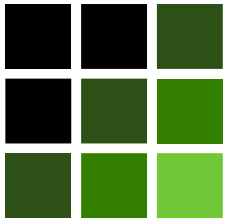


DRAMSys

The Open Source DRAM Simulator DRAMSys

Prof. Dr. Matthias Jung, JMU Würzburg

accellera SYSTEMS INITIATIVE™

RPTU Rheinland-Pfälzische Technische Universität Kaiserslautern Landau

Fraunhofer IESE

17 Computer Engineering

Julius-Maximilians-Universität Würzburg

SYSTEMC™ EVOLUTION FIKA

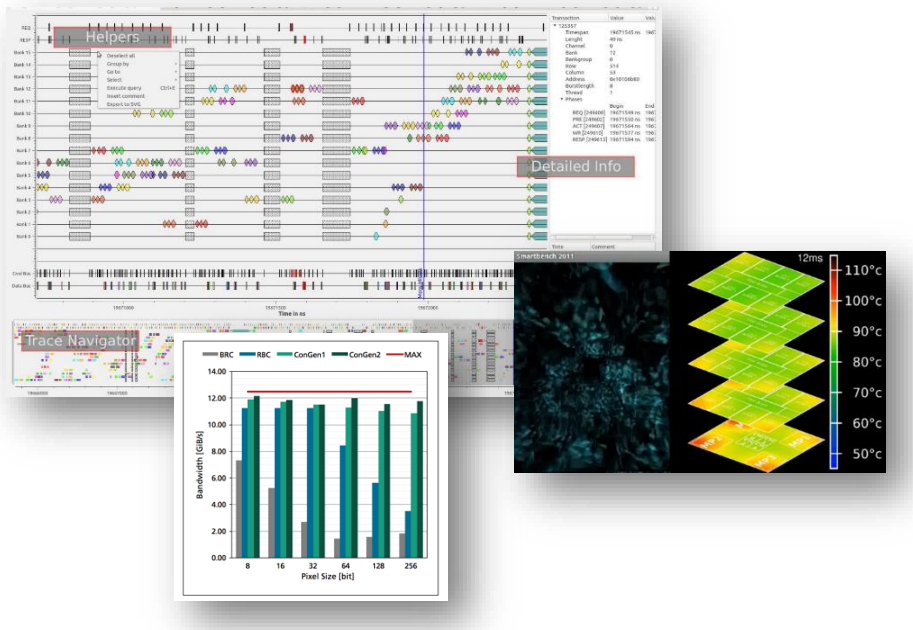
DRAMSys in a Nutshell

Simulation and Design Space Exploration of Modern DRAM-based Memory Systems:

- Which DRAM configuration?
- When to support DDR5 or LPDDR5?
- How to configure the memory controller?
- What is the system-level application behavior?

DRAMSys Offers:

- High-speed and flexible models of all standards
- Fast and accurate design space exploration
- Early identification of bottlenecks
- Connection to cores (e.g. SystemC, gem5, …)

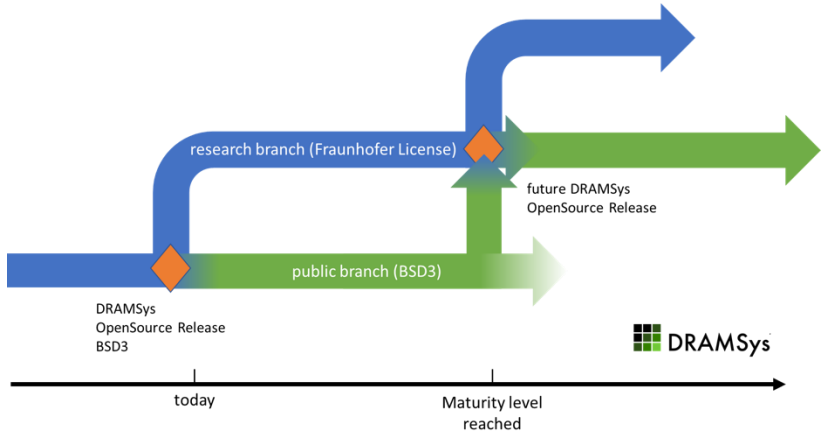


accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

DRAMSys Open Source Model

- Open source: DDR3/4, LPDDR4, Wide I/O 1/2, GDDR5/X, GDDR6, and HBM2
- Commercial/academic licenses: DDR5, LPDDR5, HBM3, Trace Analyzer tool
- New standard models will be open-sourced when a level of maturity is reached
- Customer-specific consulting, modifications and developments



Thanks to our Key Partners:

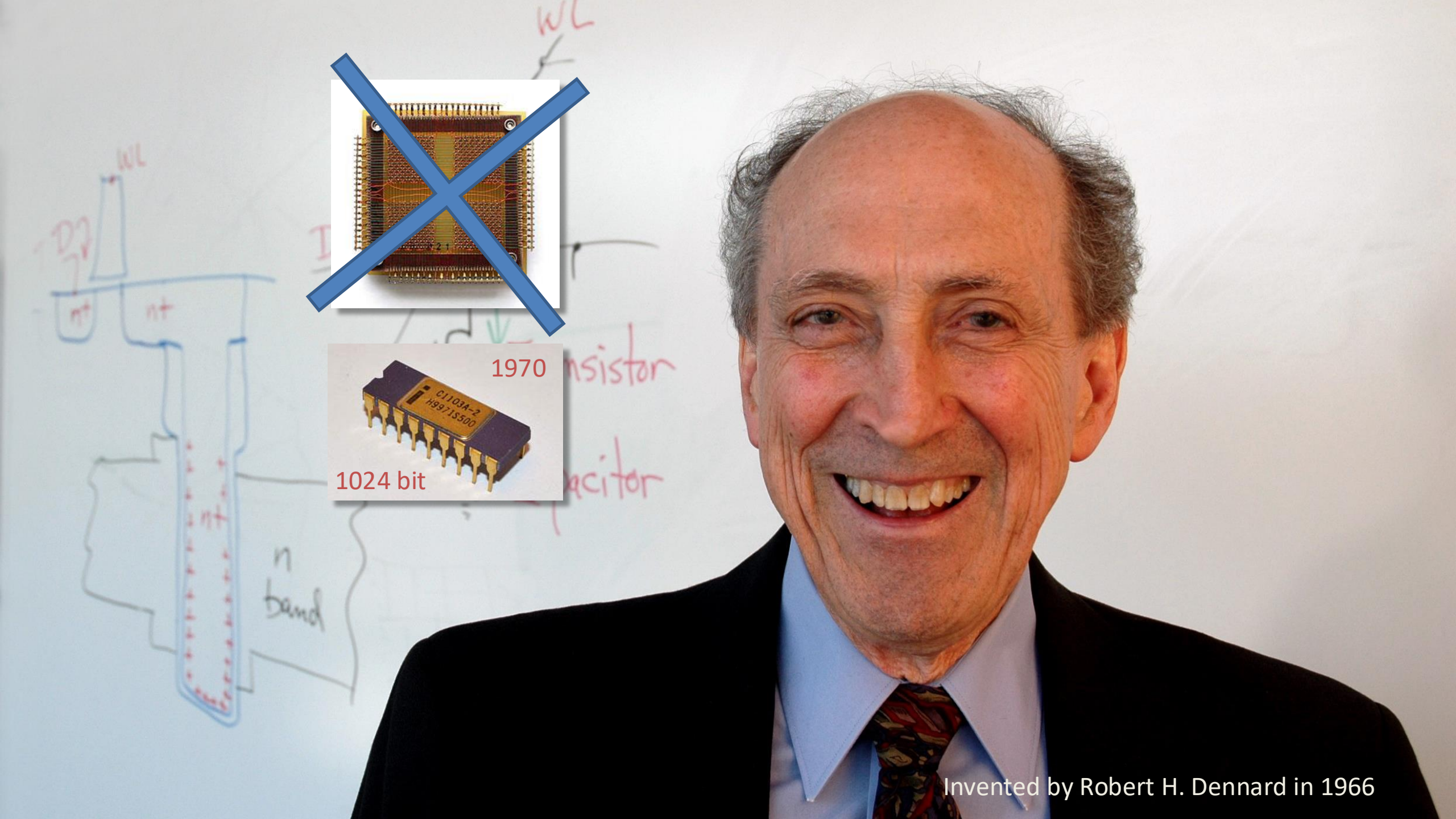
Rambus

Mercedes-Benz

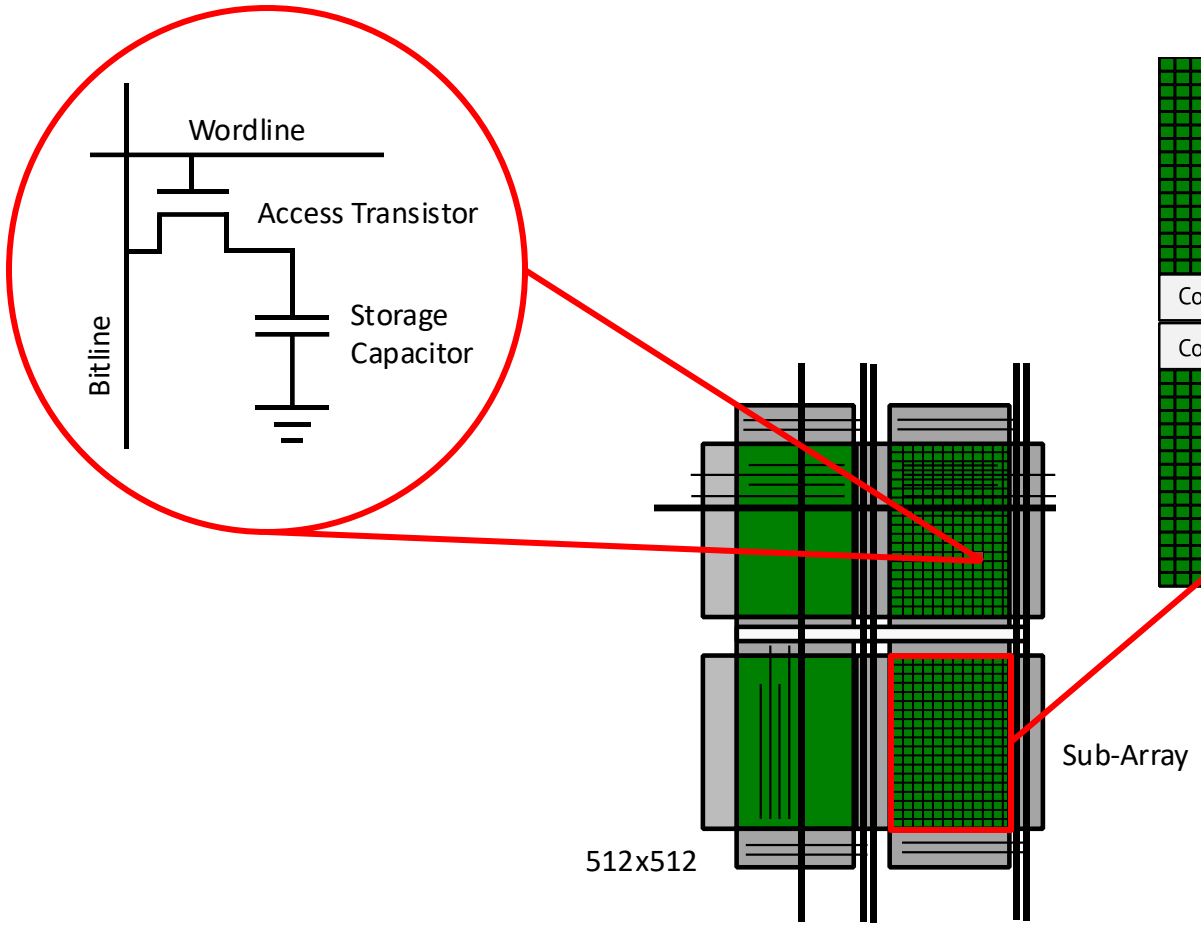
ARTERIS IP

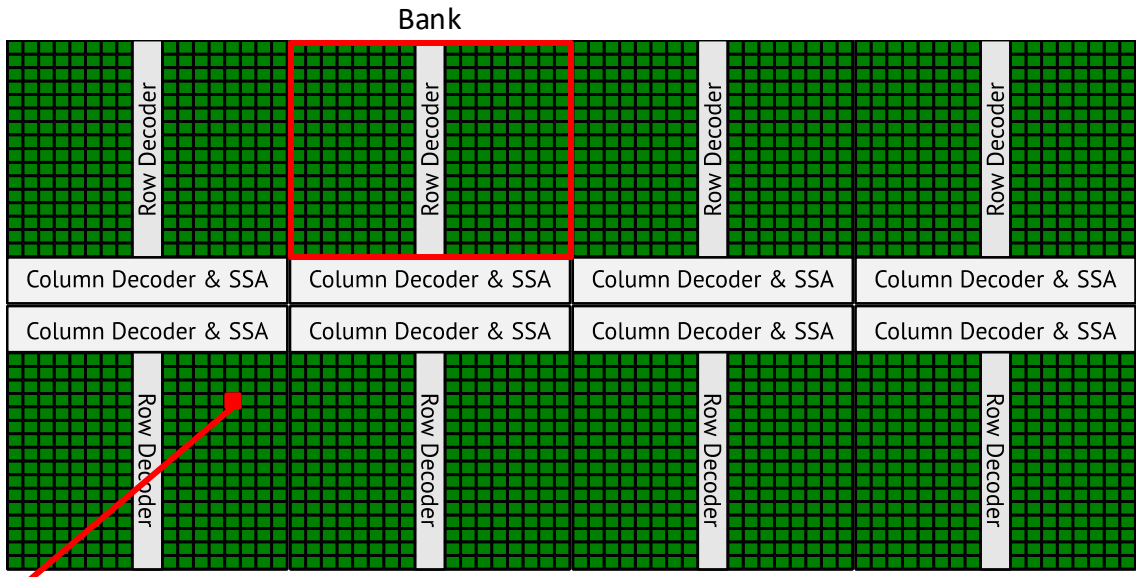
accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA


WL
WL
n+
n+
n+
n
band
nsistor
acitor
1970
C1103A-2
H9971S500
1024 bit
Invented by Robert H. Dennard in 1966

The DRAM Device / Operation





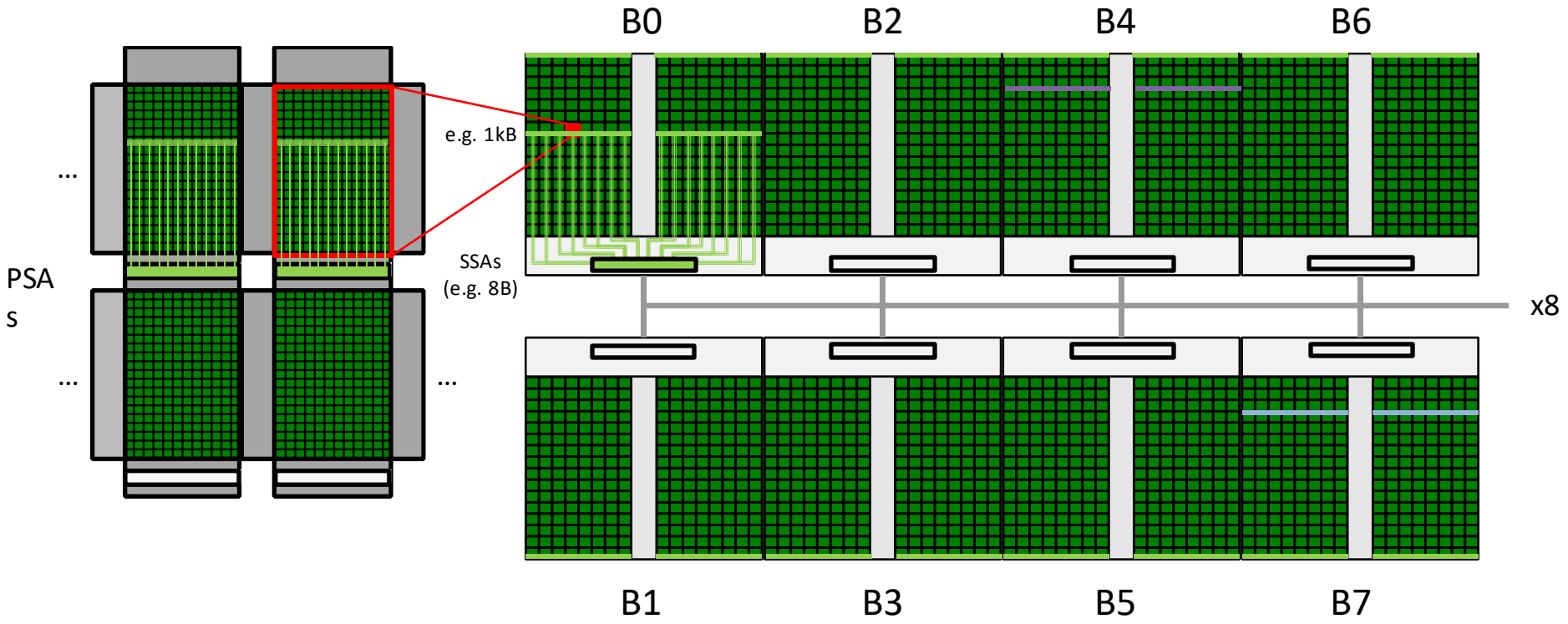
- Using Sub-Arrays for efficient wiring
- Bank parallelism, but banks share data and command bus

E.g. Samsung DDR3, by Chipworks

accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

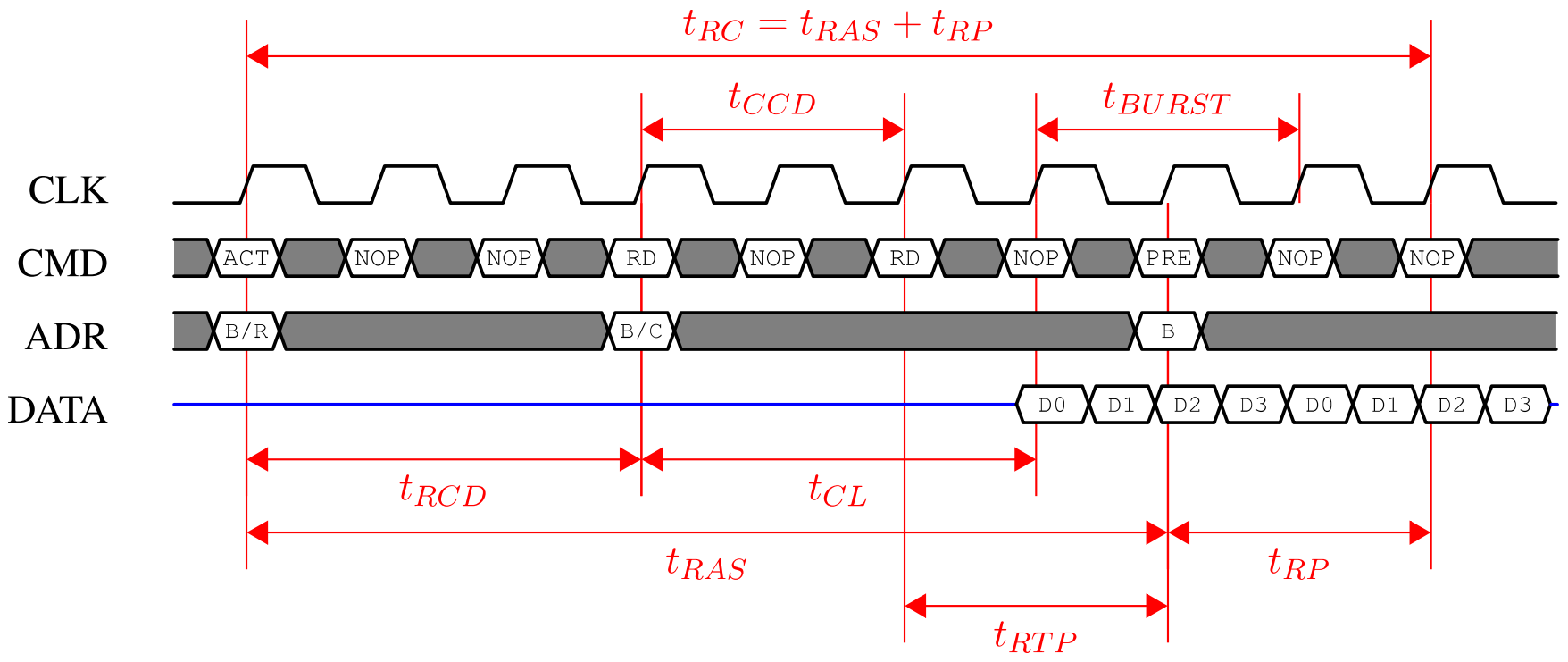
DRAMs Basic Operations



Important DRAM Commands:

- **ACT**: Activates a specific row in a specific bank (sensing into PSA) [*tRCD*]
- **RD**: Read from activated row (prefetch from PSA to SSA and burst out) [*tCL + tBURST*]
- **PRE**: Precharges set LWL=0 set LBL=VDD/2 [*tRP*]
- **REFA:** DRAM cells are leaky and have to be refreshed [*tREFI & tRFC*]

JEDEC Standard: e.g. Timing Dependencies

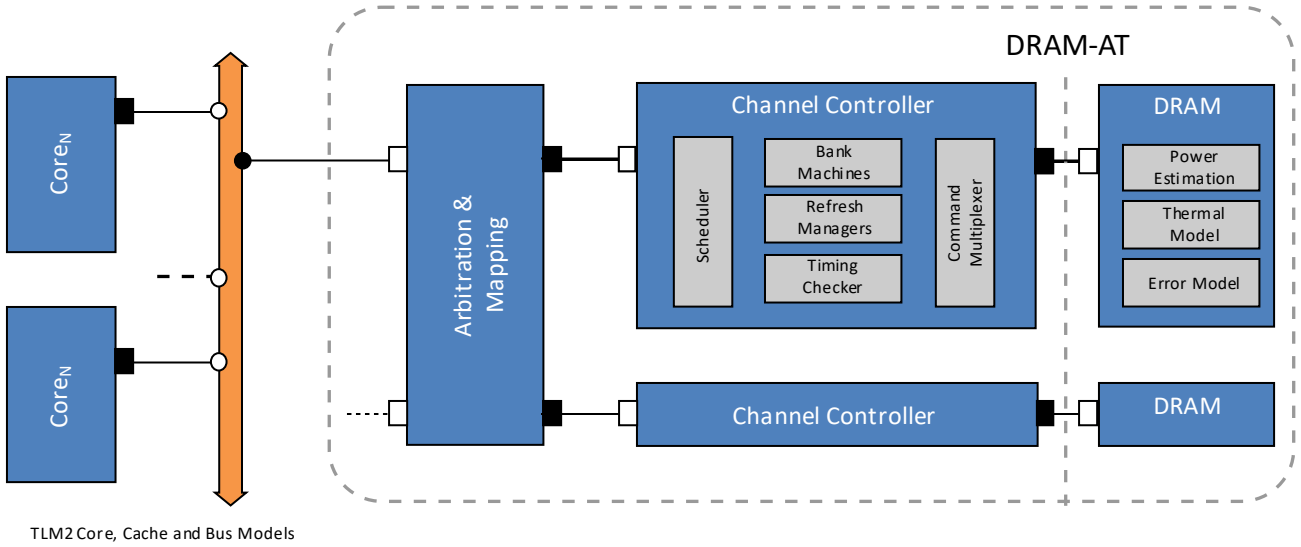


Timing dependencies must be fulfilled by the DRAM controller

accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

DRAMSys Architecture



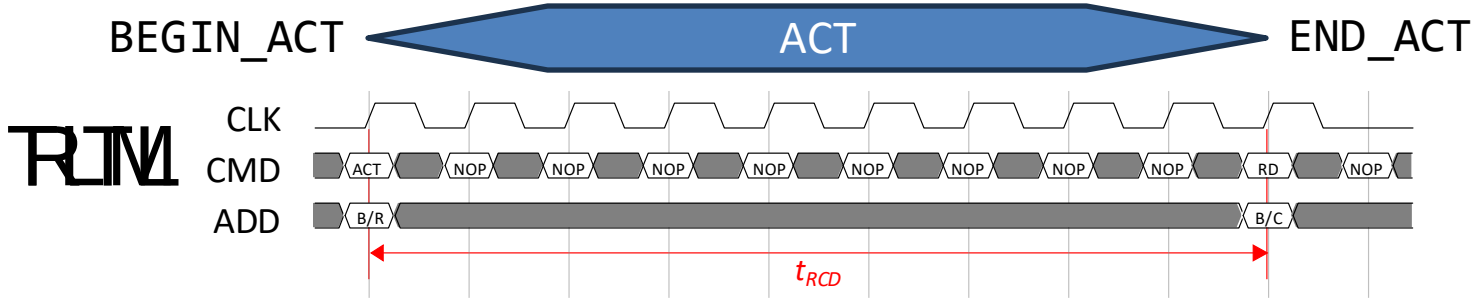
- Based on SystemC TLM2, compliant with TLM-AT coding style
- Flexible SW-Architecuture to support various JEDEC DRAM standards (e.g., DDR4, LPDDR4, GDDR6, HBM, …)
- For RTL-like accuracy a custom TLM protocol (DRAM-AT) is used

accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

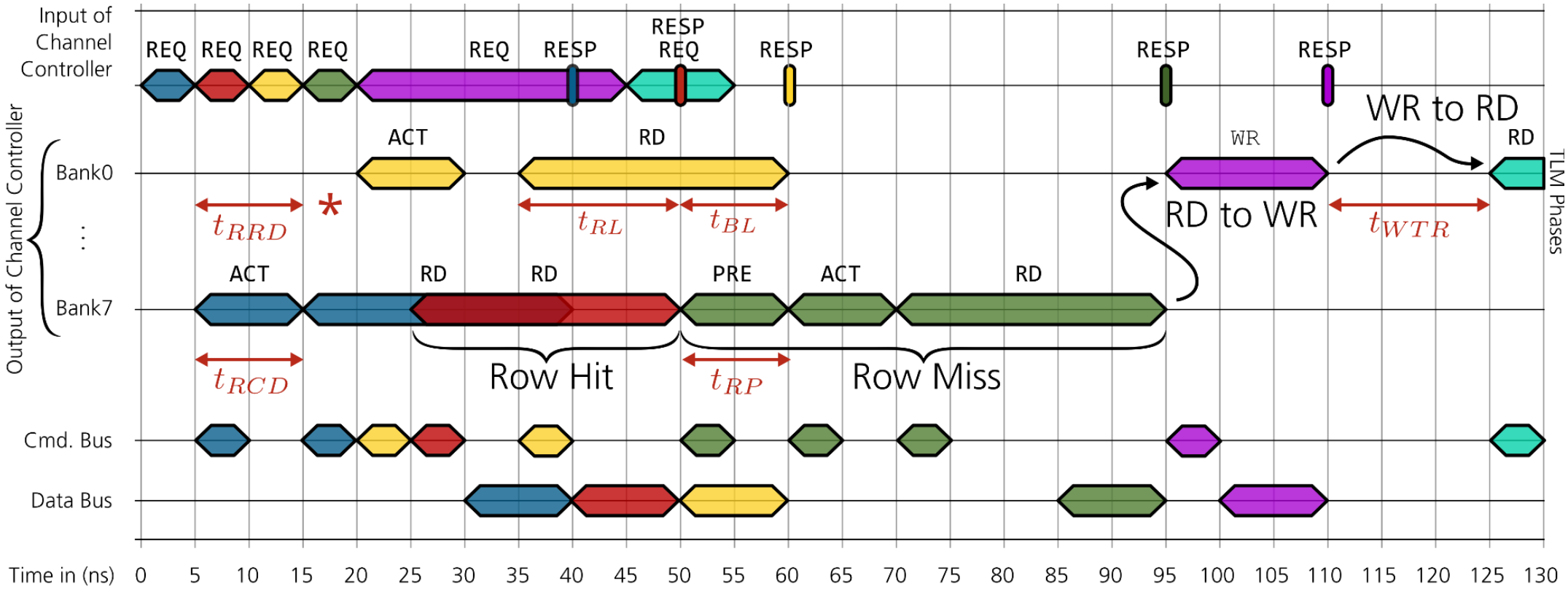
Custom TLM Protocol

- Simulation speed can be increased by reducing the number of events
- Clock signal has the highest event generation rate
- Do we need to simulate each clock cycle to generate cycle-accurate results?



- Simulation of state changes is sufficient, idle clock cycles can be skipped!
  - Large event reduction at low memory access densities
  - No loss of accuracy

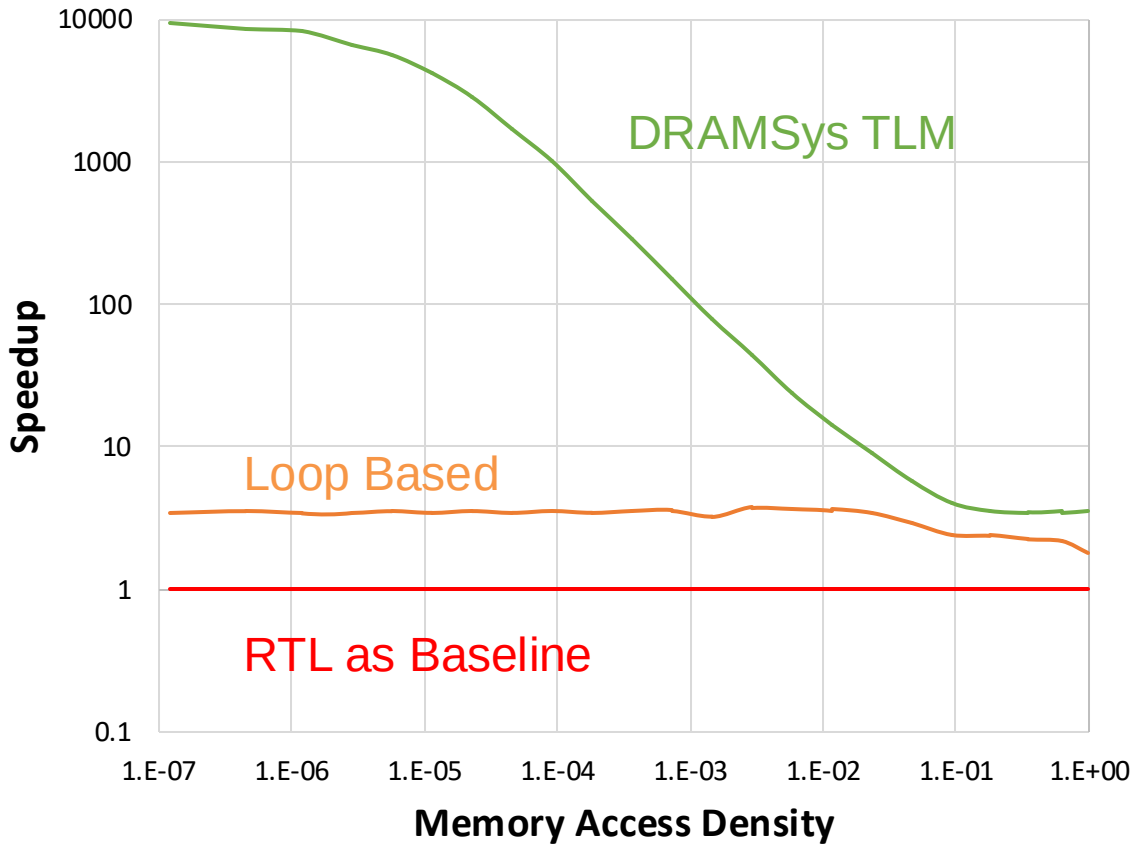
Custom TLM Protocol



accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

DRAMSys Simulation Speed

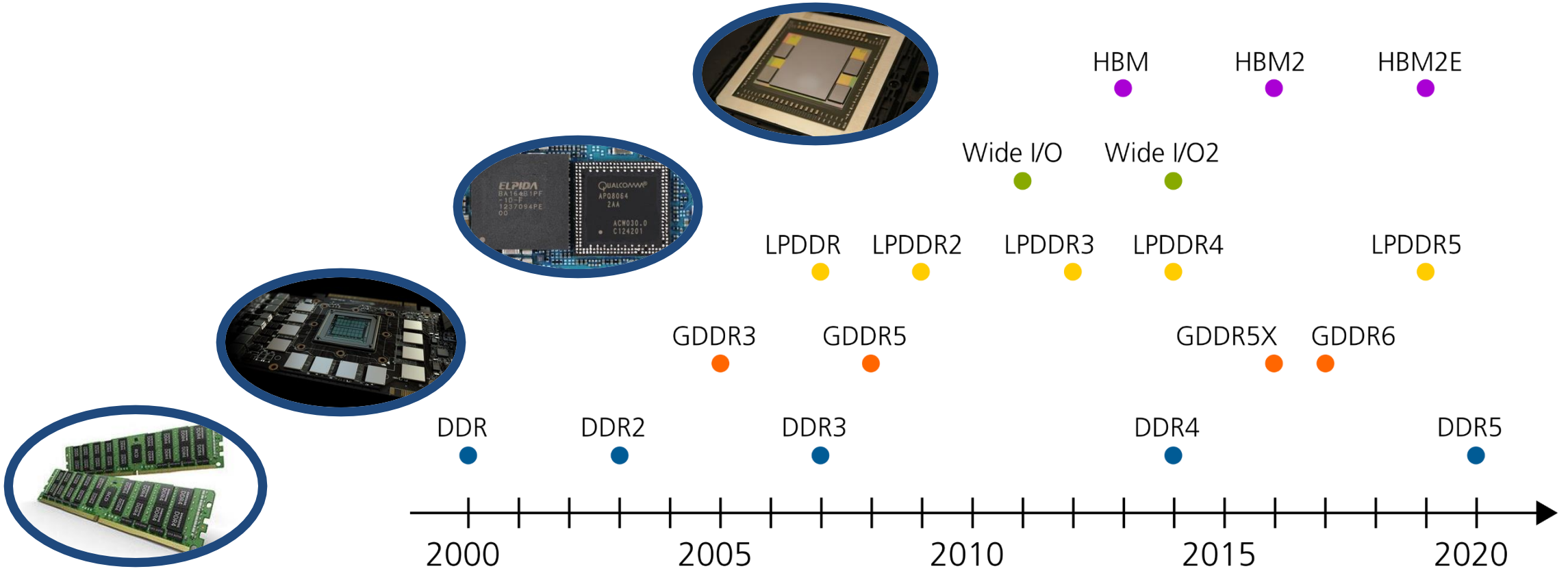


- Simulation of only the important events
- Speedup from 4x to 10.000x depending on trace density
- Average speedups depend on applications
- Typical values: 400x
- 100% RTL Accuracy

accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

Number of DRAM Standards is Growing!



accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

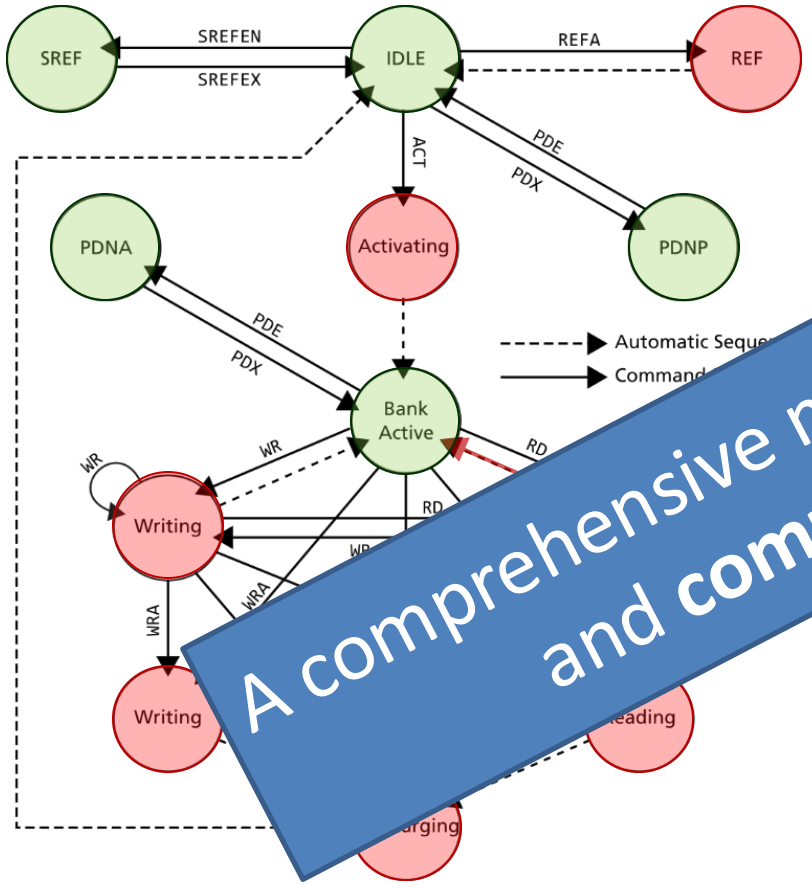
DDR5 JEDEC Standards

- Defines commands, states, timings and interface properties
- Very complex protocol
  - DDR3: 226 pages
  - DDR4: 266 pages
  - DDR5: 496 pages
- Descriptions are not formal
- And not even correct …

accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

JEDEC Standard Description “State M…ne”



DDR3 JEDEC Sta…

“This simplified S… …ew of the possible state transitions and the command… …nvolving more than one bank, the enabling or disab… …nts are not captured in full detail.”

A comprehensive mathematical model of DRAM states and **commands** and **timing** is needed!

- …k shown i.e. no bank parallelism (because of state …sion)

  States like *Activating*, *Precharging, REF …* do not exist (There are only 5 state types!)
- Double States (2x *Reading* and *Writing*)
- Inconsistencies using automatic sequences (eg. Reading state)

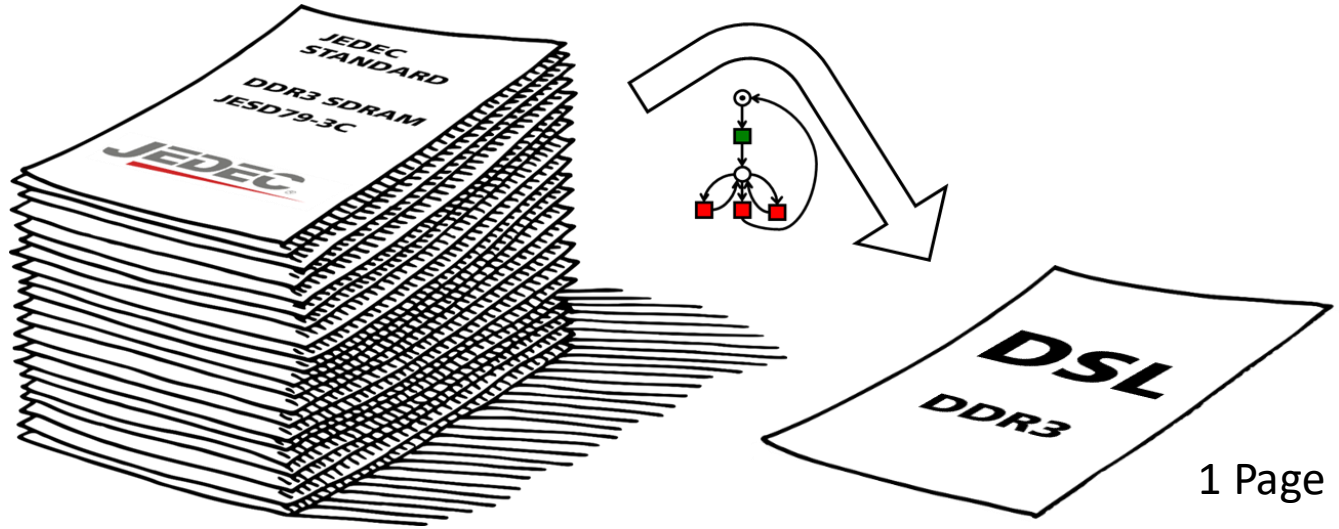
accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

DRAMml: a formal Description for JEDEC Standards

The ideal case: A *formal* language, which has the power to …

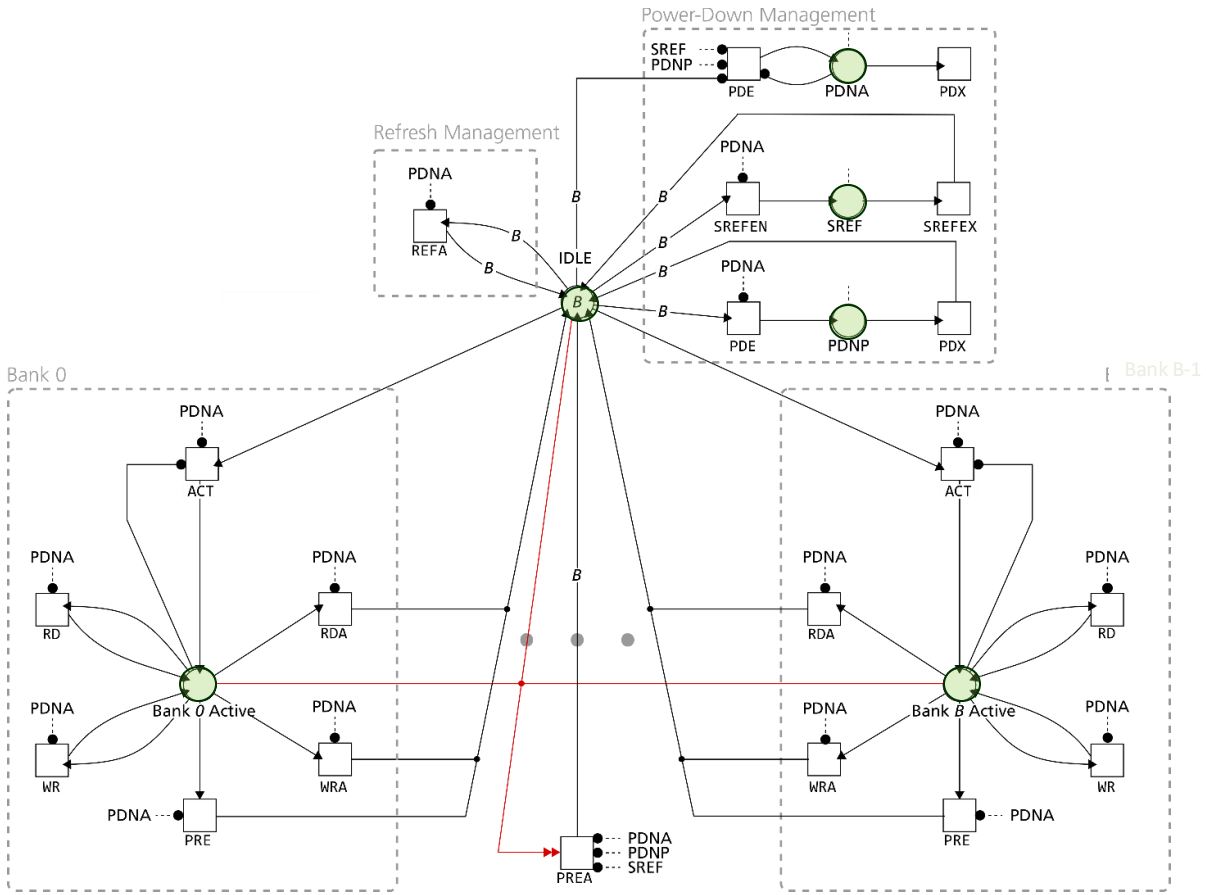
Information of ~100 Pages in the Standard



A new standard requires a serious amount of handcraft:

- New models for fast simulation and verification
- Adapt memory models and HW IP every time

Modeling All DRAM States and Commands

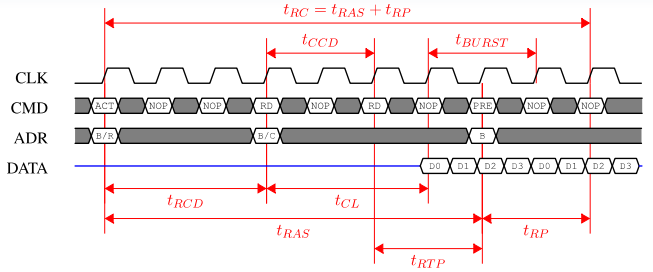


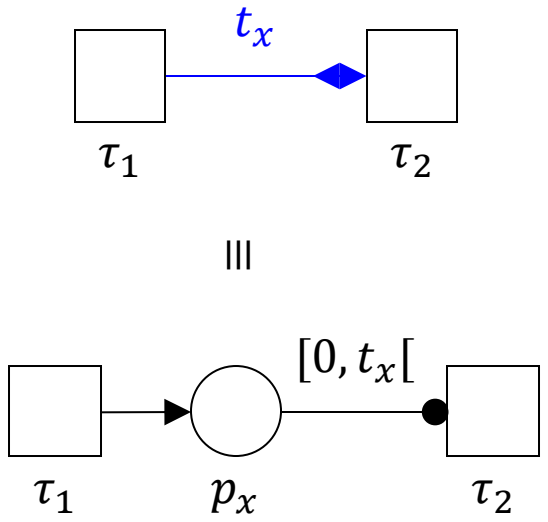
- Comprehensive Model with clear separation between states and commands
- Models only 5 state types
- Support of multiple banks i.e. bank parallelism
- Divided in several subnets:
  - Banks
  - Refresh
  - Power-Down
  - Bankgroups
  - Ranks

accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

Modeling DRAM Timing



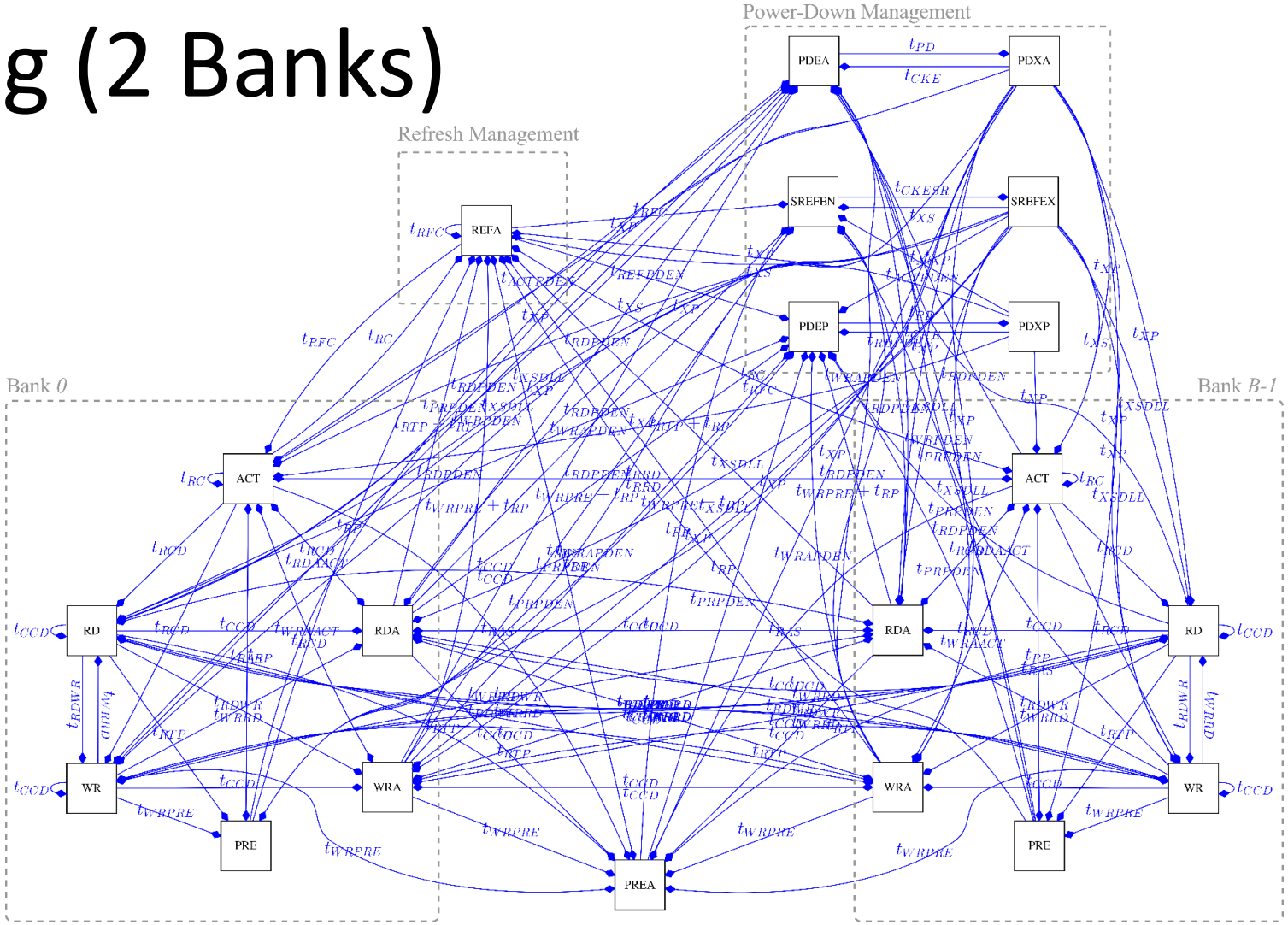


- DRAMs feature a complex timing protocol
- E.g. 90 out of 260 pages of DDR3 standard are showing timing diagrams and explanations for the timings.
- DRAM command timing dependencies can be modeled by a timed inhibitor arc:
- For example, ACT to RD would be t_{RCD}

accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

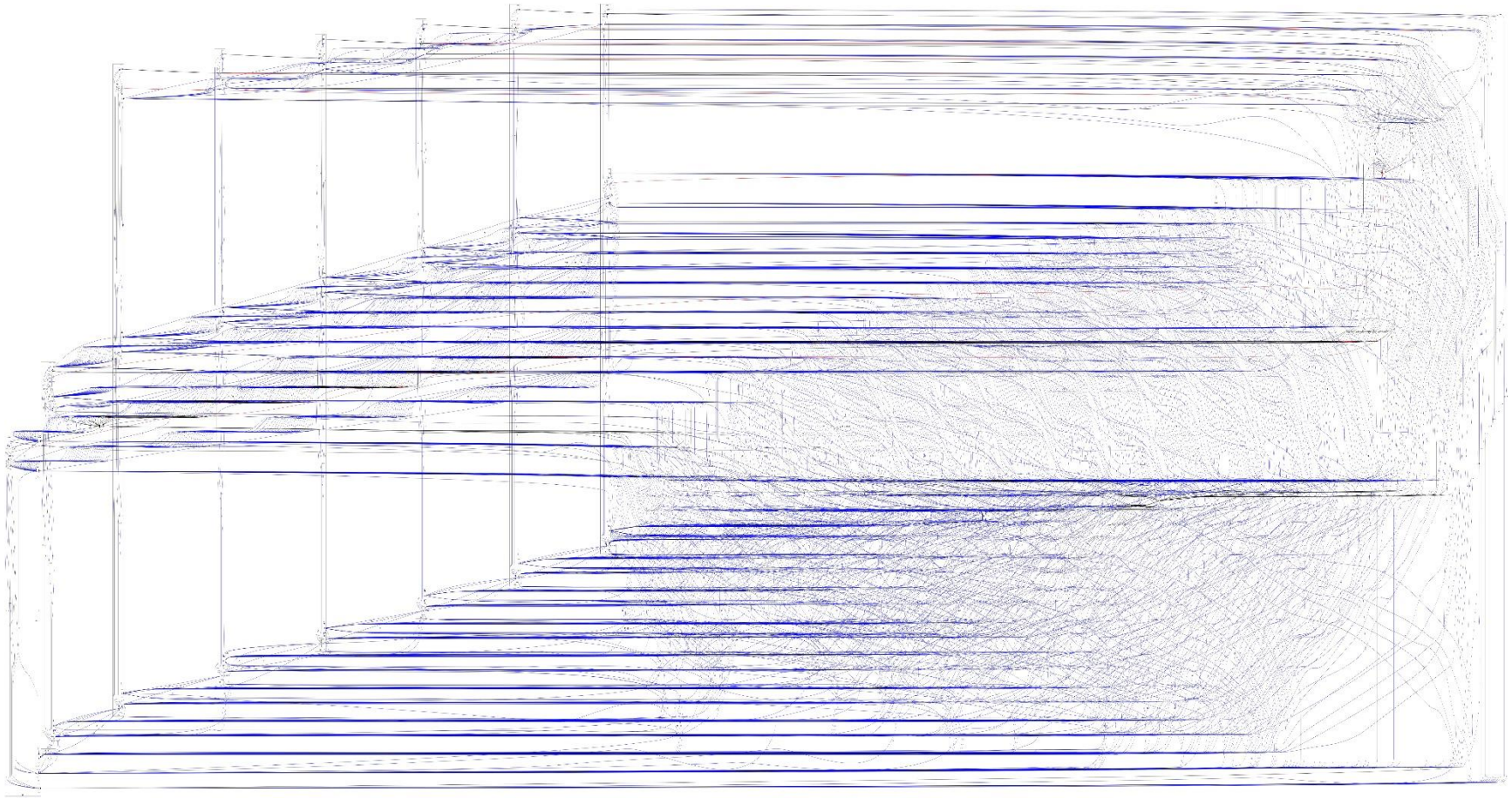
Modeling Timing (2 Banks)



accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

Modeling DRAM Timing (8 Banks)

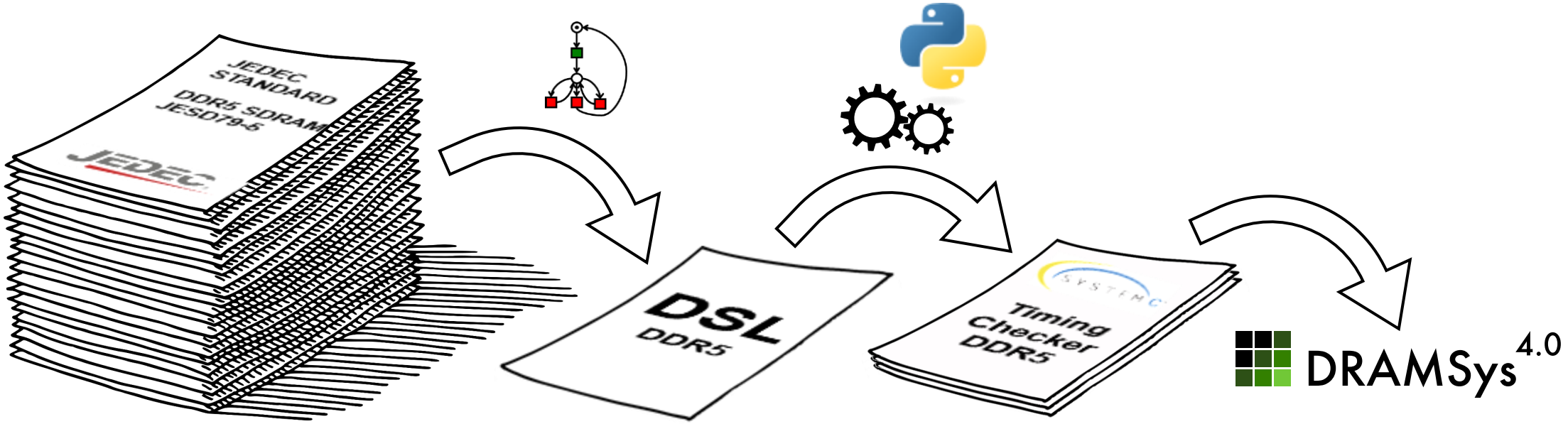


accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

Code Generation and Validation

- Timed Petri nets allow a formal representation of a DRAM protocol, however, no graphical handling possible
- DRAMml is a DSL to describe DRAM's behavior with a petri net semantic
- DSL is as a basis for correct-by-construction DRAMSys TLM code generation
- For example: from DDR5 release it took 2 weeks to implement the model



DDR3

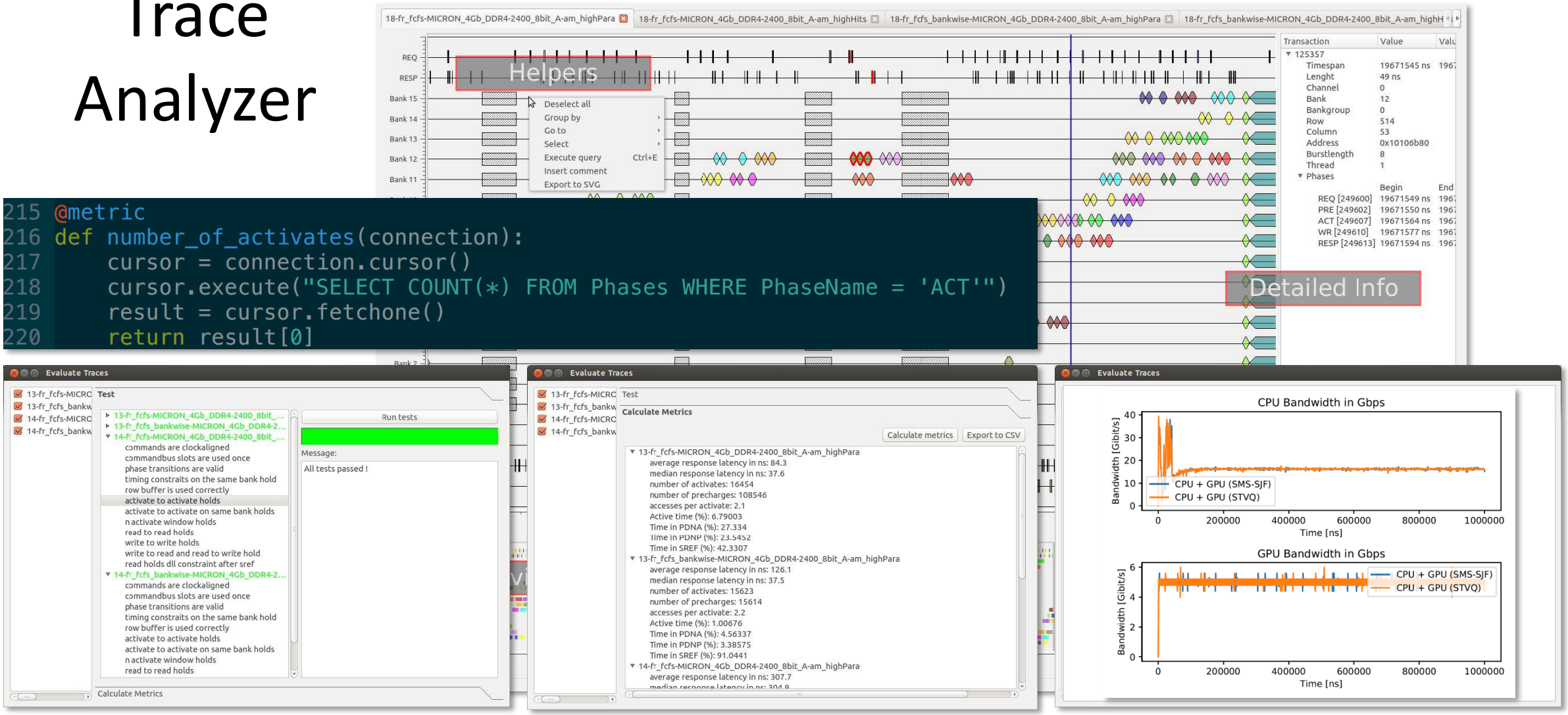
ddr3.py 4.81 KiB

```python
from drampyml.component_levels import *
from drampyml.commands import *
from drampyml.dram import Dram
from drampyml.constants import *
from drampyml.syntax import Max
from drampyml.busses import CommandBus
from drampyml.conditions import *
from drampyml.constraints import *

tBURST = Variable("tBURST", defaultBurstLength / dataRate * tCK)
tRDWR = Variable("tRDWR", tRL + tBURST + tCK * 2 - tWL)
tRDWR_R = Variable("tRDWR_R", tRL + tBURST + tRTRS - tWL)
tWRRD = Variable("tWRRD", tWL + tBURST + tWTR - tAL)
tWRRD_R = Variable("tWRRD_R", tWL + tBURST + tRTRS - tRL)
tWRPRE = Variable("tWRPRE", tWL + tBURST + tWR)
tRDPDEN = Variable("tRDPDEN", tRL + tBURST + tCK)
tWRPDEN = Variable("tWRPDEN", tWL + tBURST + tWR)
tWRAPDEN = Variable("tWRAPDEN", tWL + tBURST + tWR + tCK)

custom_timings = [
    tBURST,
    tRDWR,
    tRDWR_R,
    tWRRD,
    tWRRD_R,
    tWRPRE,
    tRDPDEN,
    tWRPDEN,
    tWRAPDEN,
]

# fmt: off
command_timing_constraints = [
    # Bank
    CommandTimingConstraint(Bank, [ACT], [PREPB], tRAS),
    CommandTimingConstraint(Bank, [ACT], [RD, WR, MWR, RDA, WRA, MWRA], tRCD - tAL),
    CommandTimingConstraint(Bank, [ACT], [ACT], tRC),
    CommandTimingConstraint(Bank, [RD], [PREPB], tAL + tRTP),
    CommandTimingConstraint(Bank, [RD], [WR, MWR, WRA, MWRA], tRDWR),
    CommandTimingConstraint(Bank, [RDA], [ACT], tAL + tRTP + tRP),
    CommandTimingConstraint(Bank, [WR, MWR], [PREPB], tWRPRE),
    CommandTimingConstraint(Bank, [WR, MWR], [WR, MWR, WRA, MWRA], tCCD),
    CommandTimingConstraint(Bank, [WR, MWR], [RD], tWRRD),
    CommandTimingConstraint(Bank, [WR, MWR], [RDA], Max(tWRRD, tWRPRE - tRTP - tAL)),
    CommandTimingConstraint(Bank, [WRA, MWRA], [ACT], tWRPRE + tRP),
    CommandTimingConstraint(Bank, [PREPB], [ACT], tRP),

    # Rank
    CommandTimingConstraint(Rank, [ACT], [PREAB], tRAS),
    CommandTimingConstraint(Rank, [ACT], [ACT], tRRD),
    CommandTimingConstraint(Rank, [ACT], [PDEA], tACTPDEN),
    CommandTimingConstraint(Rank, [ACT], [REFAB, SREFEN], tRC),
    CommandTimingConstraint(Rank, [RD], [PREAB], tAL + tRTP),
    CommandTimingConstraint(Rank, [RD, RDA], [PDEA, PDEP], tRDPDEN),
    CommandTimingConstraint(Rank, [RD, RDA], [RD, RDA], tCCD),
    CommandTimingConstraint(Rank, [RD, RDA], [WR, MWR, WRA, MWRA], tRDWR),
    CommandTimingConstraint(Rank, [RDA], [REFAB], tAL + tRTP + tRP),
    CommandTimingConstraint(Rank, [RDA], [PREAB], tAL + tRTP),
    CommandTimingConstraint(Rank, [RDA], [SREFEN], Max(tRDPDEN, tAL + tRTP + tRP)),
    CommandTimingConstraint(Rank, [WR, MWR], [PDEA], tWRPDEN),
    CommandTimingConstraint(Rank, [WRA, MWRA], [PDEA, PDEP], tWRAPDEN),
    CommandTimingConstraint(Rank, [WR, MWR, WRA, MWRA], [WR, MWR, WRA, MWRA], tCCD),
    CommandTimingConstraint(Rank, [WR, MWR, WRA, MWRA], [RD, RDA], tWRRD),
    CommandTimingConstraint(Rank, [WRA, MWRA], [REFAB], tWRPRE + tRP),
    CommandTimingConstraint(Rank, [WRA, MWRA], [PREAB], tWRPRE),
    CommandTimingConstraint(Rank, [WRA, MWRA], [SREFEN], Max(tWRAPDEN, tWRPRE + tRP)),
    CommandTimingConstraint(Rank, [PREPB], [REFAB], tRP),
    CommandTimingConstraint(Rank, [PREPB], [PDEA, PDEP], tPRPDEN),
    CommandTimingConstraint(Rank, [PREPB], [SREFEN], tRP),
    CommandTimingConstraint(Rank, [PREAB], [ACT, REFAB, SREFEN], tRP),
    CommandTimingConstraint(Rank, [PREAB], [PDEP], tPRPDEN),
    CommandTimingConstraint(Rank, [PDEP], [PDXP], tPD),
    CommandTimingConstraint(Rank, [PDEA], [PDXA], tPD),
    CommandTimingConstraint(Rank, [PDXA], [PDEA], tCKE),
    CommandTimingConstraint(Rank, [PDXA], [PDEP], tCKE),
    CommandTimingConstraint(Rank, [PDXP], [REFAB, SREFEN, ACT], tXP),
    CommandTimingConstraint(Rank, [PDXA], [ACT, PREPB, PREAB, RD, RDA, WR, MWR, WRA, MWRA], tXP),
    CommandTimingConstraint(Rank, [REFAB], [ACT, REFAB, SREFEN], tRFC),
    CommandTimingConstraint(Rank, [REFAB], [PDEP], tREFPDEN),
    CommandTimingConstraint(Rank, [SREFEX], [ACT, REFAB, PDEP, SREFEN], tXS),
    CommandTimingConstraint(Rank, [SREFEX], [RD, RDA, WR, MWR, WRA, MWRA], tXSDLL),
    CommandTimingConstraint(Rank, [SREFEX], [SREFEX], tCKESR),

    # Channel
    CommandTimingConstraint(Channel, [RD, RDA], [RD, RDA], tBURST + tRTRS, [], Different(Rank)),
    CommandTimingConstraint(Channel, [RD, RDA], [WR, MWR, WRA, MWRA], tRDWR_R, [], Different(Rank)),
    CommandTimingConstraint(Channel, [WR, MWR, WRA, MWRA], [WR, MWR, WRA, MWRA], tBURST + tRTRS, [], Different(Rank)),
    CommandTimingConstraint(Channel, [WR, MWR, WRA, MWRA], [RD, RDA], tWRRD_R, [], Different(Rank)),
]
# fmt: on

faw_constraints = [FAWConstraint([ACT], Rank, tFAW)]

bus_commands = {
    ACT,
    RD,
    RDA,
    WR,
    WRA,
    MWR,
    MWRA,
    PREPB,
    PREAB,
    PDEP,
    PDEA,
    PDXA,
    PDXP,
    REFAB,
    SREFEX,
    SREFEN,
}

command_bus = CommandBus("Bus", bus_commands)

dram = Dram(
    "DDR3",
    [command_bus],
    command_timing_constraints,
    faw_constraints,
    custom_timings,
)
```

accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

Trace Analyzer



accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA

Fork me on Github!

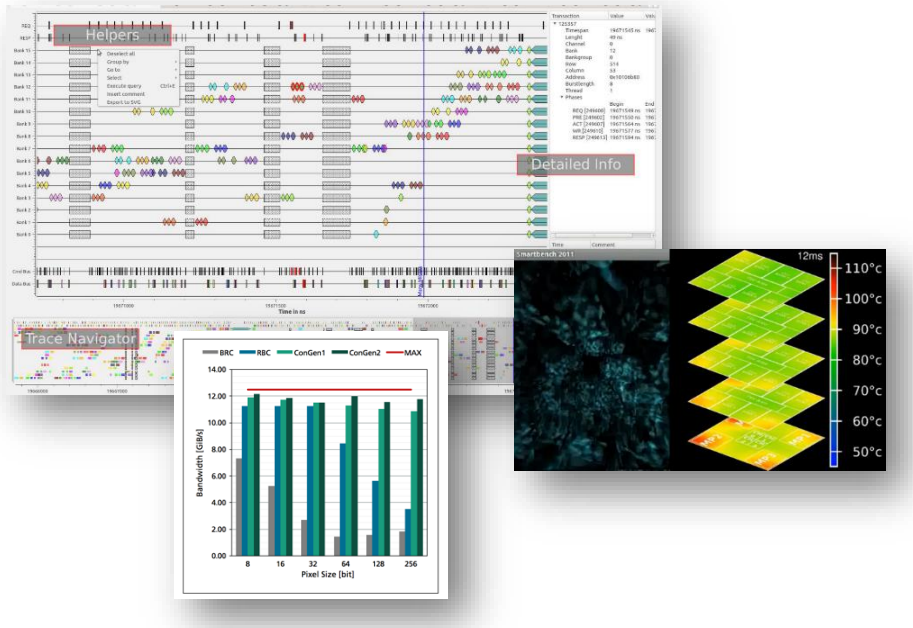
Where to get the tools:

DRAMSys:
https://github.com/tukl-msd/DRAMSys

gem5:
https://gem5.googlesource.com/

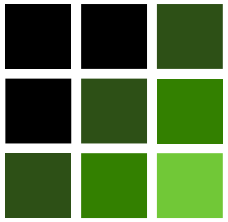
DRAMPower:
https://github.com/tukl-msd/DRAMPower

3D-ICE:
https://github.com/esl-epfl/3d-ice



accellera SYSTEMS INITIATIVE™

SYSTEMC™ EVOLUTION FIKA



DRAMSys

The Open Source DRAM Simulator DRAMSys

Prof. Dr. Matthias Jung, JMU Würzburg

accellera SYSTEMS INITIATIVE™

RPTU Rheinland-Pfälzische Technische Universität Kaiserslautern Landau

Fraunhofer IESE

17 Computer Engineering

Julius-Maximilians-Universität Würzburg

SYSTEMC™ EVOLUTION FIKA