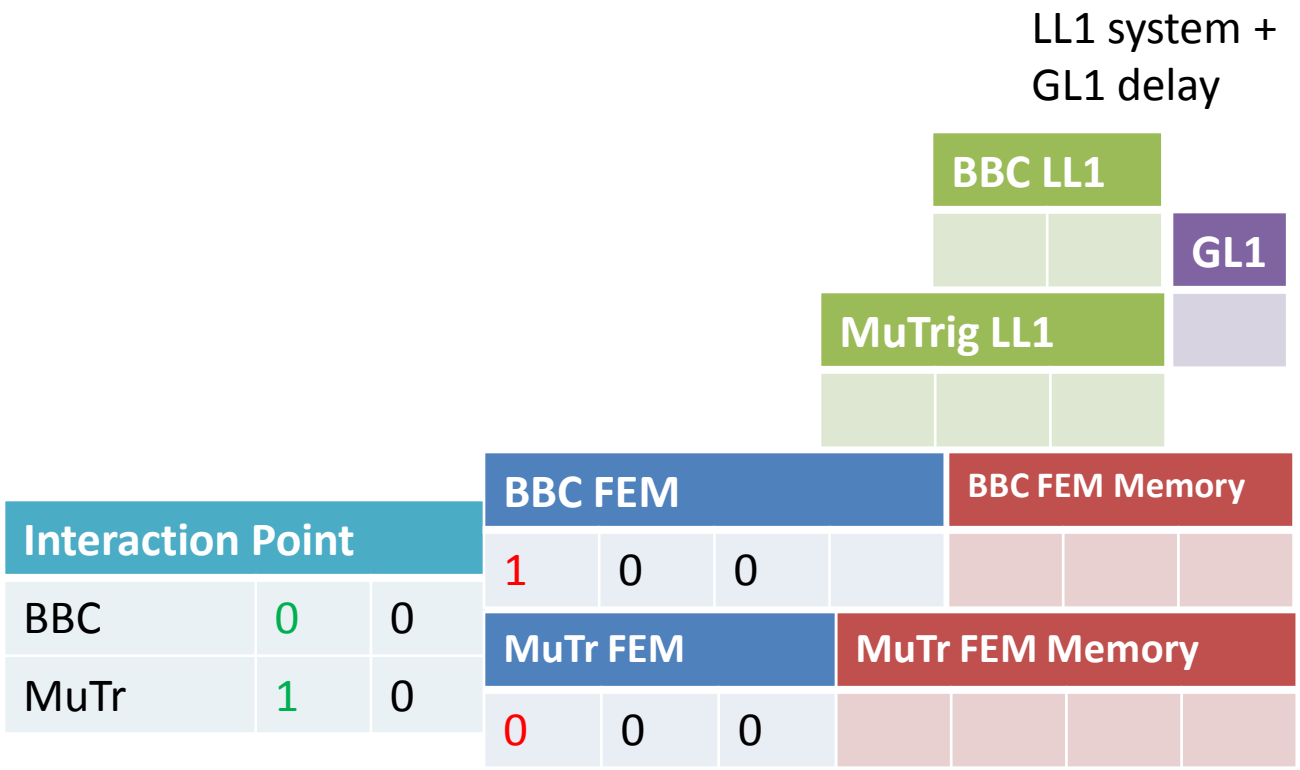
Interaction Point		
BBC	1	0
MuTr	0	0

If things are in time



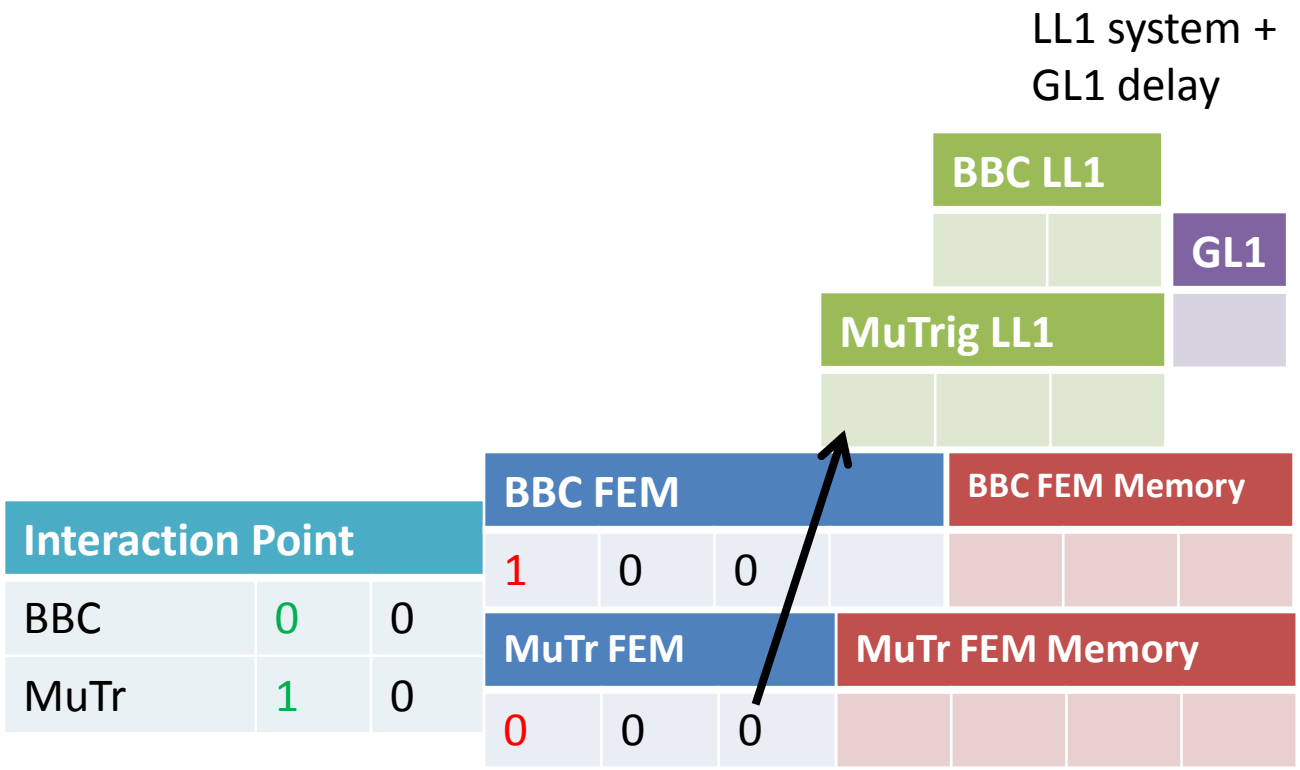
Interaction Point		
BBC	0	1
MuTr	0	0

If things are in time



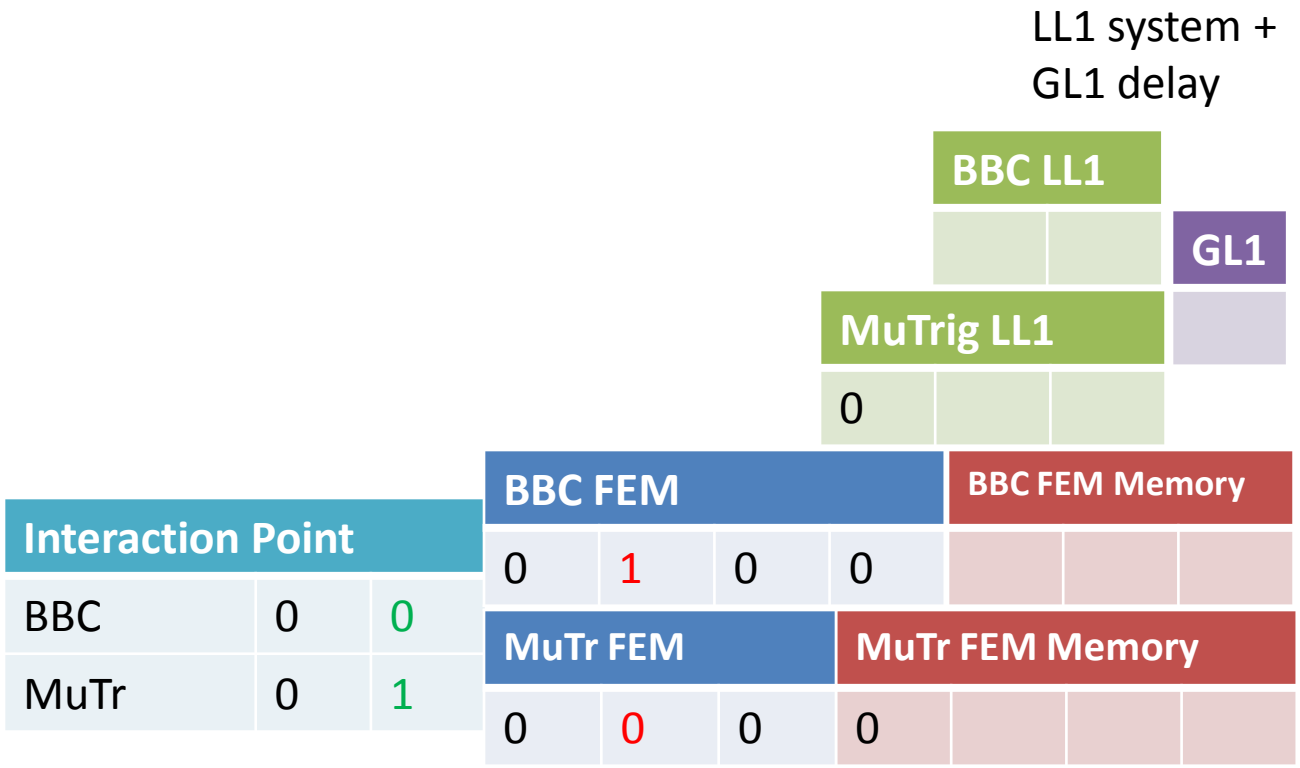
Interaction Point		
BBC	0	0
MuTr	1	0

If things are in time

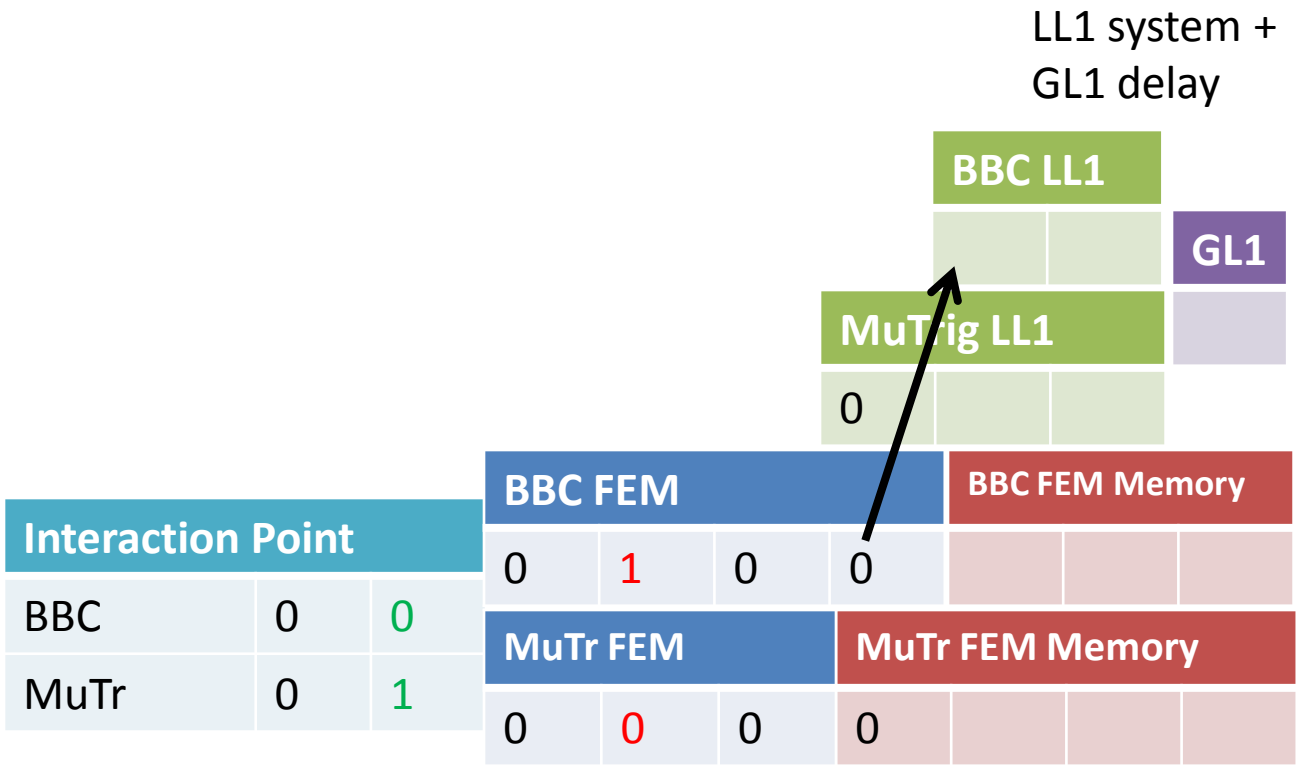


Interaction Point		
BBC	0	0
MuTr	1	0

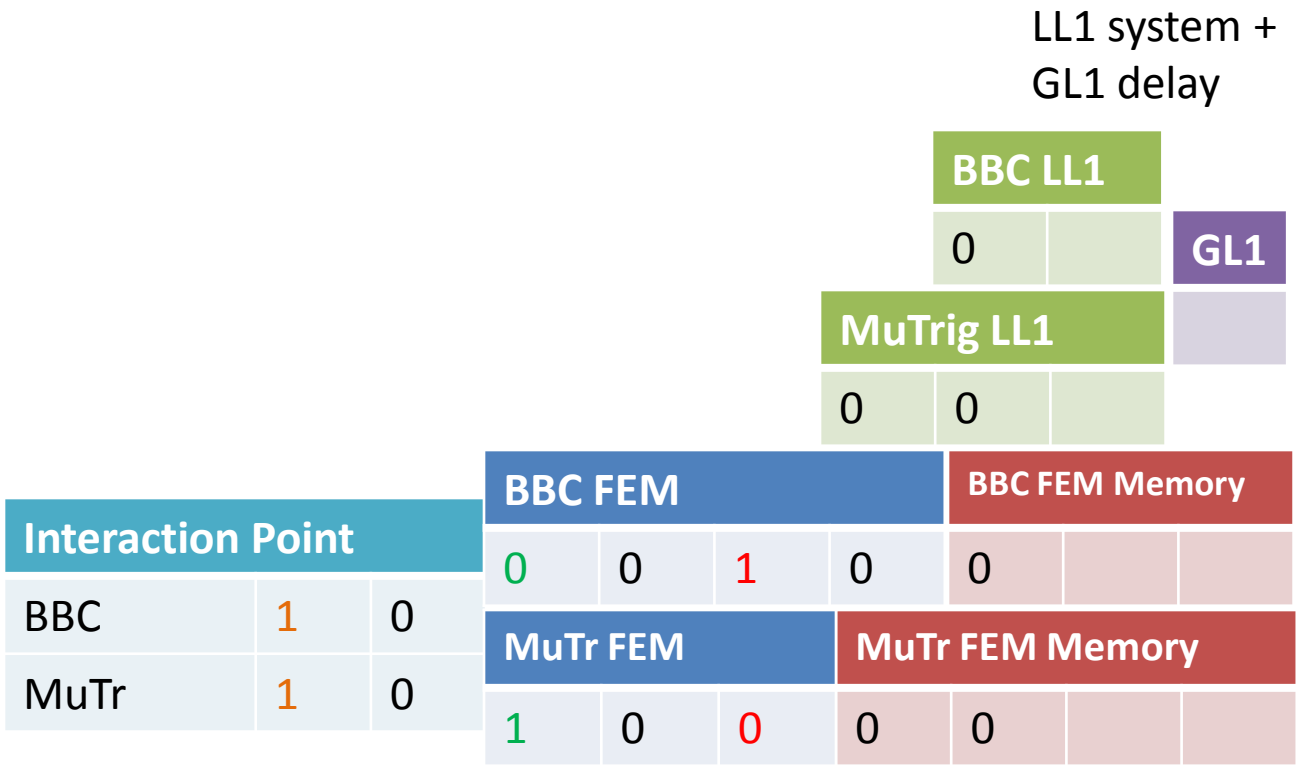
If things are in time



If things are in time

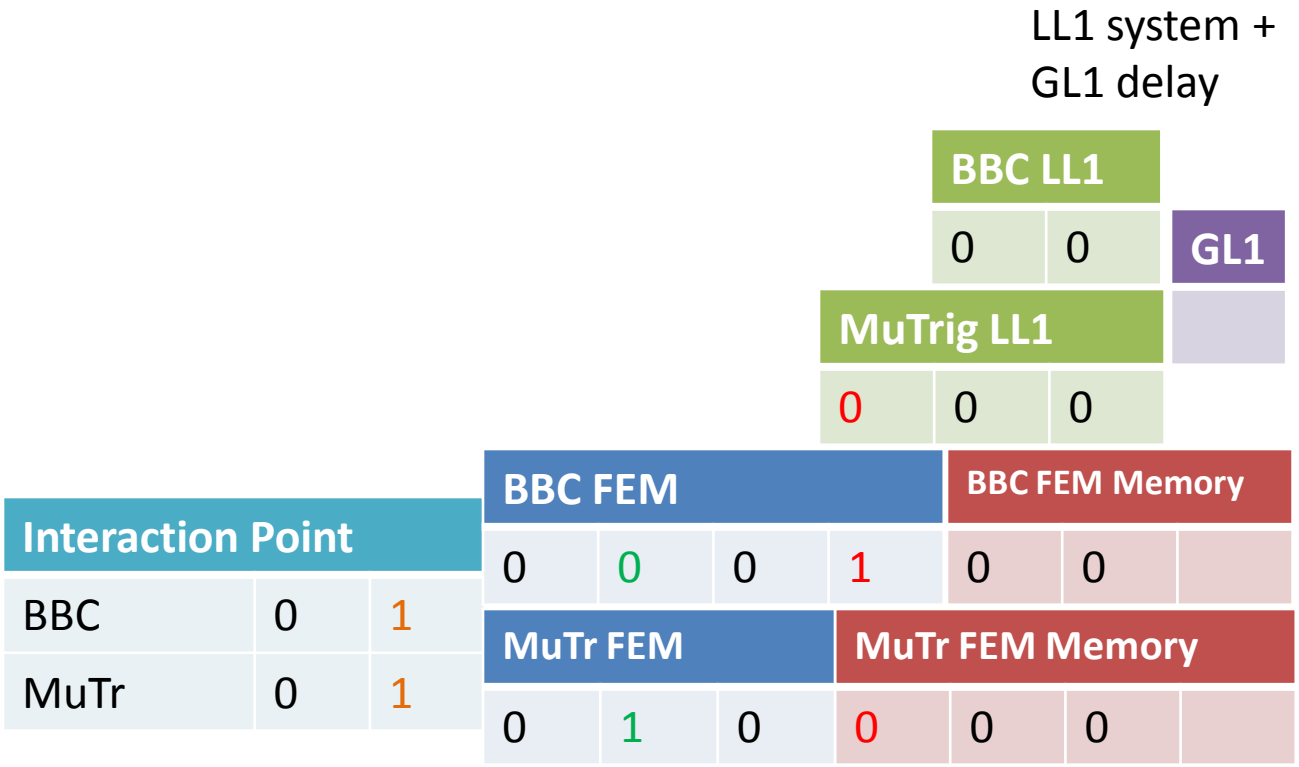


If things are in time

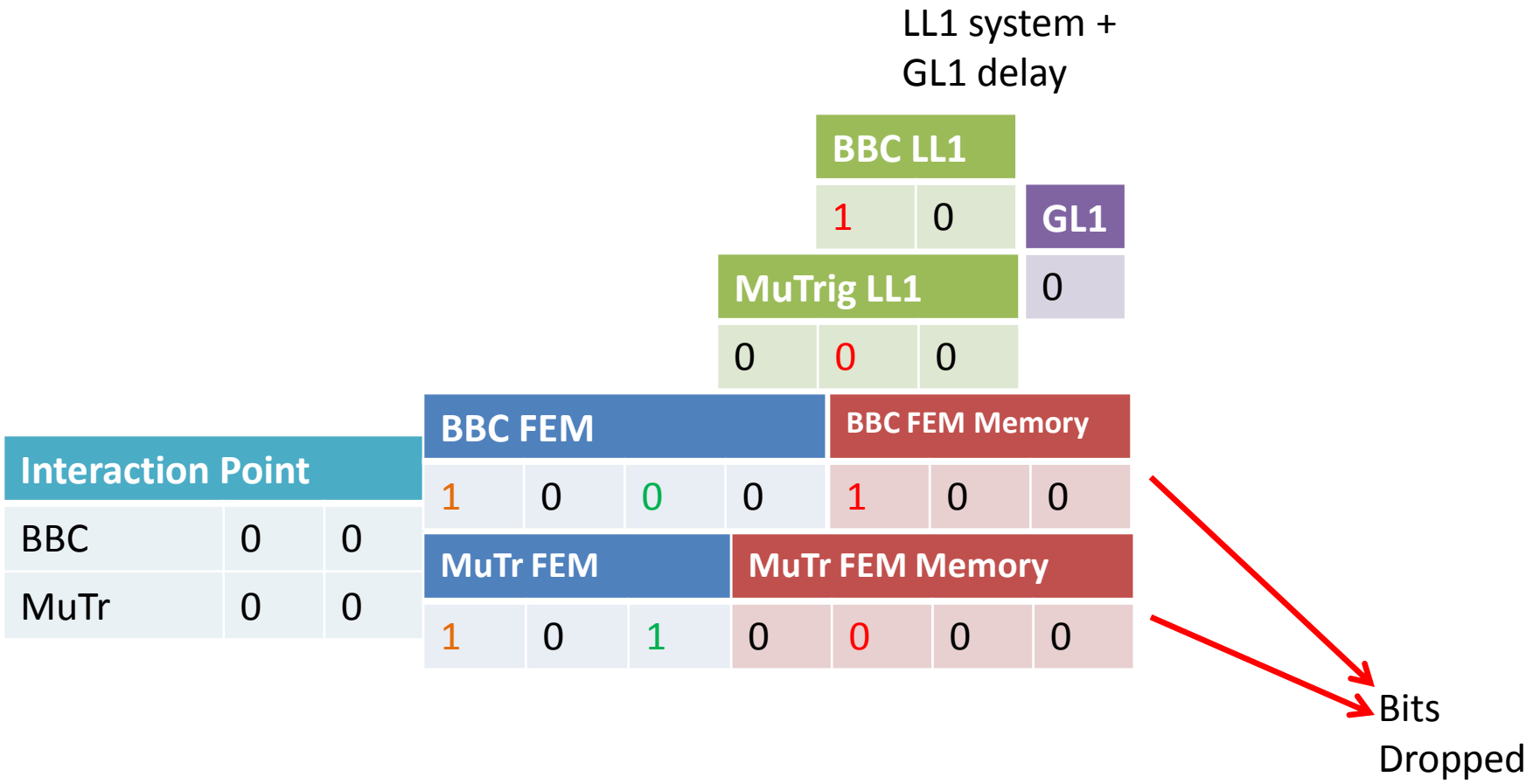


If things are in time

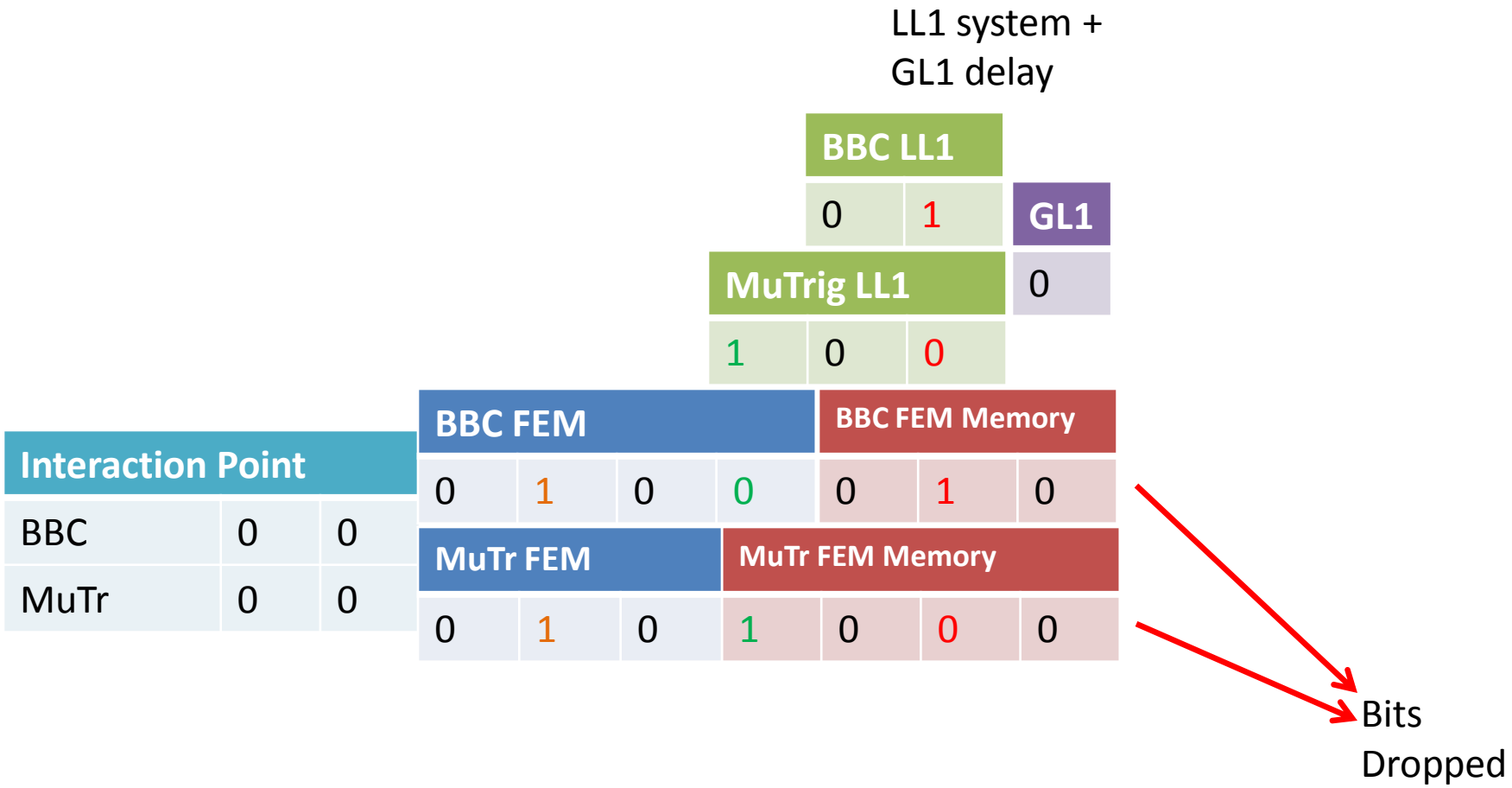
LL1 system +
GL1 delay



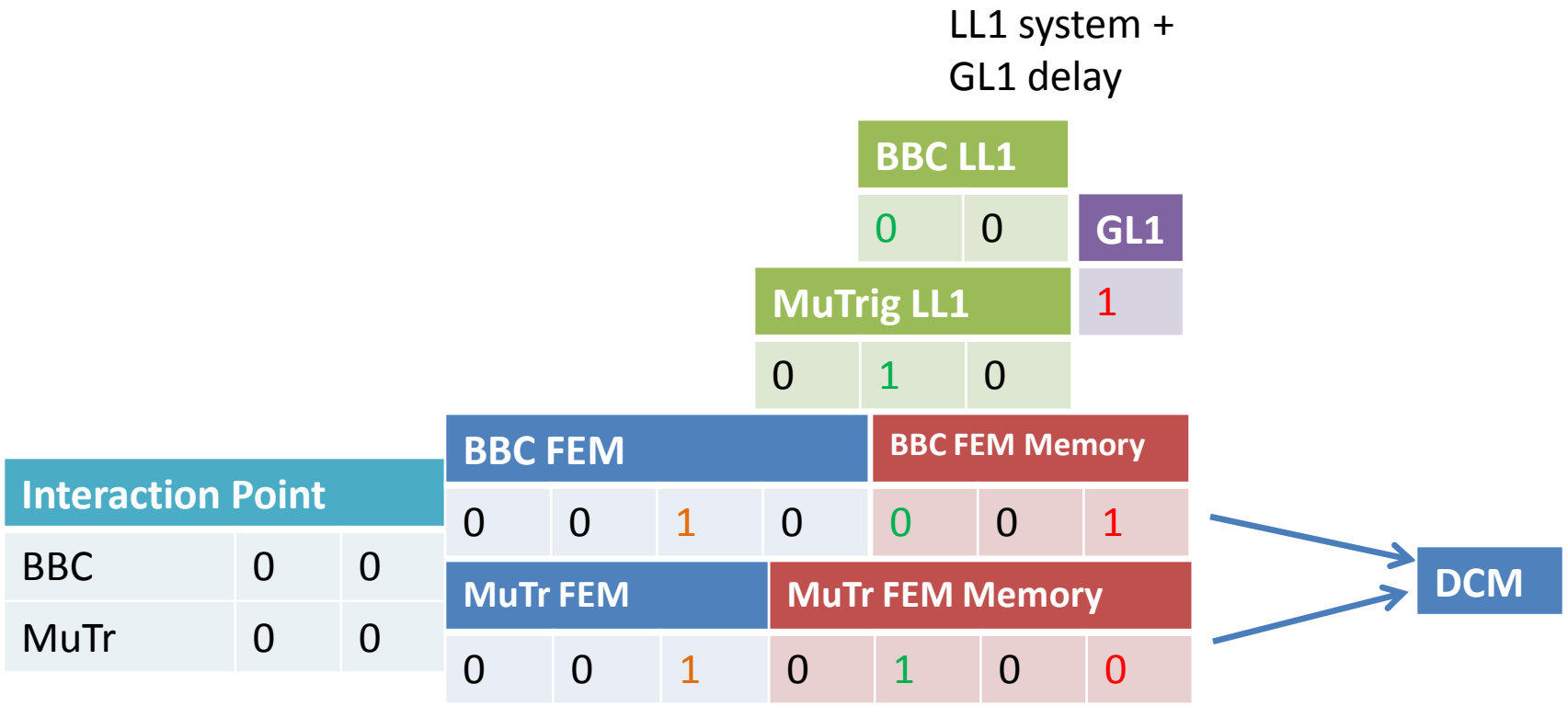
If things are in time



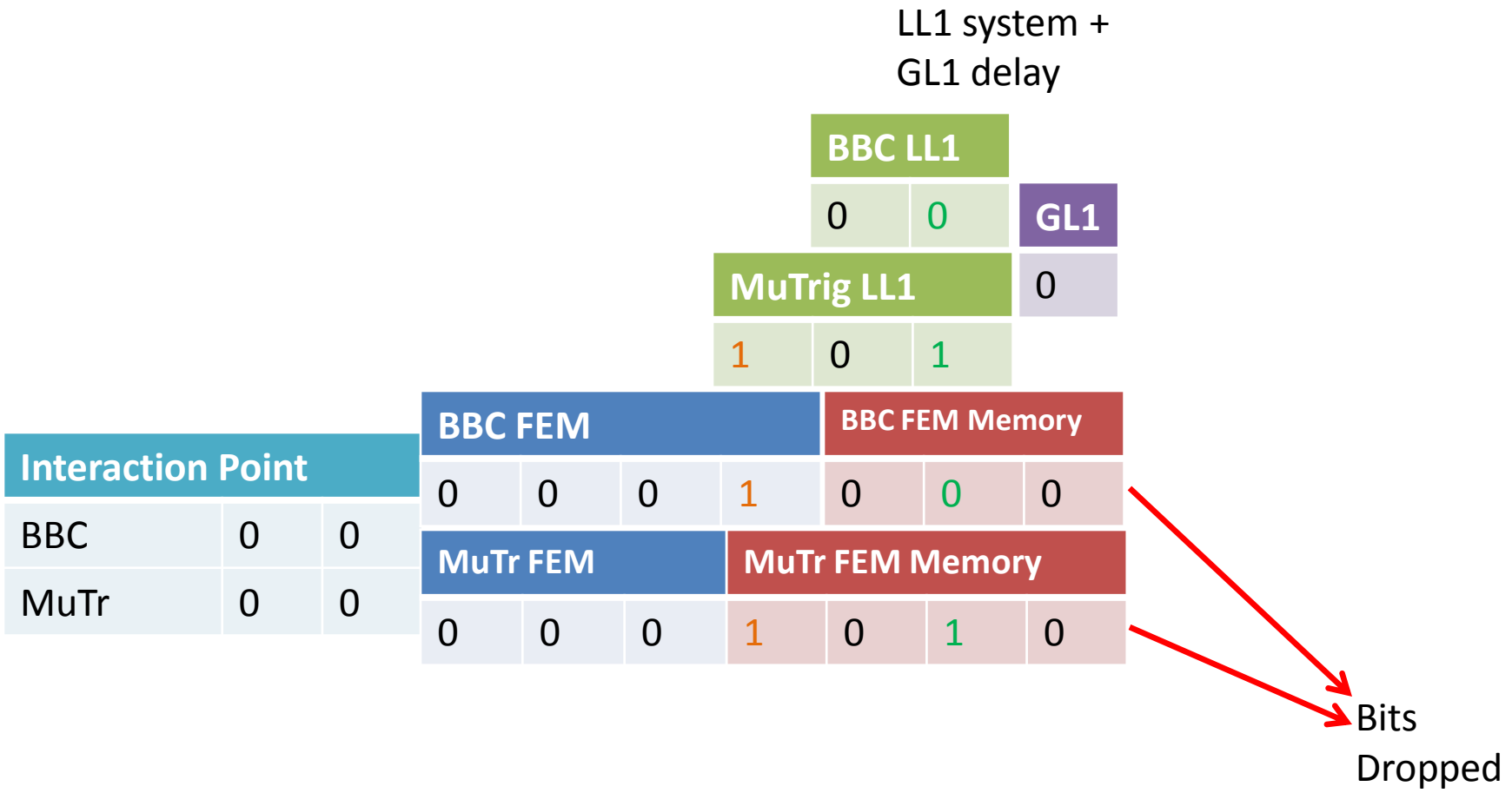
If things are in time



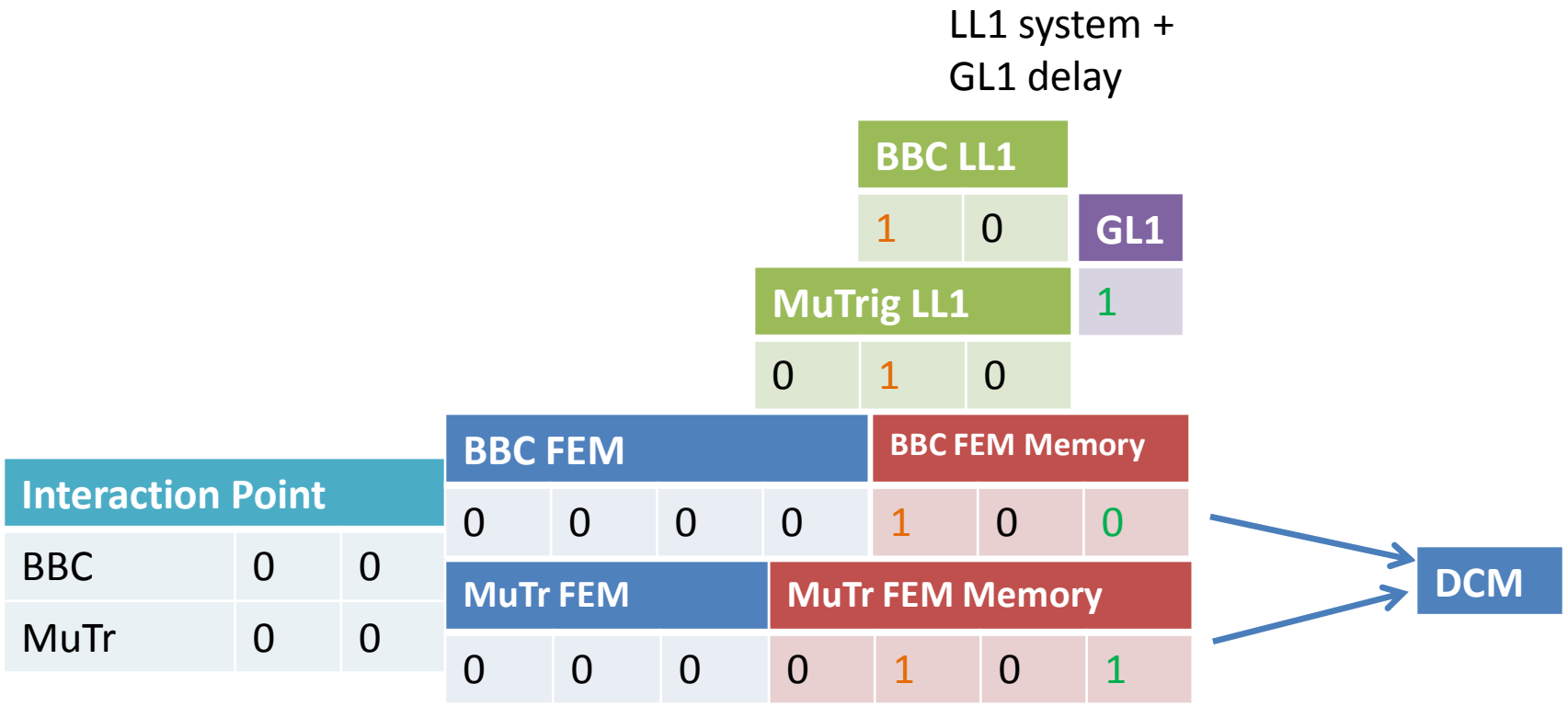
If things are in time



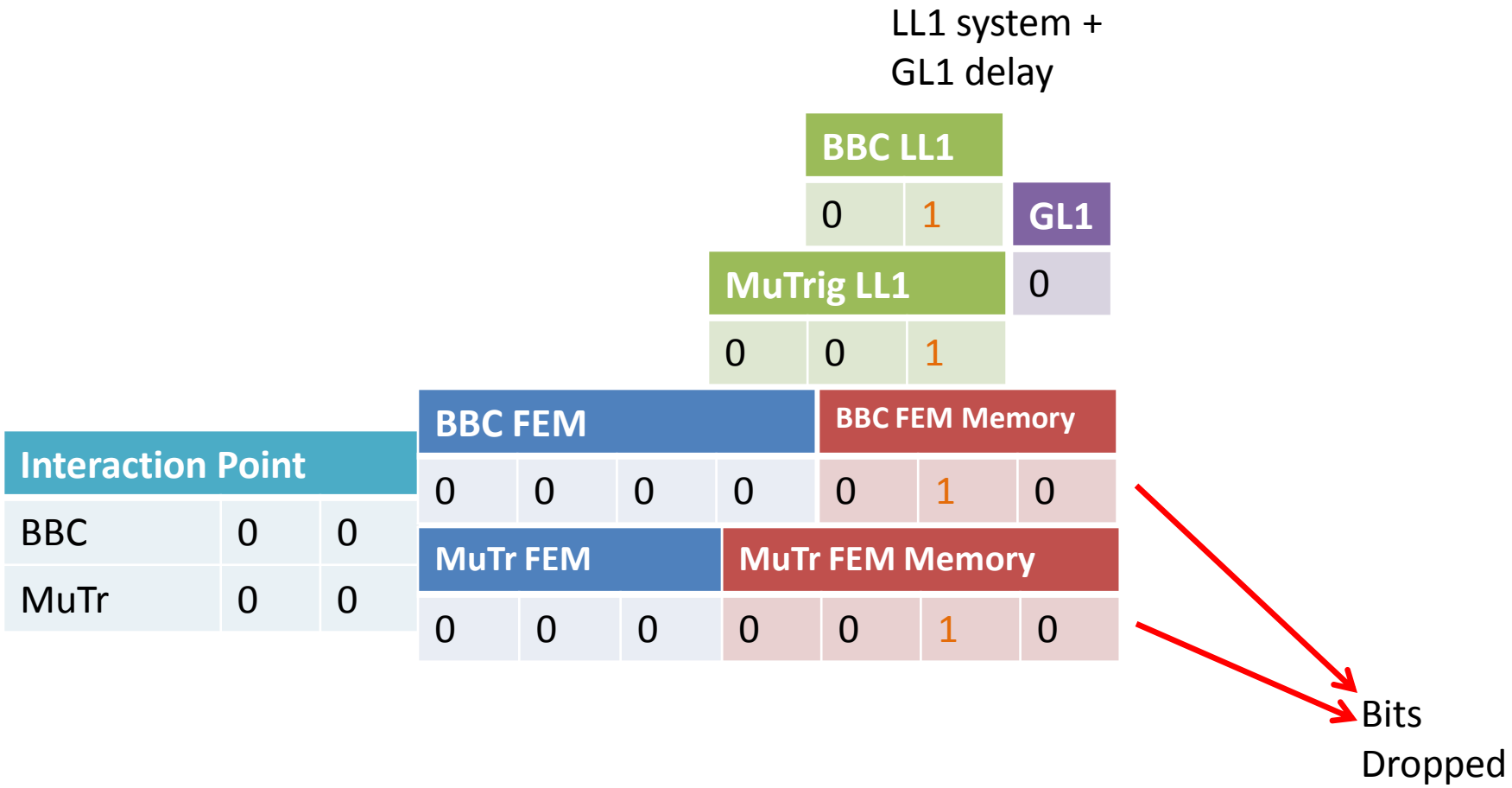
If things are in time



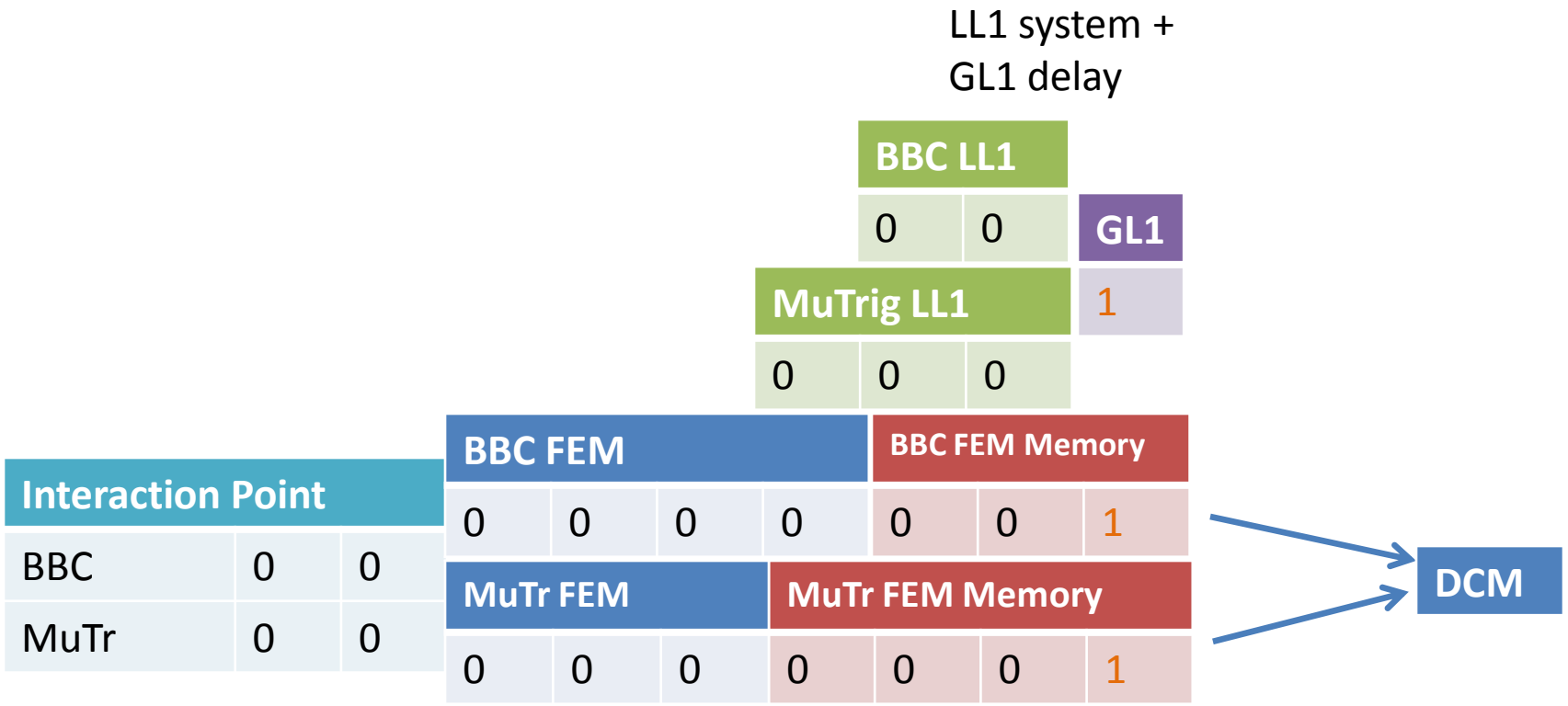
If things are in time



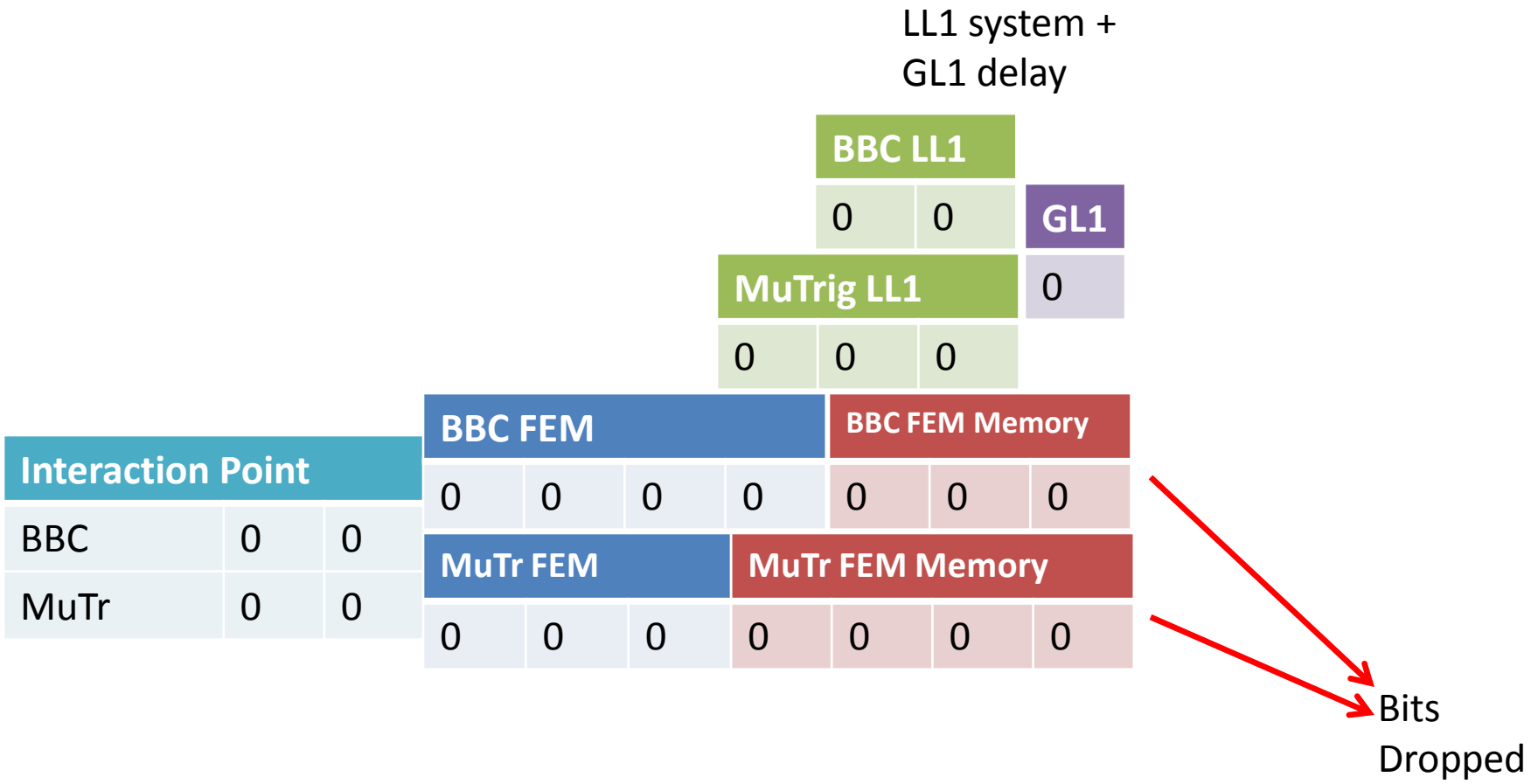
If things are in time



If things are in time

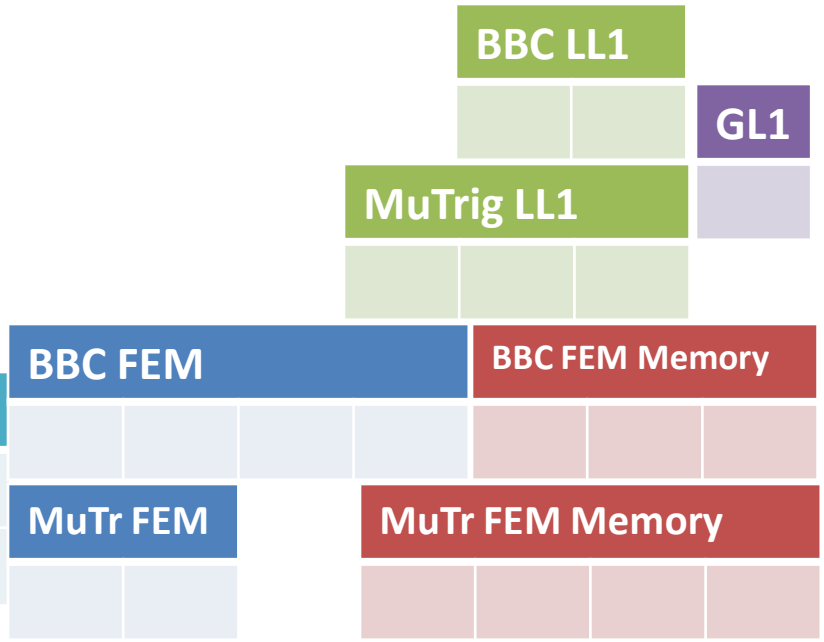


If things are in time



If things are NOT in time

LL1 system +
GL1 delay

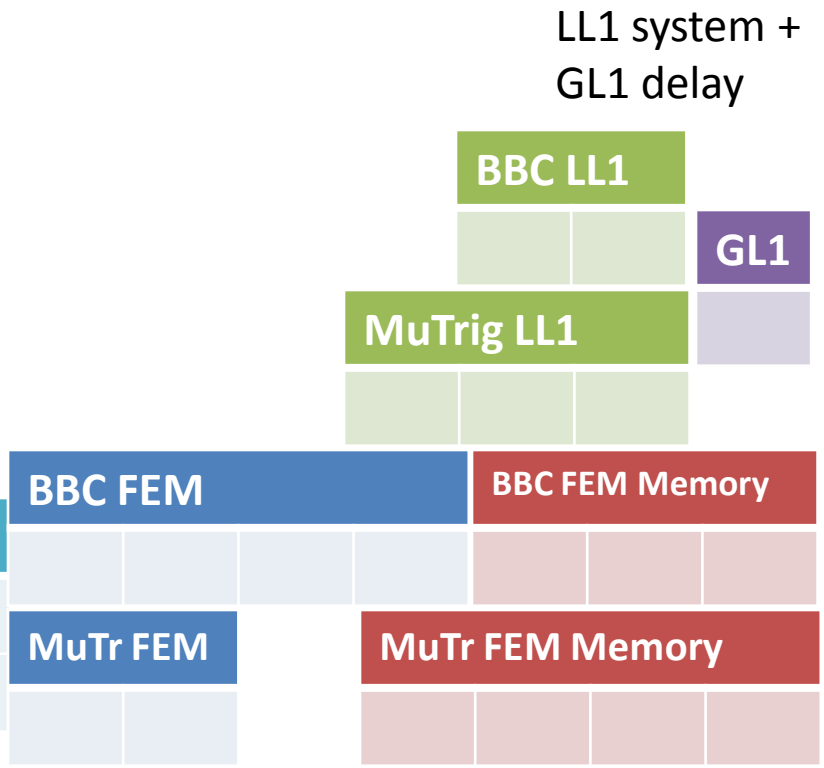


Interaction Point		
BBC	0	
MuTr	0	

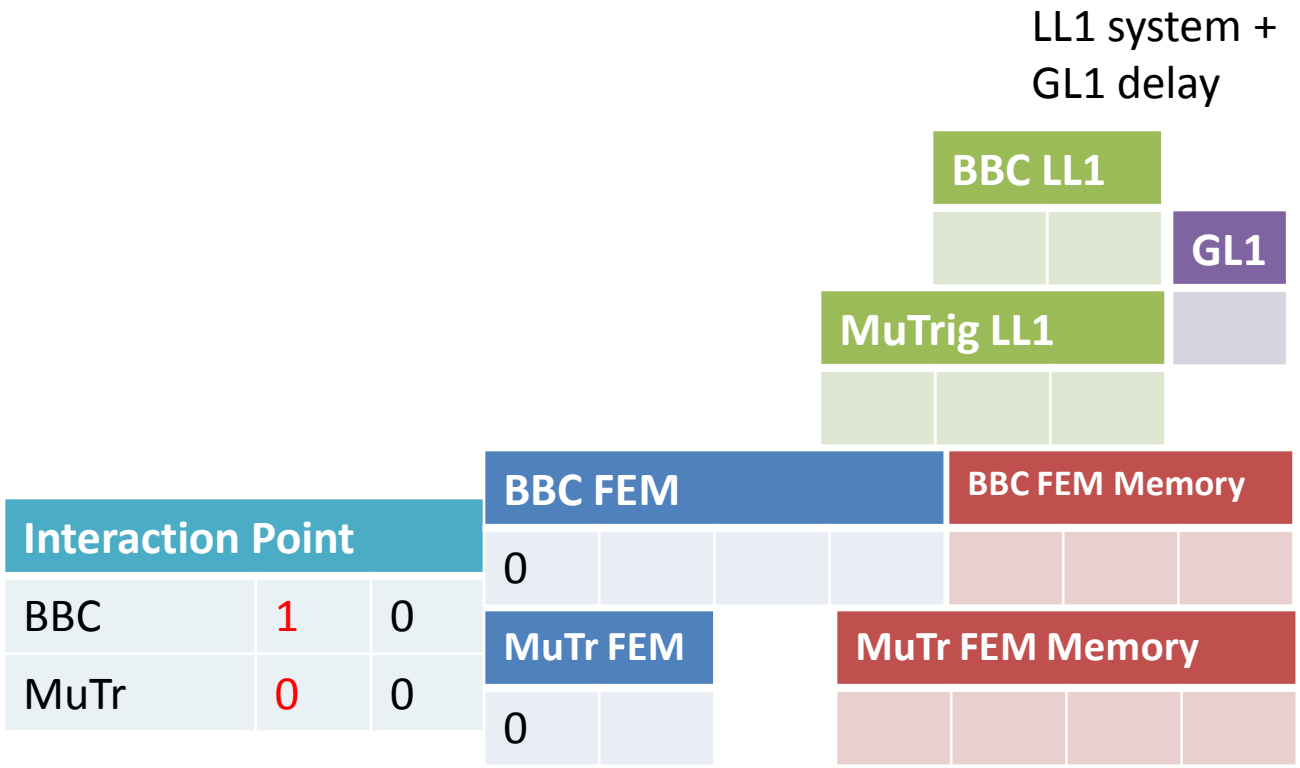
MuTr FEM is
faster than we
think it is!

If things are NOT in time

Interaction Point		
BBC	0	0
MuTr	0	0

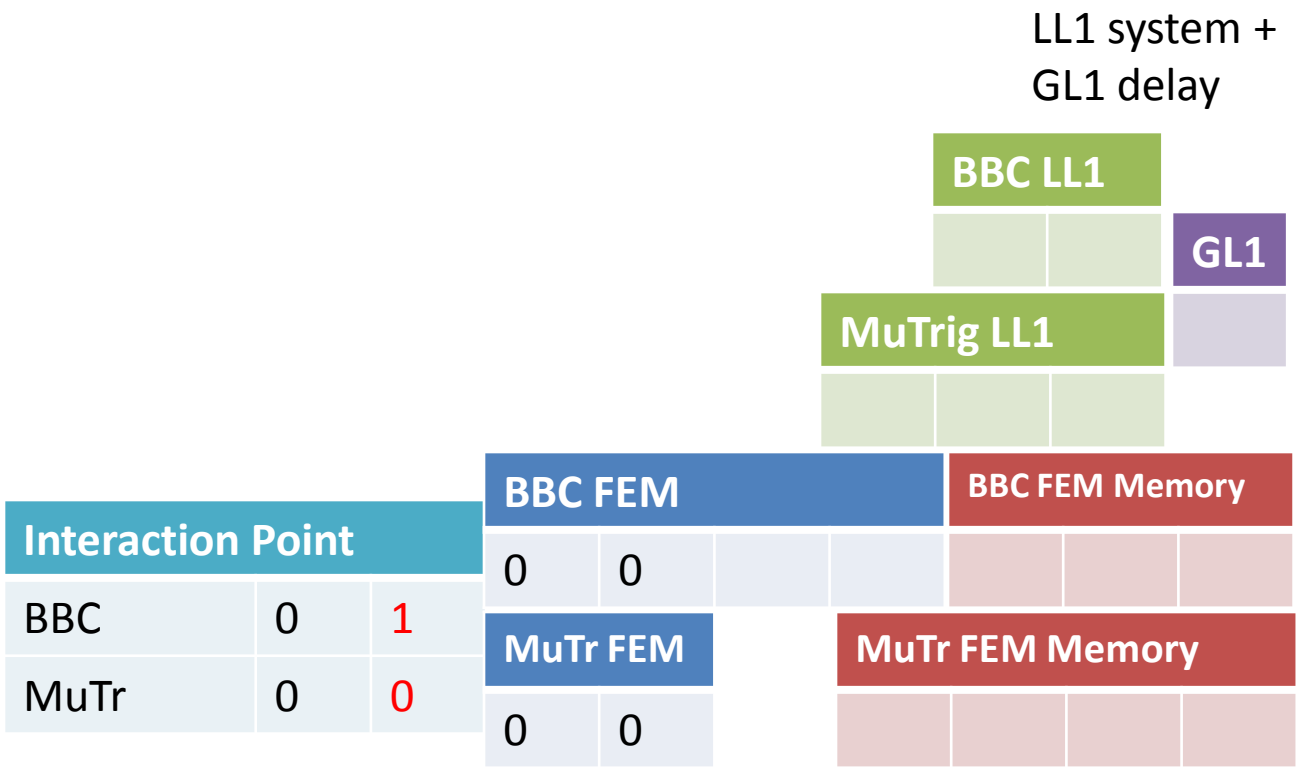


If things are NOT in time



Interaction Point		
BBC	1	0
MuTr	0	0

If things are NOT in time

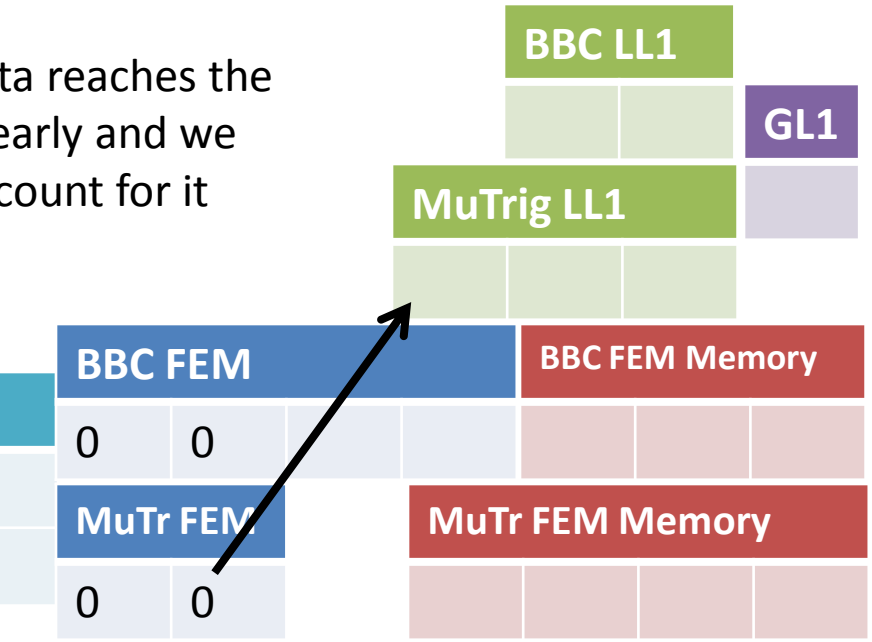


Interaction Point		
BBC	0	1
MuTr	0	0

If things are NOT in time

LL1 system +
GL1 delay

MuTr data reaches the LL1 too early and we don't account for it

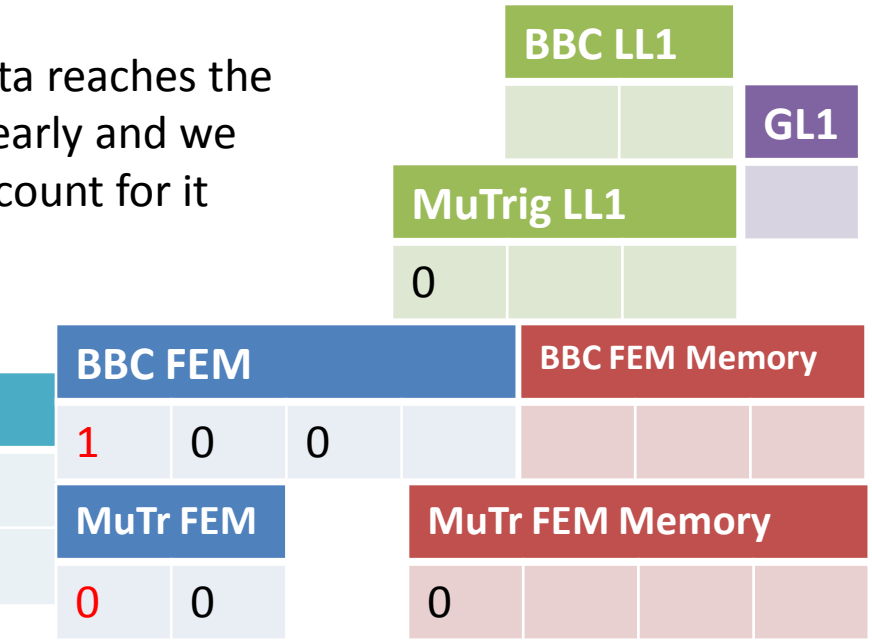


Interaction Point		
BBC	0	1
MuTr	0	0

If things are NOT in time

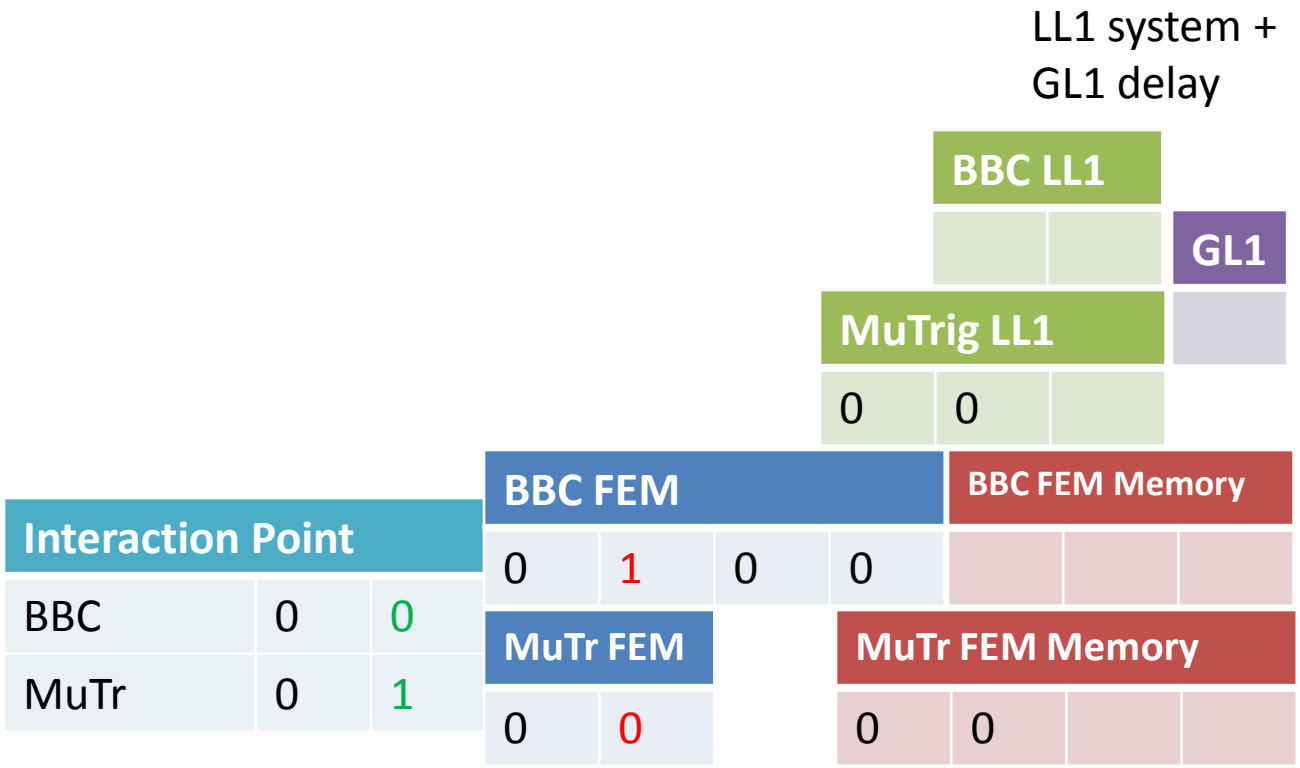
LL1 system +
GL1 delay

MuTr data reaches the LL1 too early and we don't account for it

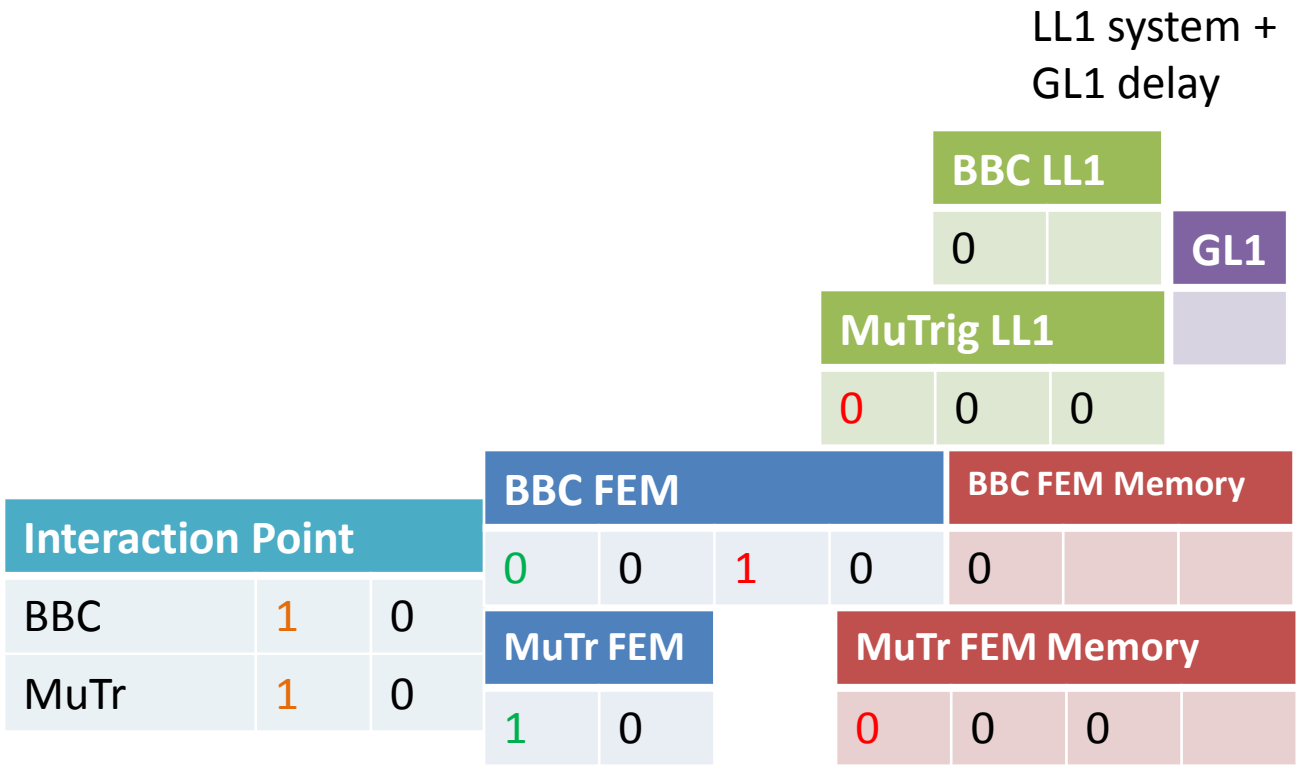


Interaction Point		
BBC	0	0
MuTr	1	0

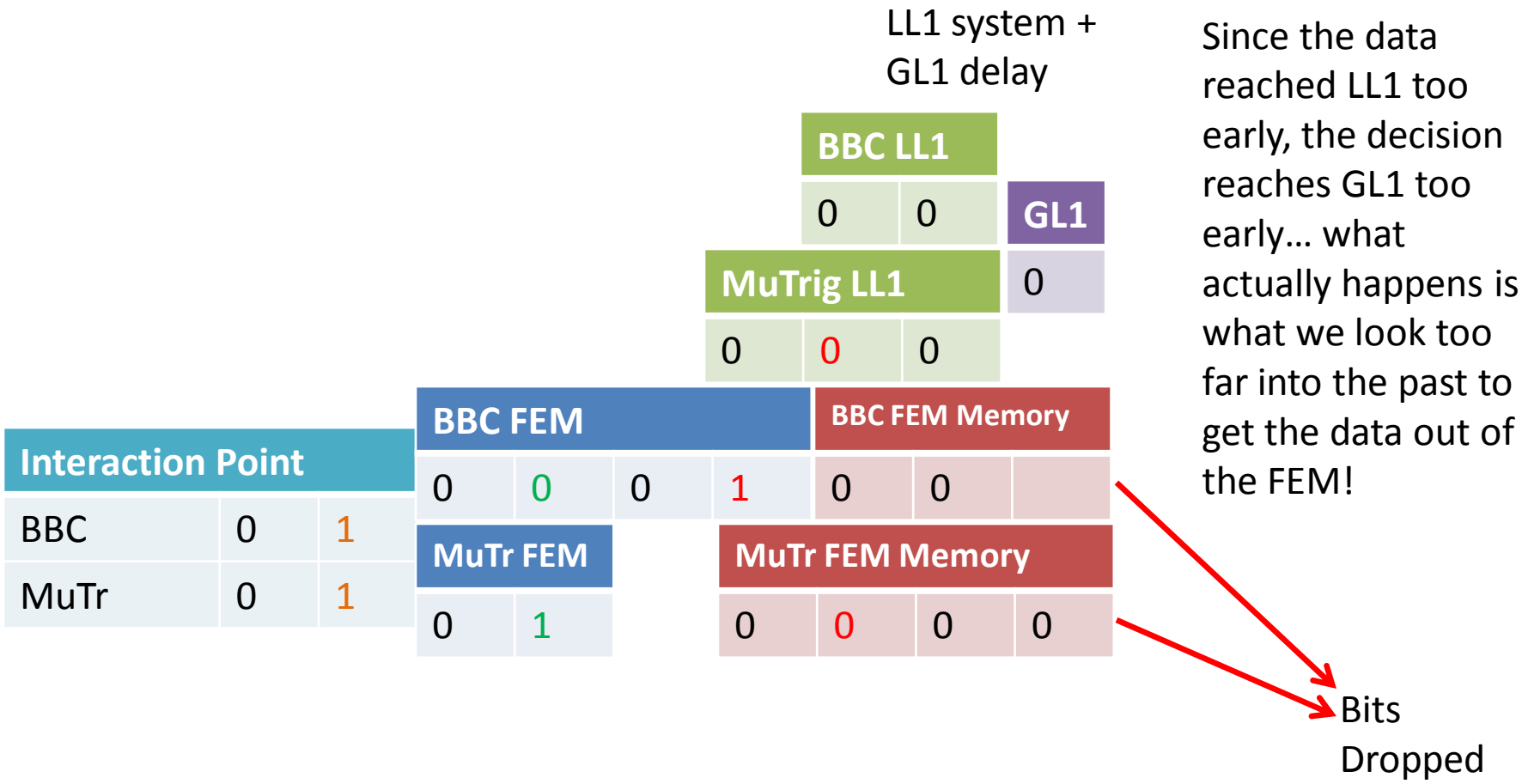
If things are NOT in time



If things are NOT in time

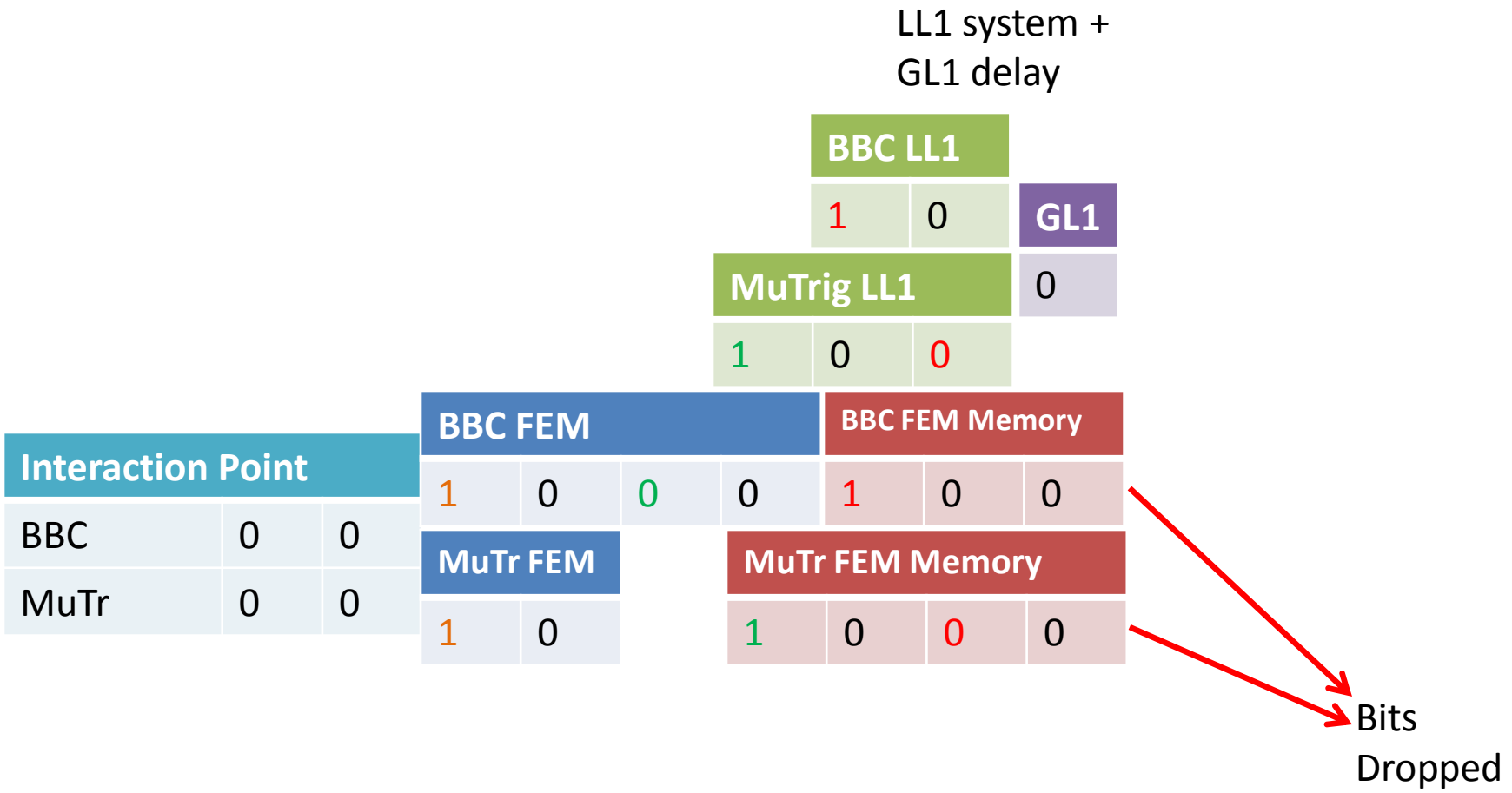


If things are NOT in time

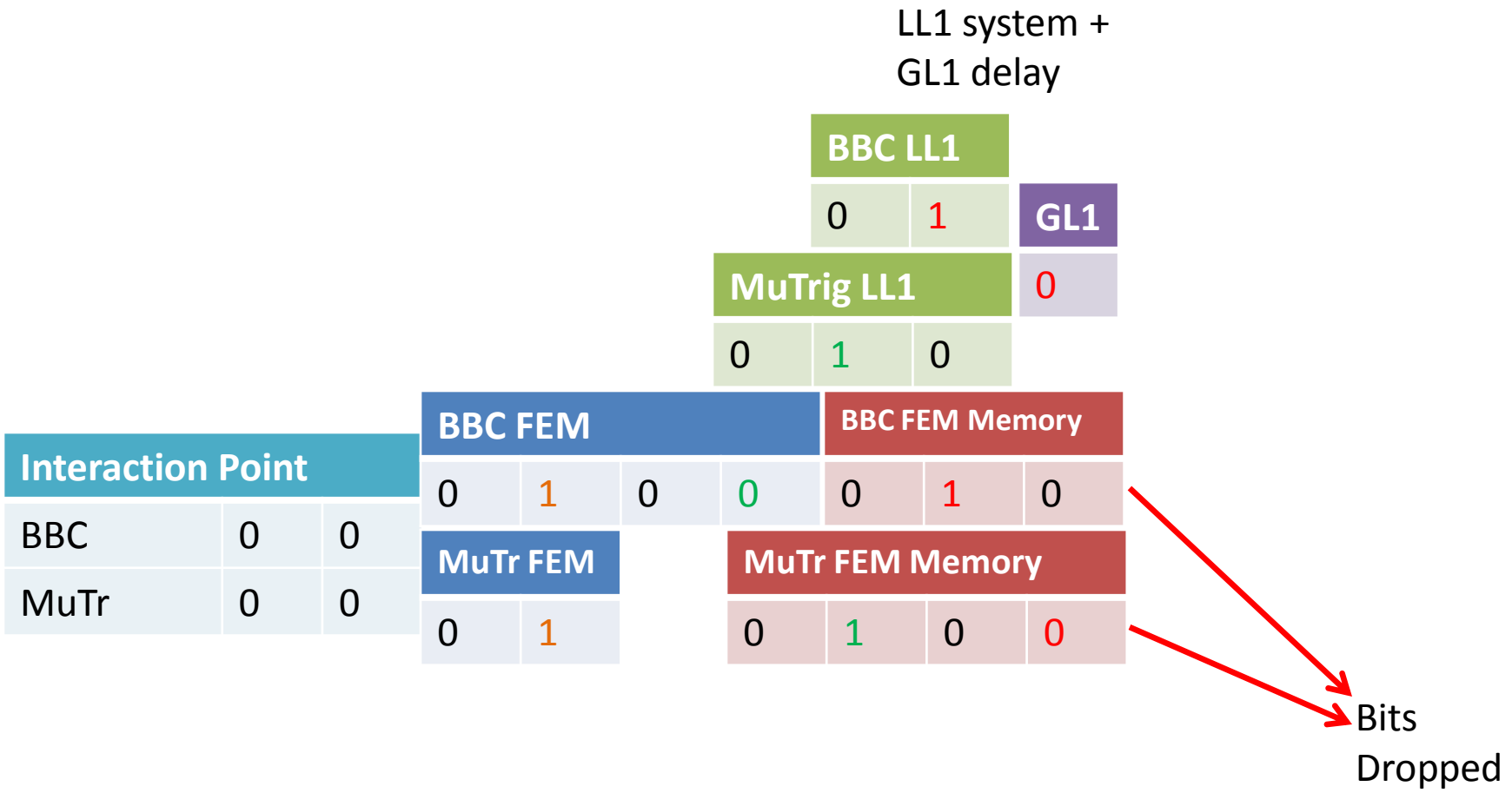


Since the data reached LL1 too early, the decision reaches GL1 too early… what actually happens is what we look too far into the past to get the data out of the FEM!

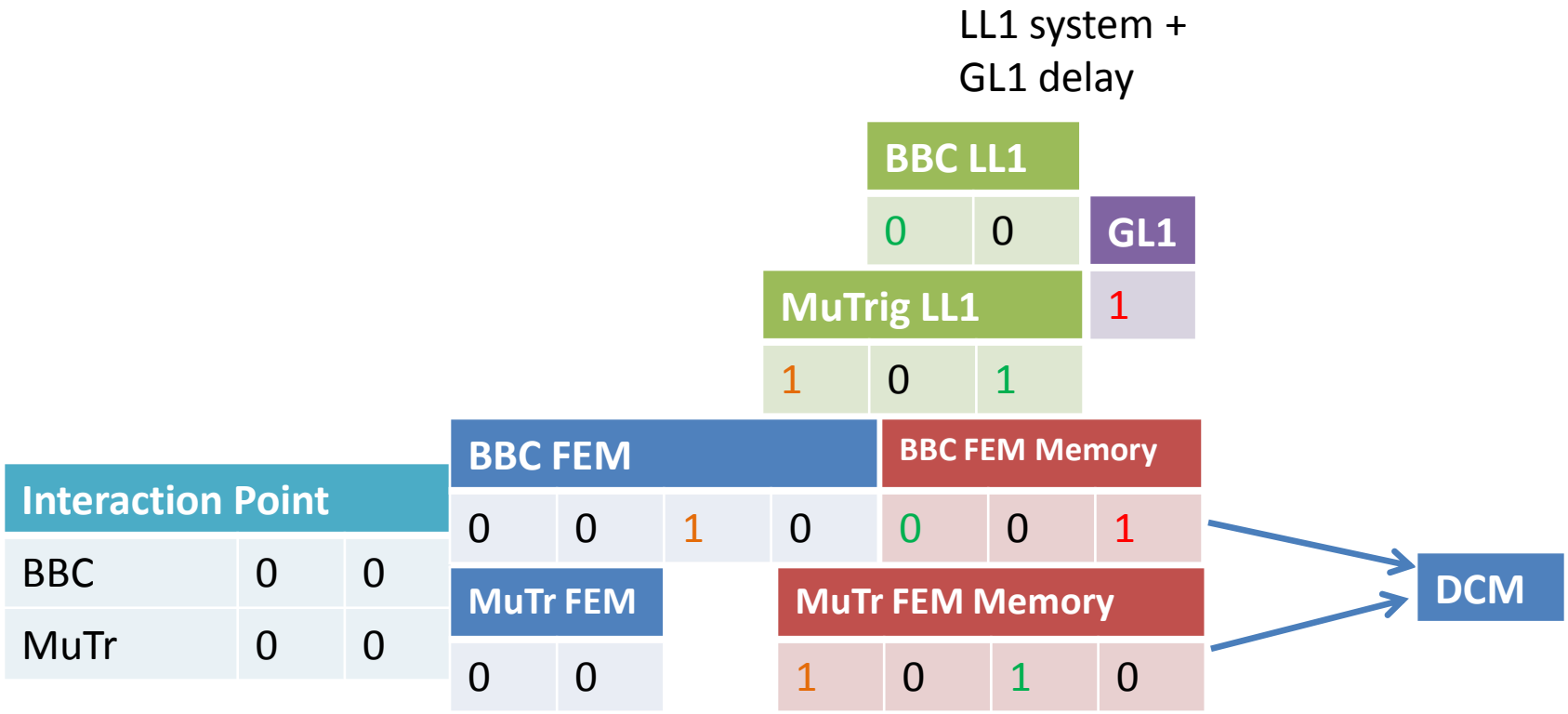
If things are NOT in time



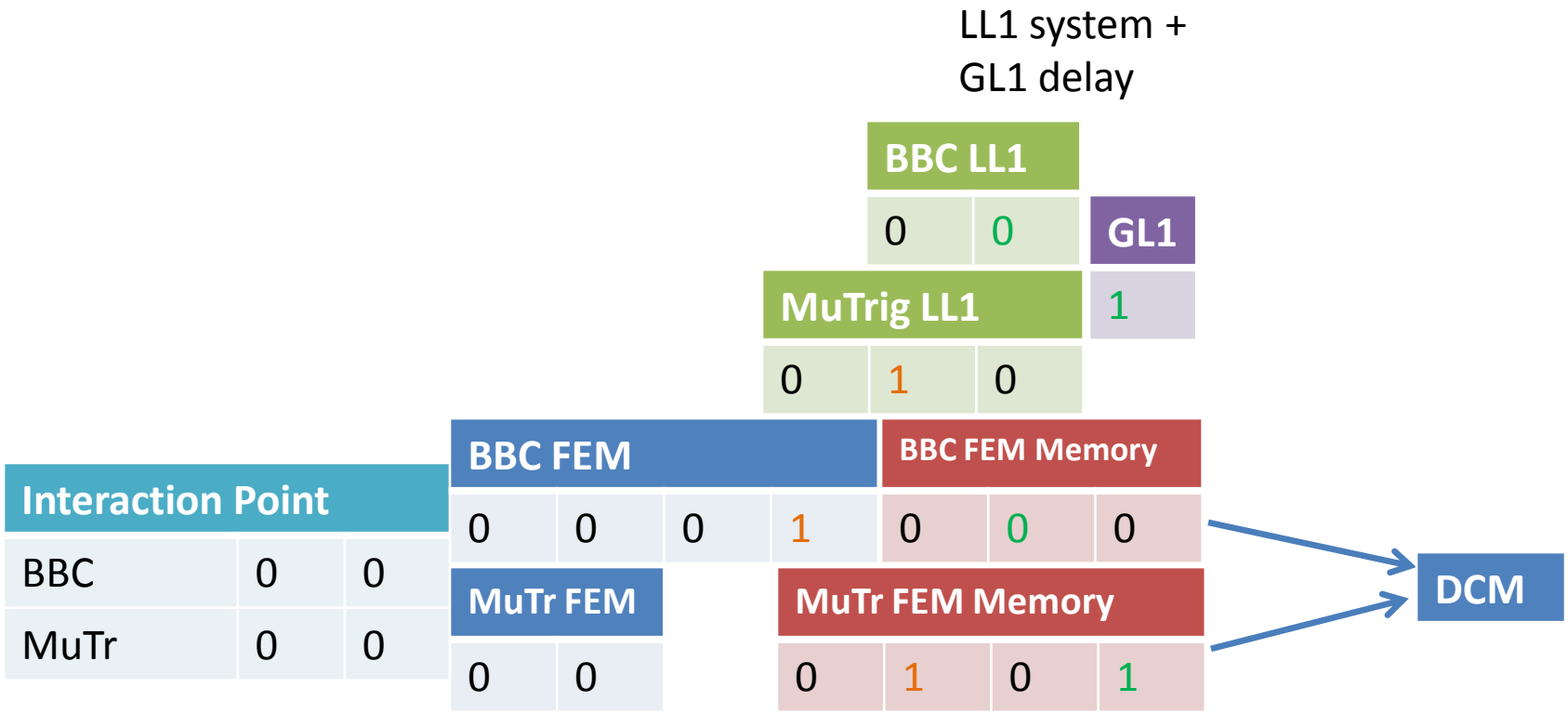
If things are NOT in time



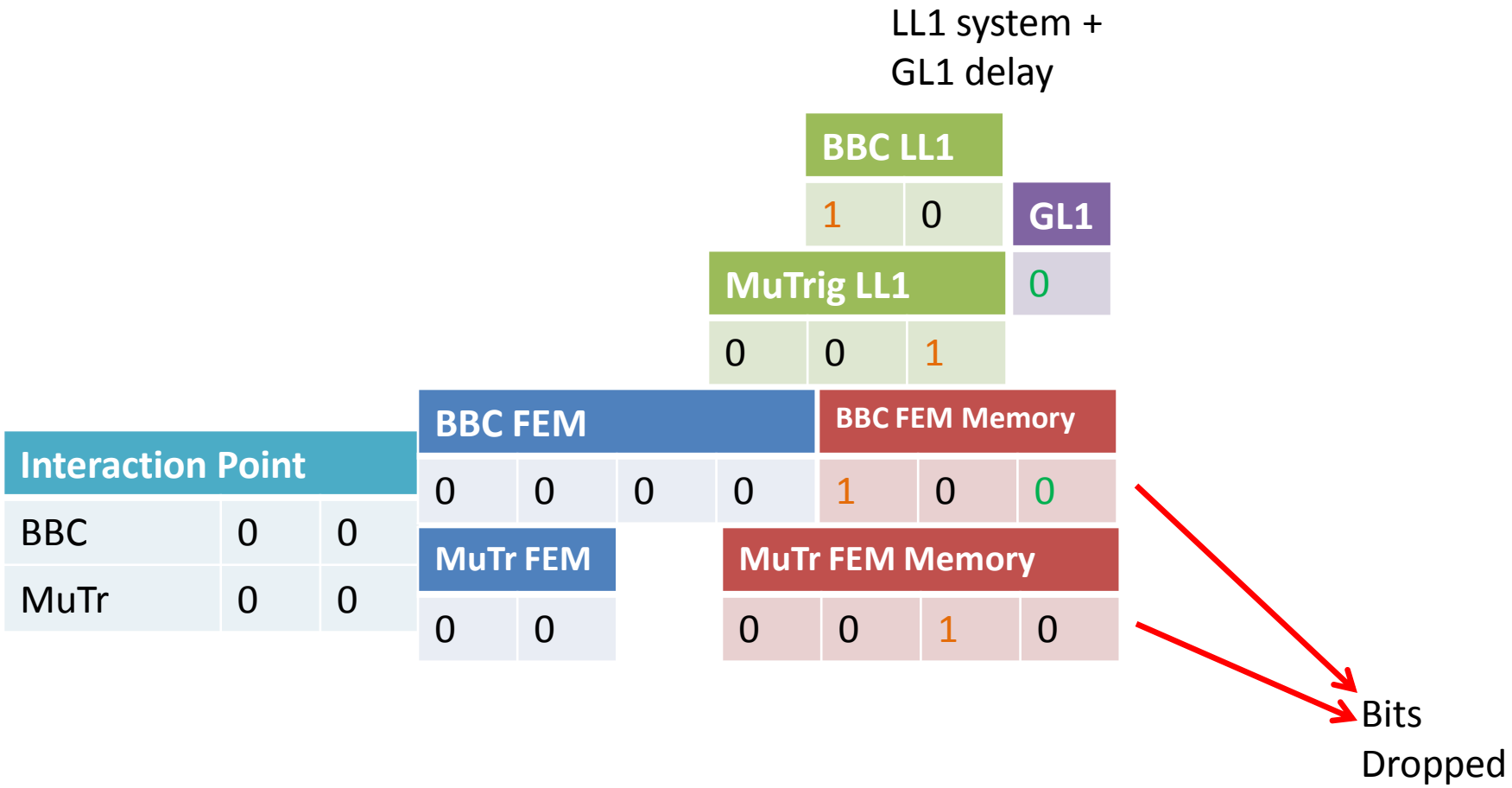
If things are NOT in time



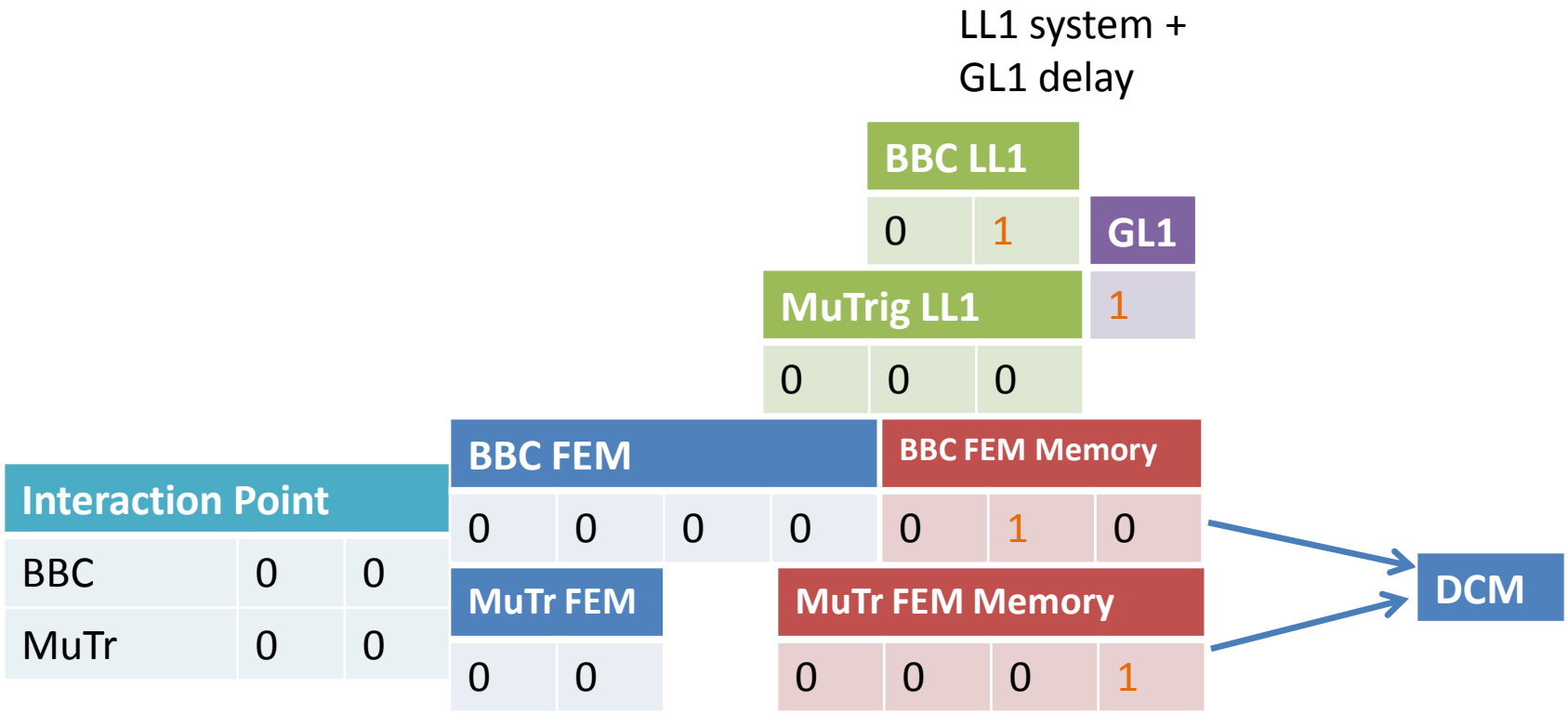
If things are NOT in time



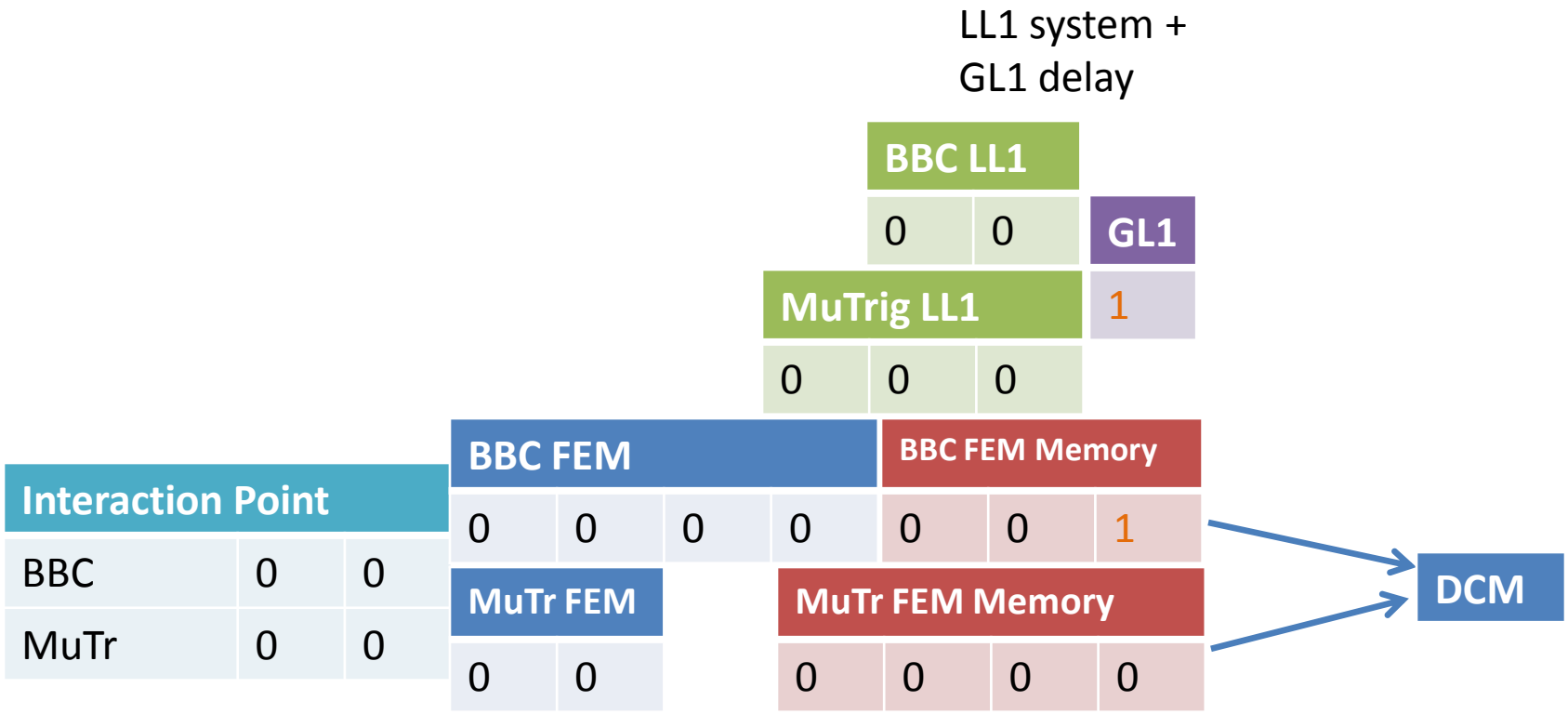
If things are NOT in time



If things are NOT in time



If things are NOT in time



If things are NOT in time