



ASML

EUV reticle handling and storage

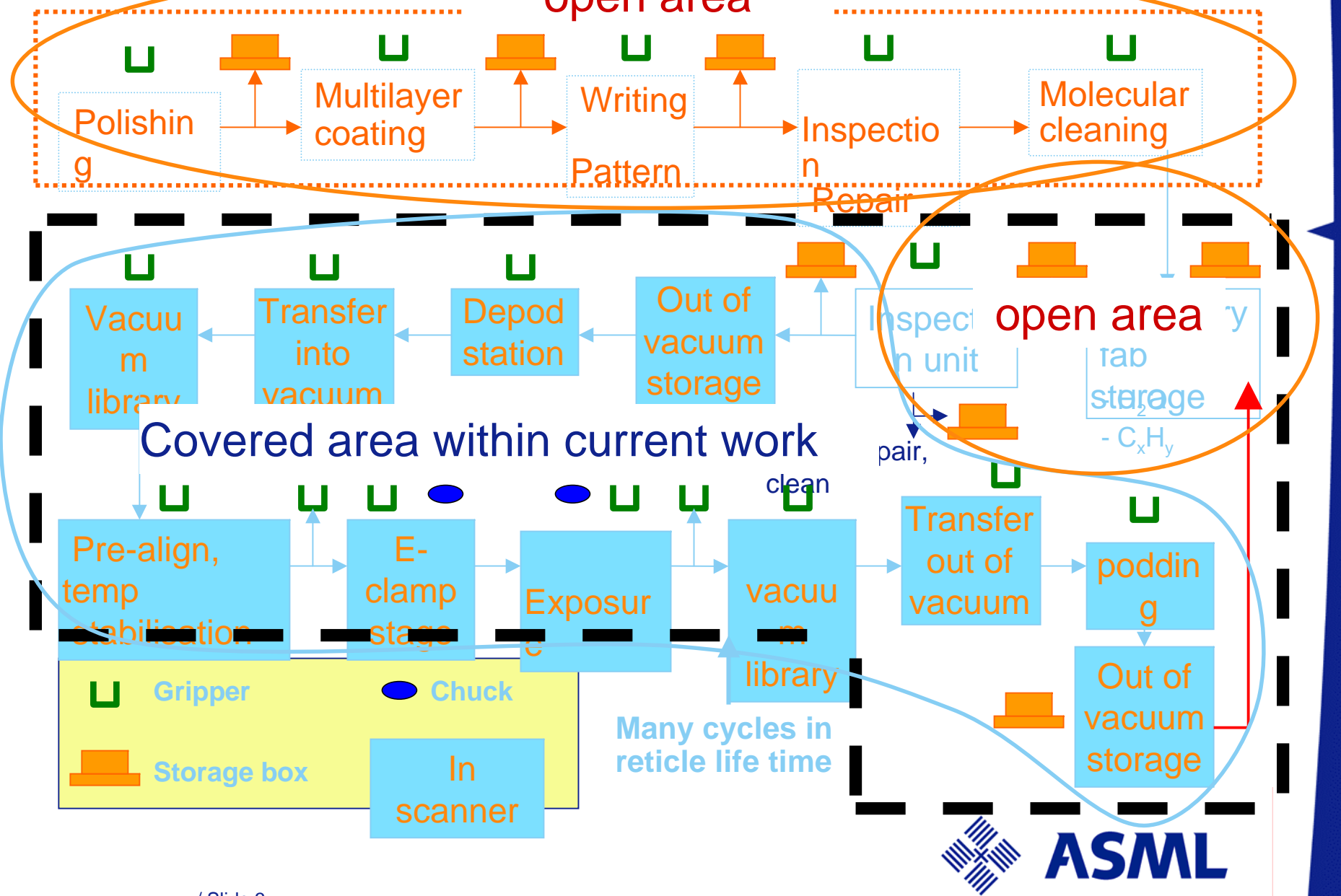
Brian Blum, Bas Mertens, Erik Ham, Gert-Jan Heerens, Rob Snel,
Rob Lansbergen, Hans Meiling, Roel Moors, Henk Meijer

Overview

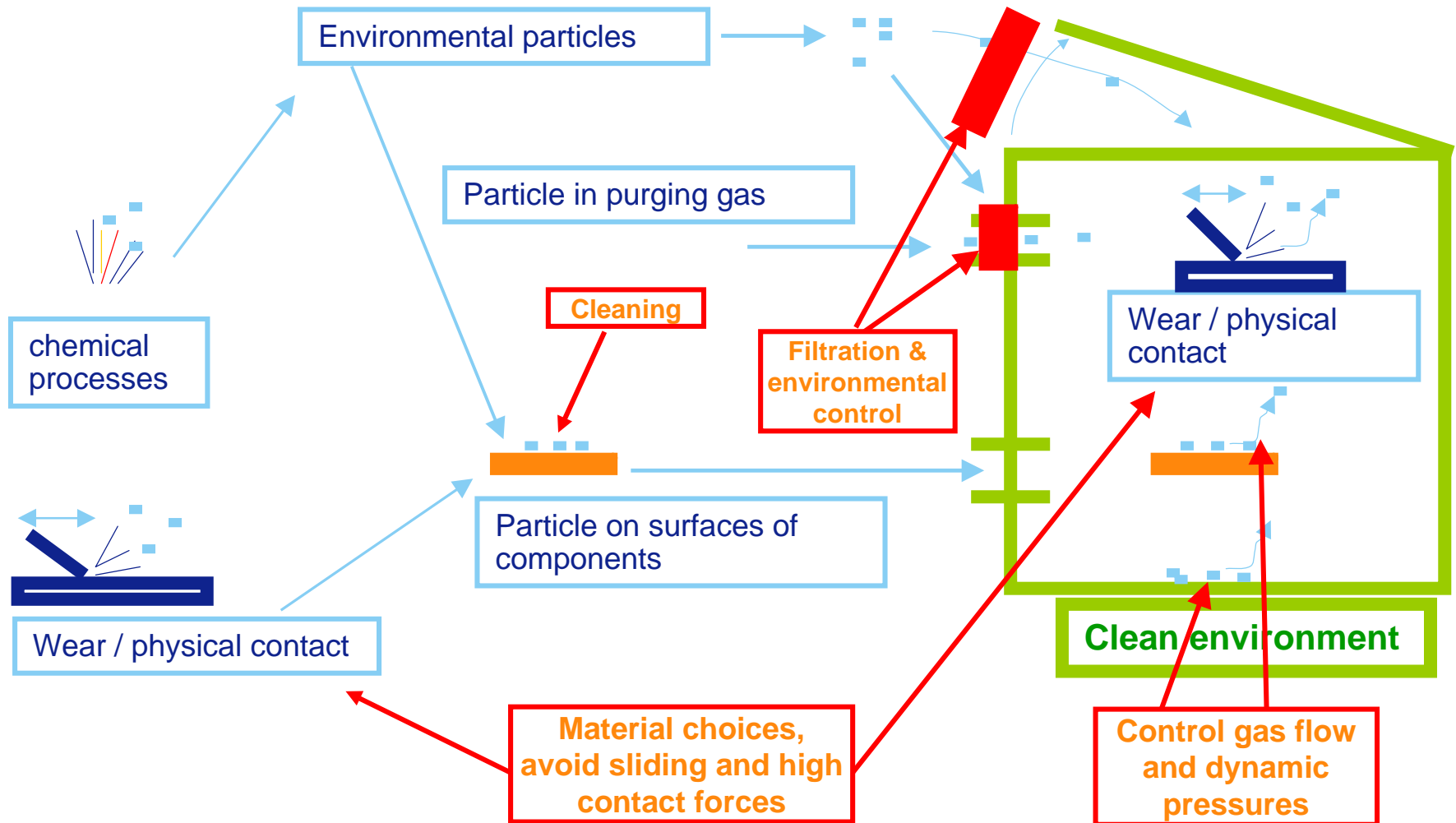
- Introduction
 - Life of a reticle: open areas on storage / transport
 - Sources of particle contamination
- Strategy for particle contamination control
- Reticle Handler Design: reticle frame
- Loadlock/FuMo Test Series
- Reticle Handler Design: storage box
- Development areas

Manufacturing: once in life time

Life of a reticle



Sources of particle contamination



Strategy for particle contamination control

- Minimize particle generation:
 - Minimize number of contacts (by design)
 - Choose materials that produce low numbers of particles through contact (get knowledge through basic experiments)
- Prevent particle migration
 - By design
 - Proper pumping / venting sequences (develop through experiments)

Use reticle handling frame

Suitable materials combinations for practical designs have been found

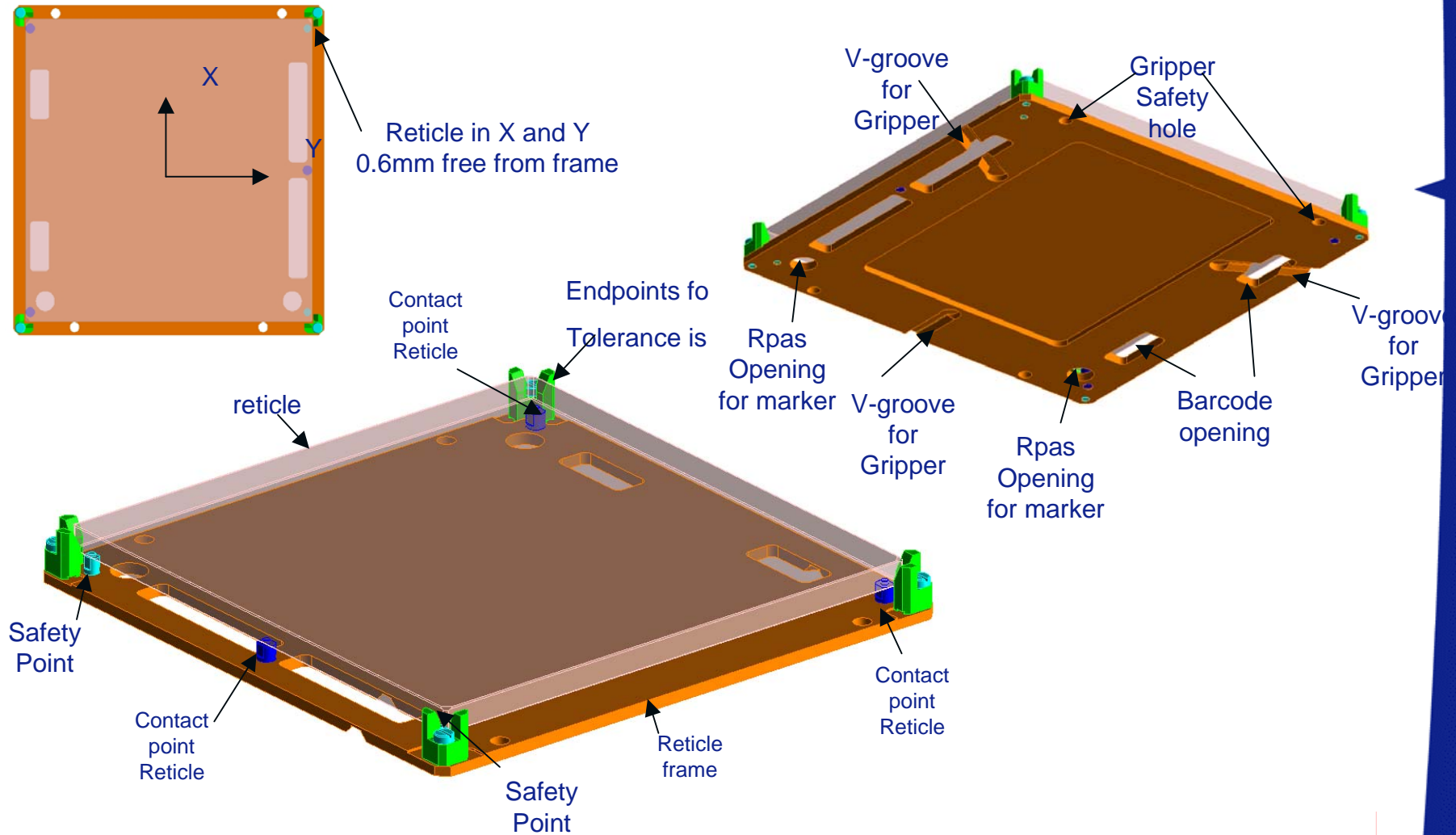
Loose particles dislodge, particles generated through contact do not dislodge



ASML

RH Design: reticle frame

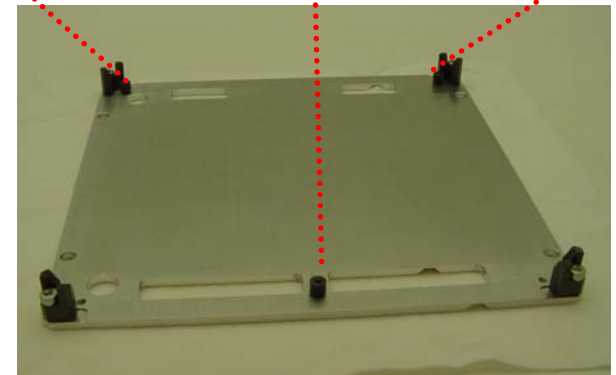
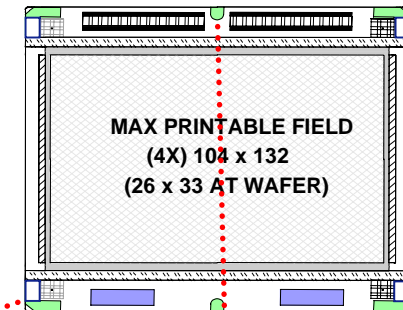
- frame is used to minimize number of reticle contacts -



RH design: reticle frame

- Handling of reticle limited to:
 - take-over from mask-shop box to reticle storage box
 - release to reticle stage / take from reticle stage
- In all other cases, the reticle frame is handled

proposed handling areas
(green areas)

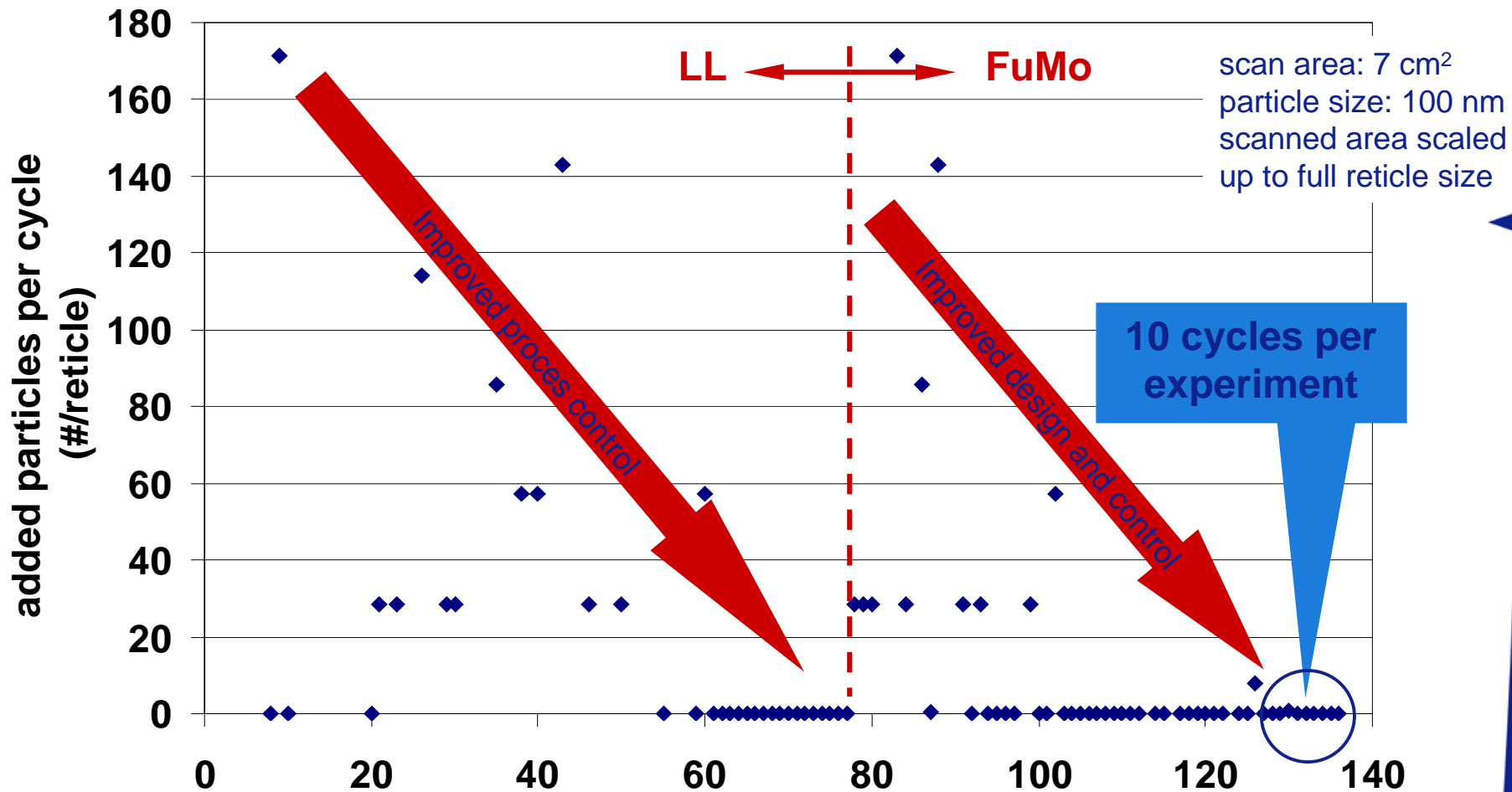


RH frame



ASML

Frame, loadloack and box for RH have been optimized and proto HW is being built based on FuMo results



Outcome: zero added particles measured routinely !

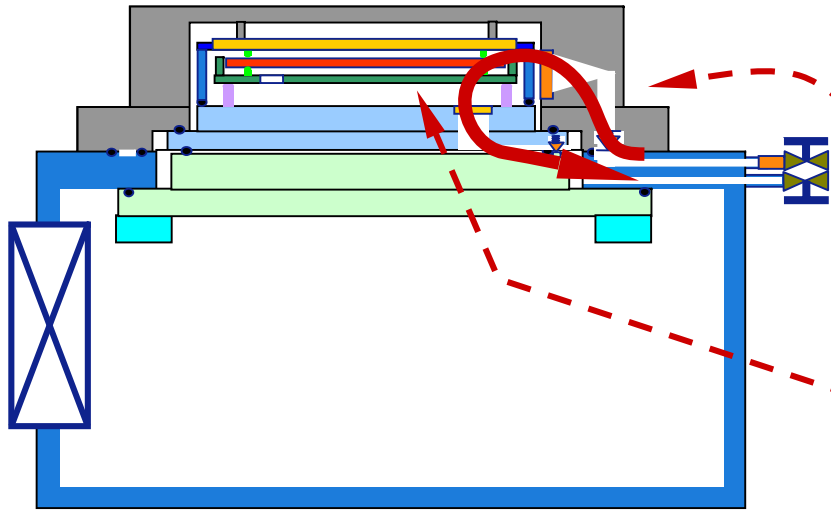


Overall conclusions from LL/FuMo test series

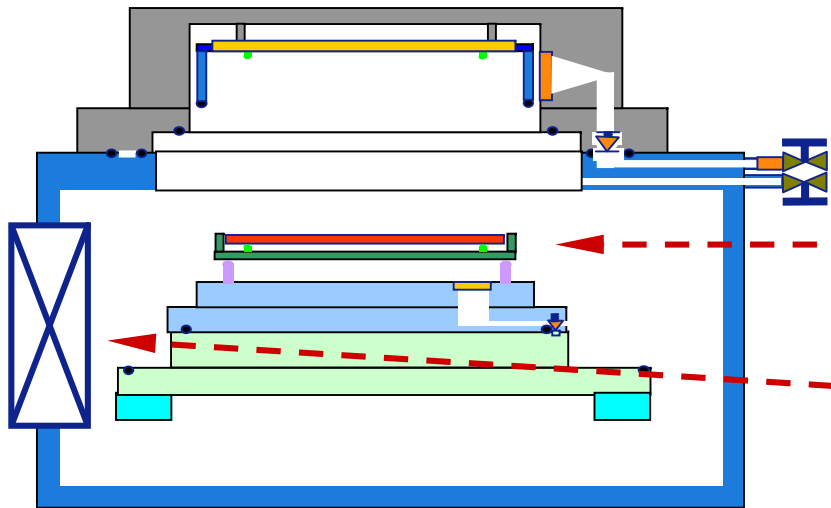
- Design & process for zero added particles
 - LL/Storage box design optimized for pump down speed
 - Materials selected for vacuum compatibility (outgassing, pressure difference)
 - Thorough cleaning of all components
 - Proper particle filters and vacuum valves
 - Integration of reticle storage box with LL
 - Unidirectional gas flow during pumping and venting
 - Protective inner cover incorporated for use during pressure transitions
 - Develop dedicated mechanics to avoid particle production (reticle storage box release mechanism)
 - Reticle movement and handling only in vacuum



RH design: storage box and LL

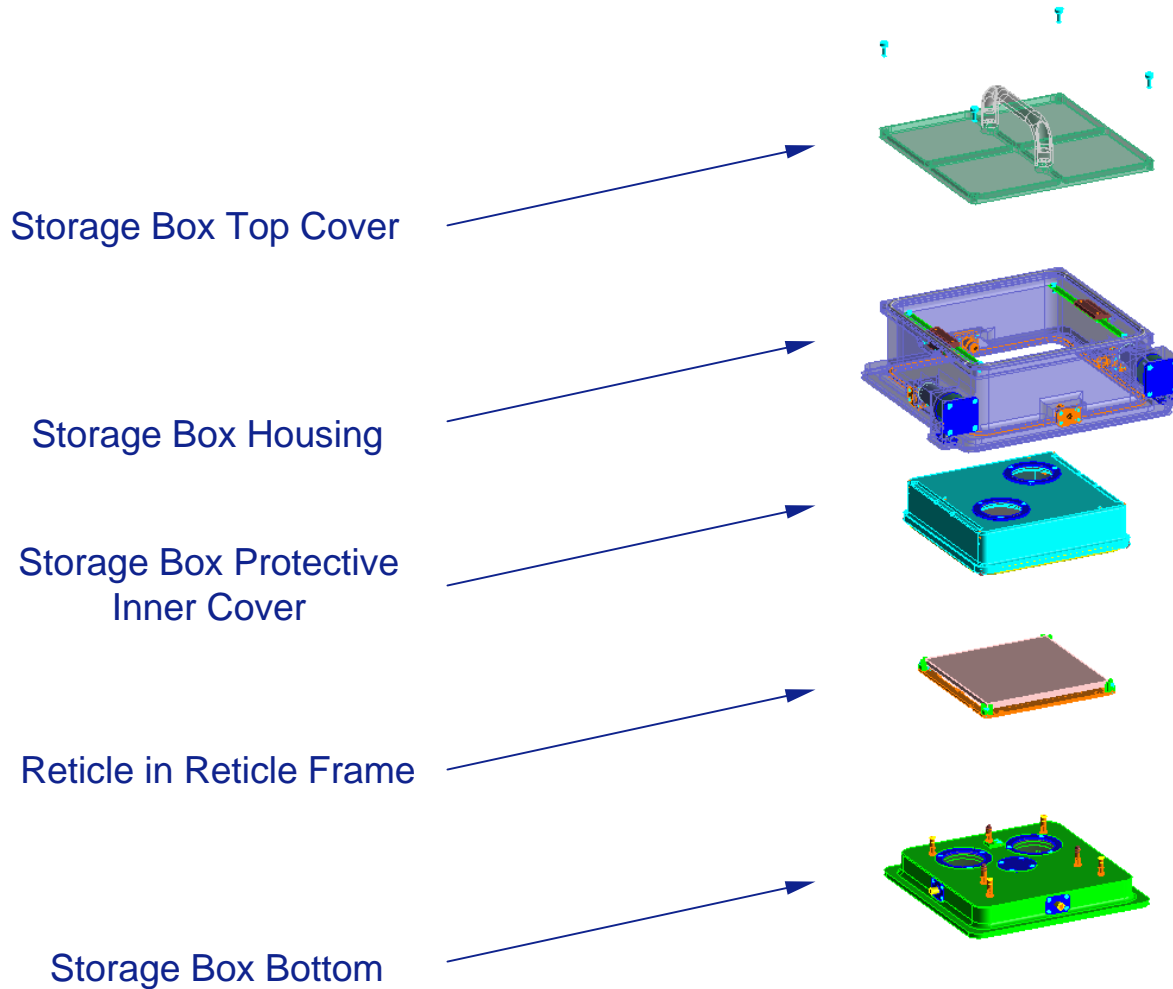


- Gasflow directed away from pattern area
- Gasflow through filters in same direction during pumping and venting
- During pumping / venting a protective inner cover keeps reticle clean by making gasflow diffuse



- Movement of reticle only under vacuum conditions
- Use particle-free valve towards robot chamber and RS area

ASML Alpha Demo Storage Box - exploded view



Development areas in reticle storage

- Basics:

- Do we want one storage box through complete reticle life - manufacturing (including patterning) and fab-life?

or

- A solution just for “life in the IC fab” which would include:
 - reception: repack in ASML-type storage box with reticle frame
 - storage: vacuum, purge or other environment ?
 - transport: high G-forces, shock, vacuum environment?
 - inspection tool: reticle orientation, vacuum tool ?
 - cleaning tool: vacuum tool ?
 - litho tool: design compatible with high reticle throughput for production tools



Development areas in reticle storage

- Development areas still open:
 - Environment of reticle during transport and storage (vacuum, purge, ...) outside the exposure tool
 - Interfacing to mask shop and fab infrastructure (e.g., to particle cleaning tools, particle inspection tools)
- We offer the ASML solution as a starting point for discussions
 - ASML storage box is based upon current standard (RSP150)
 - Differences exist in because of pump down & venting, vacuum compatibility of materials, mechanics to avoid particle production (bottom release mechanism) and construction to withstand vacuum forces
 - Of these, construction to withstand vacuum forces is the only thing unique to the ASML concept (is not the cost driver)

