sm (QStateMachine)

uninitState → **spim.initialized** → readyState

readyState ← **spim.stopped** | **spim.captureStarted** → capturingState

capturingState

freeRun == true

freeRunState
entered:
for each orca:
    orca->cap_start();
galvoRamp->start();
cameraTrigger->start();

freeRun == false

acquisitionState
see box on the right

spim.error → errorState

(H)

---

acquisitionState

entered:
for each orca:
    orca->buf_alloc();

precaptureState (QState::ParallelStates)

entered:
if (!capturing)
    return;
galvoRamp->stop();
cameraTrigger->stop();
reset stage velocity;
move stages to target pos;
pollTimer->start(200);

pollingState
pollingInProgressState → **dev.onTarget** → ⦿ pollingDone

pollingState
pollingInProgressState → **dev.onTarget** → ⦿ pollingDone

precaptureState.finished →

captureState (QState::ParallelStates)

entered:
pollTimer->stop();
for each orca:
    orca->cap_start();
    acqThread->start();
galvoRamp->start();
cameraTrigger->start();
set stage velocity;
move stackStage;

camState
camBusyState → **orca.stopped** → ⦿ finalState

camState
camBusyState → **orca.stopped** → ⦿ finalState

finished:
currentStep++;
if currentStep >= N
    stop();
    return;
increment currentSteps[];

captureState.finished →

exited:
pollTimer->stop();

---

pollTimer

timeout:
for each dev in stages:
    dev->isOnTarget();