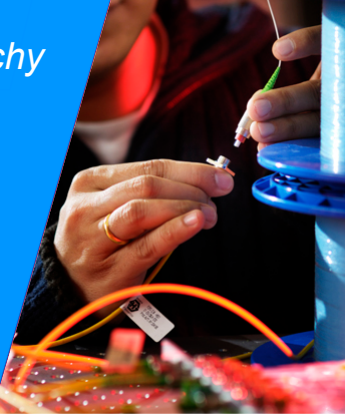


*Time-predictable memory hierarchy
and SDRAM controller*


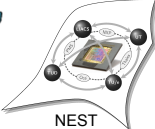



Kees Goossens
and the CompSOC team
at TU Eindhoven, TU Delft, and CISTER



TU/e Technische Universiteit
Eindhoven
University of Technology

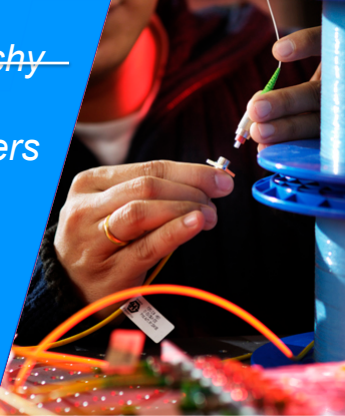
Where innovation starts

Kees Goossens <k.g.w.goossens@tue.nl>
Electronic Systems Group
Electrical Engineering Faculty



~~*Time-predictable memory hierarchy
and*~~
mixed-criticality SDRAM controllers

Kees Goossens
and the CompSOC team
at TU Eindhoven, TU Delft, and CISTER



TU/e Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

Kees Goossens <k.g.w.goossens@tue.nl>
Electronic Systems Group
Electrical Engineering Faculty

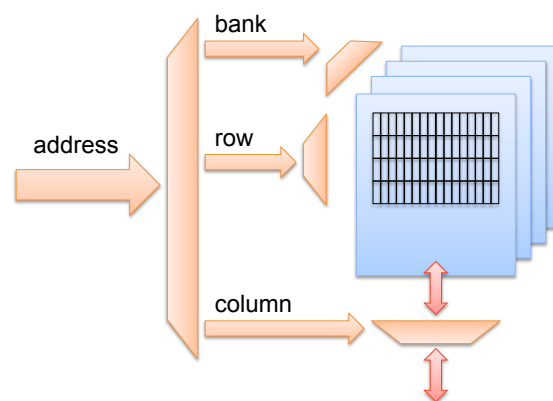
outline

2

- SDRAM basics & performance
- NRT –RT performance
- NRT – RT – RT+NRT memory controllers

SDRAM architecture

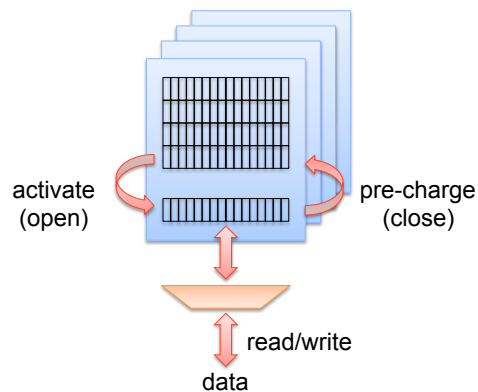
3



SDRAM architecture

4

- can read & write only to **open** rows

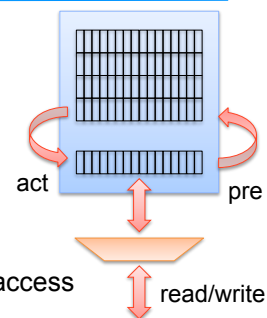


access time

all data for DDR3-800, x16, BC=1, BI=4, 64B data accesses;
faster, wider memories fare worse

6

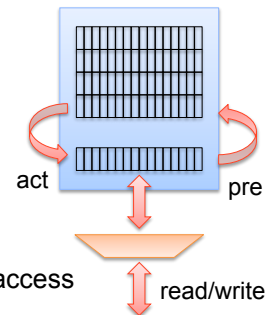
- depends on **address locality**
 - bank: can be kept open simultaneously
 - row
 - column
- worst case:
 - no locality → precharge, activate, R/W for every access
 - bandwidth is at most **16%** of maximum



access time

7

- depends on **address locality**
 - bank: can be kept open simultaneously
 - row
 - column
- worst case:
 - no locality → precharge, activate, R/W for every access
 - bandwidth is at most **16%** of maximum
- best case:
 - maximum locality → activate, R/W, R/W, ...
 - **98%** bandwidth
- real case:
 - aim for 75% (and dropping)



memory controller objectives

8

- real-time (RT):
 - maximise guaranteed bandwidth [and/or]
 - minimise guaranteed latency
- non real-time (NRT):
 - minimise average latency [and/or]
 - maximise average bandwidth
- mixed-criticality (RT & NRT):
 - first, guarantee *enough* bandwidth (for RT)
 - and then, minimise average latency (for SRT or NRT)

common memory controller policies

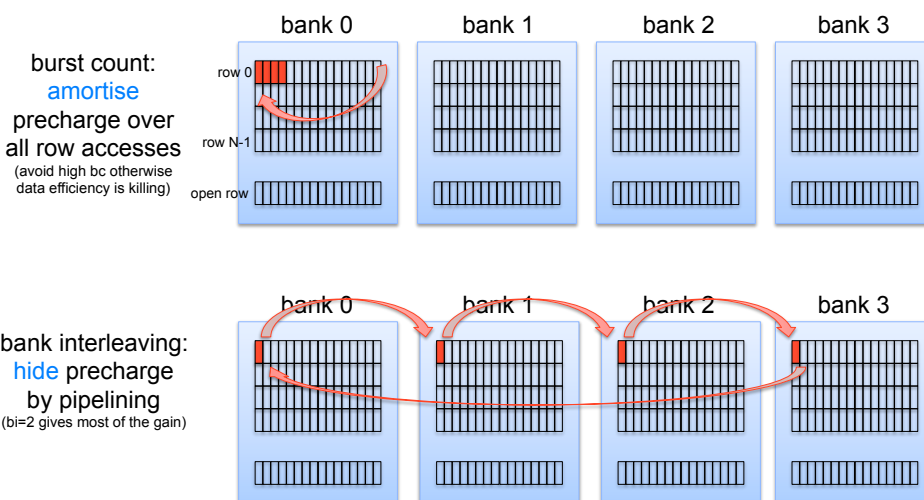
9

- **close page**
 - activate, read/write, pre-charge
 - leave the memory in a known state for the next request
 - row access: 37 cycles, 16 of which are data (BC=1, BI=4, write)
 - good when locality is low, or when it **cannot be assumed**
- **open page**
 - pre-charge & activate when necessary, read/write
 - leave current row open
 - row miss: 15 cycles
 - row hit: 4 cycles
 - good when locality is high

make these numbers
correspond to the 37
cycles above

access time & mapping

[assume continuous memory map] 10



memory controller strategies

11

NRT dyn.
controller

non real time

assume &
optimise
for locality

© Kees Goossens
Electronic Systems

ARPA
2013-01-22

TU/e Technische Universiteit
Eindhoven
University of Technology

memory controller strategies

12

static memory
controller

real time

cannot assume locality
optimise for max. bw.

NRT dyn.
controller

non real time

assume &
optimise
for locality

© Kees Goossens
Electronic Systems

ARPA
2013-01-22

TU/e Technische Universiteit
Eindhoven
University of Technology

memory controller strategies

13

static memory controller	RT Predator	conservative open page	speculative open page	RT+NRT open page	NRT dyn. controller
--------------------------	-------------	------------------------	-----------------------	------------------	---------------------

© Kees Goossens
Electronic Systems

ARPA
2013-01-22

TU/e Technische Universiteit
Eindhoven University of Technology

memory controller strategies

14

static memory controller	RT Predator	conservative open page	speculative open page	RT+NRT open page	NRT dyn. controller
--------------------------	-------------	------------------------	-----------------------	------------------	---------------------

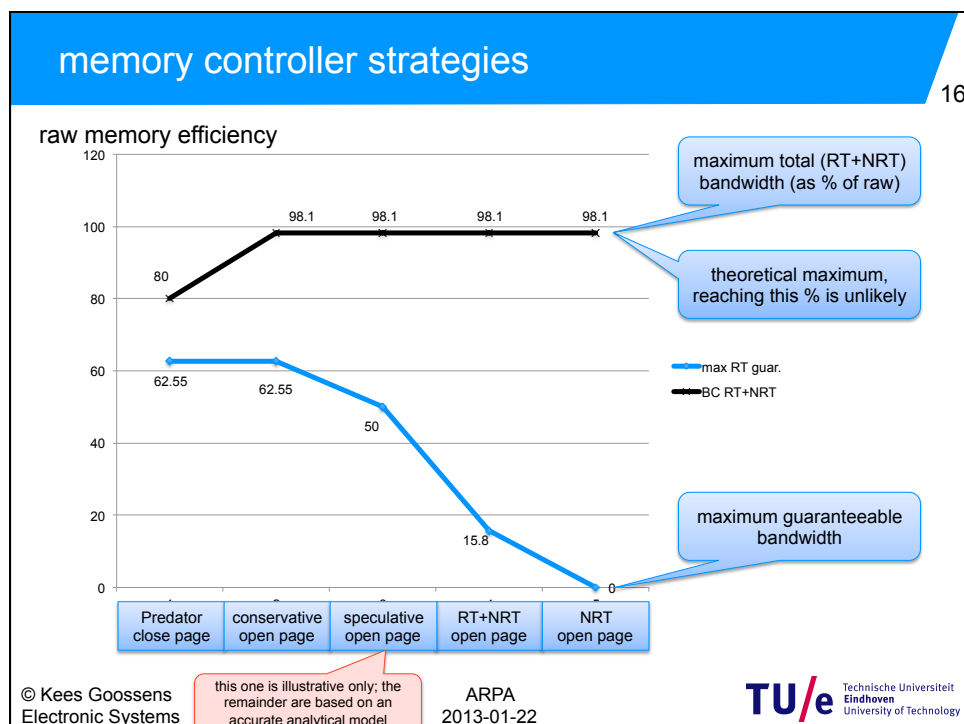
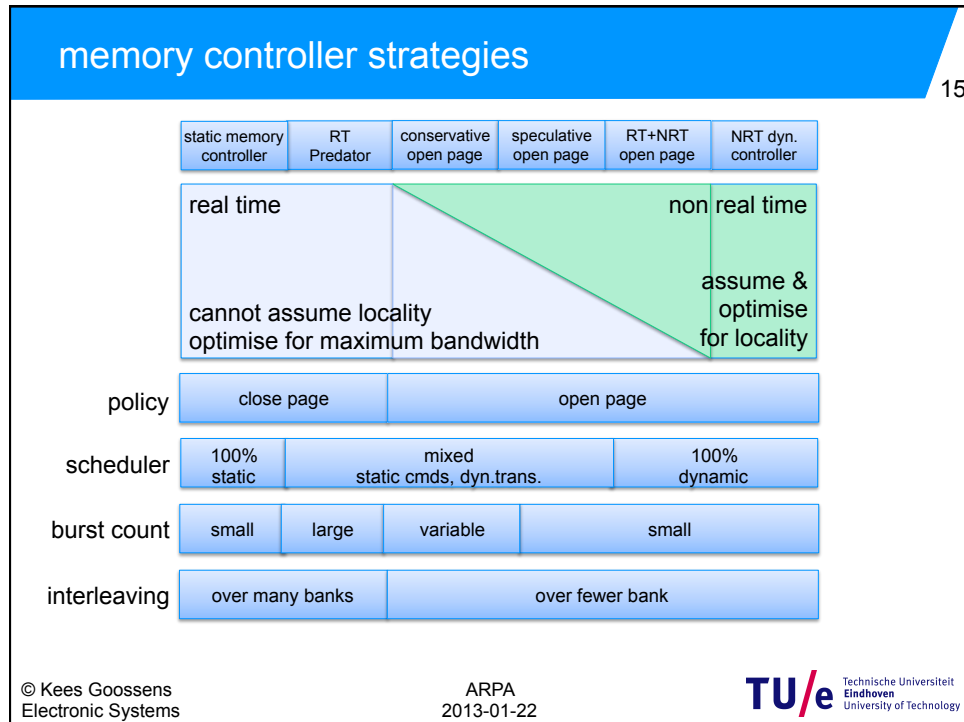
policy	close page	open page
--------	------------	-----------

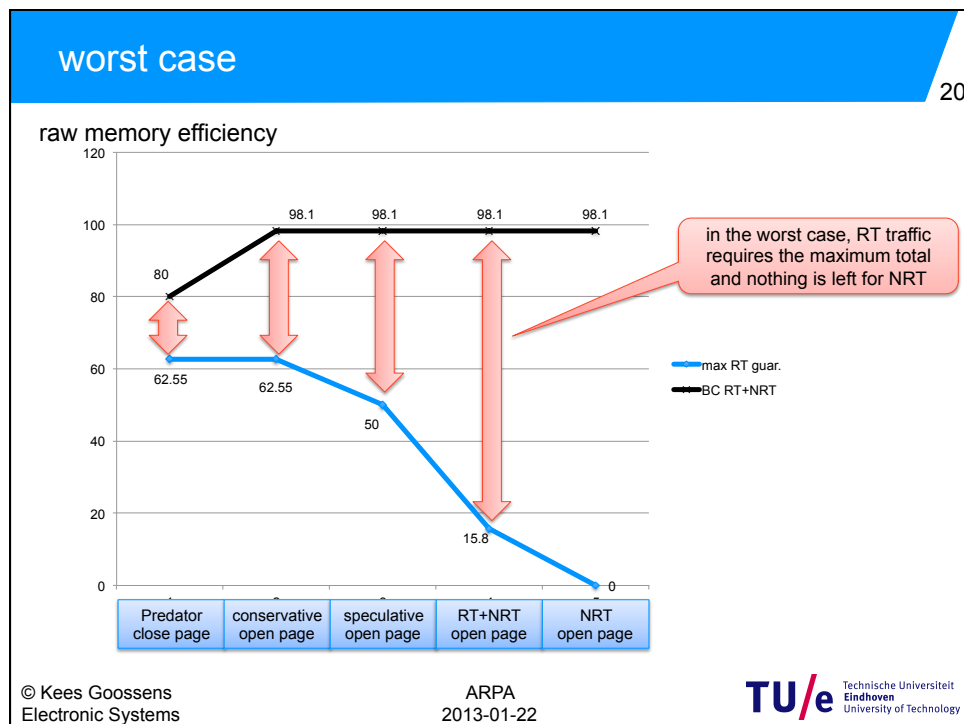
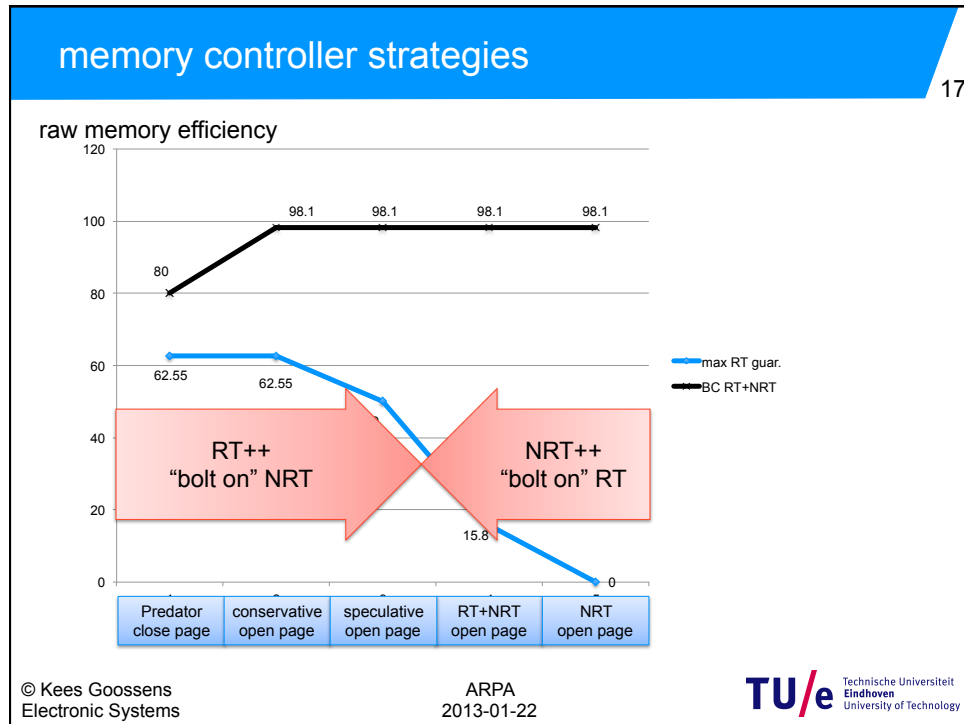
scheduler	100% static	mixed static cmds, dyn.trans.	100% dynamic
-----------	-------------	-------------------------------	--------------

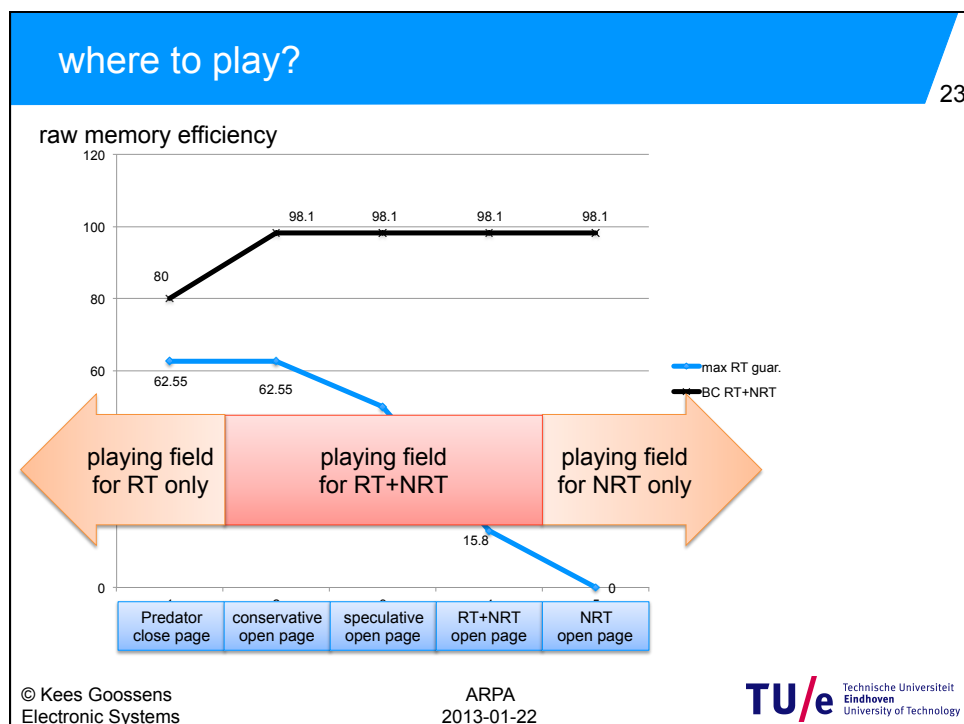
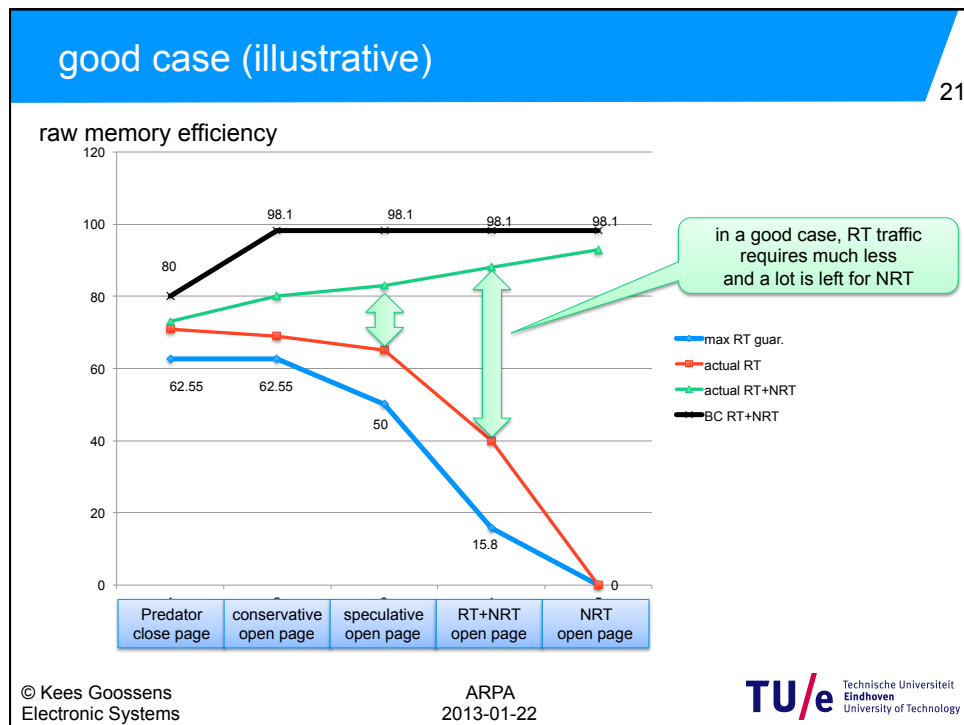
© Kees Goossens
Electronic Systems

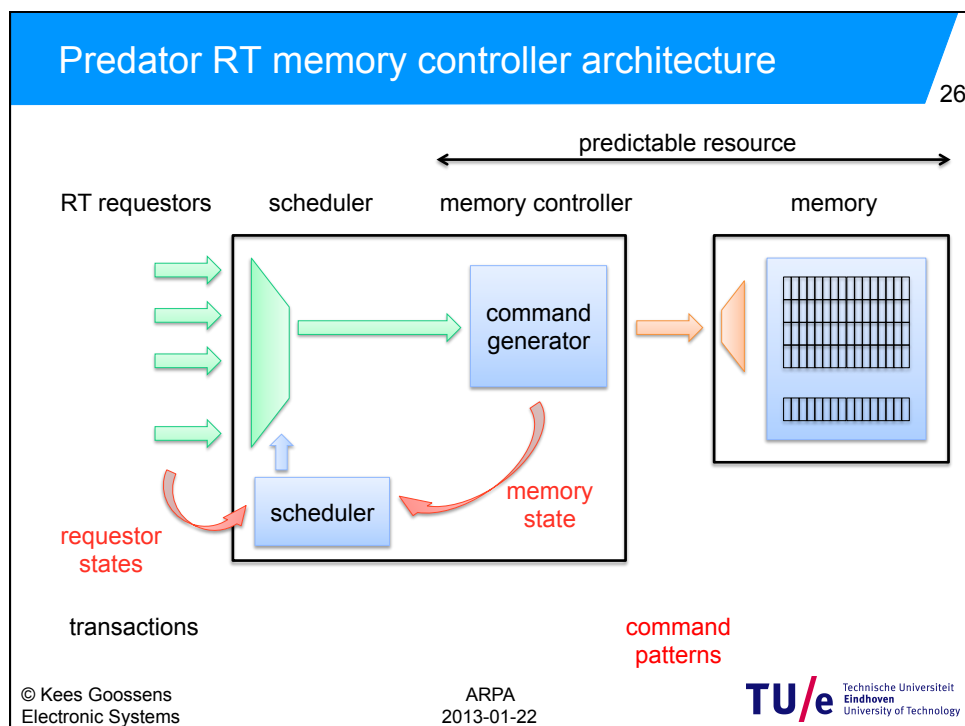
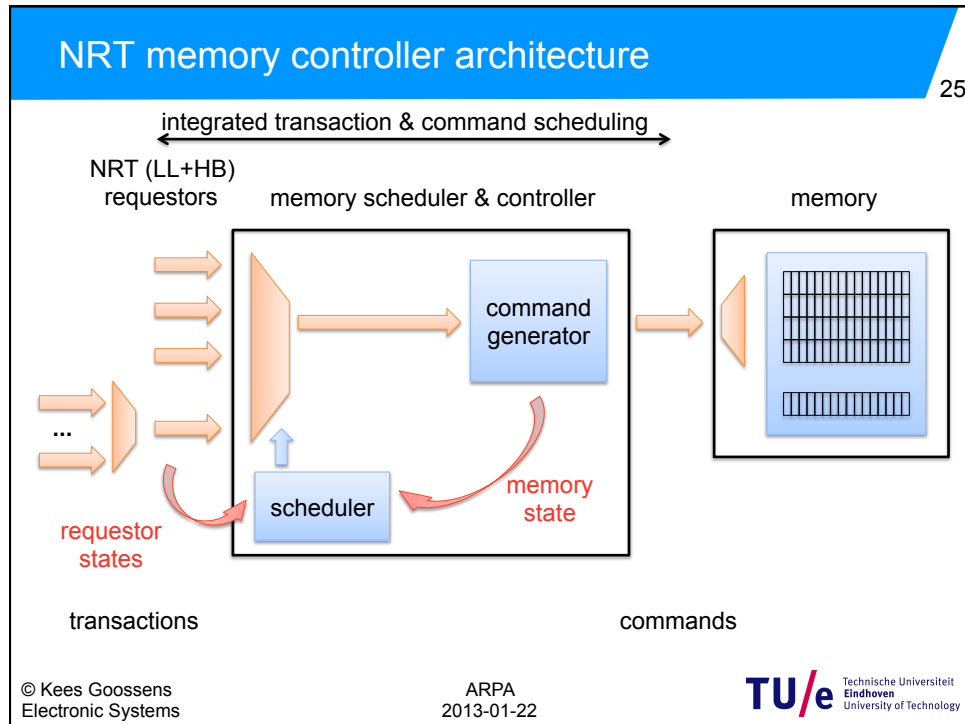
ARPA
2013-01-22

TU/e Technische Universiteit
Eindhoven University of Technology



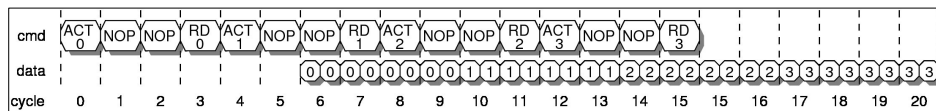






Predator RT memory controller architecture

27

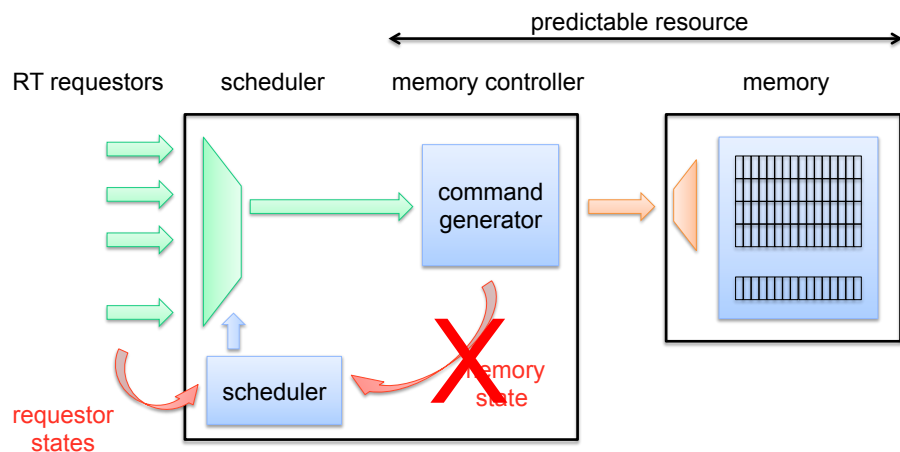


transactions

command patterns

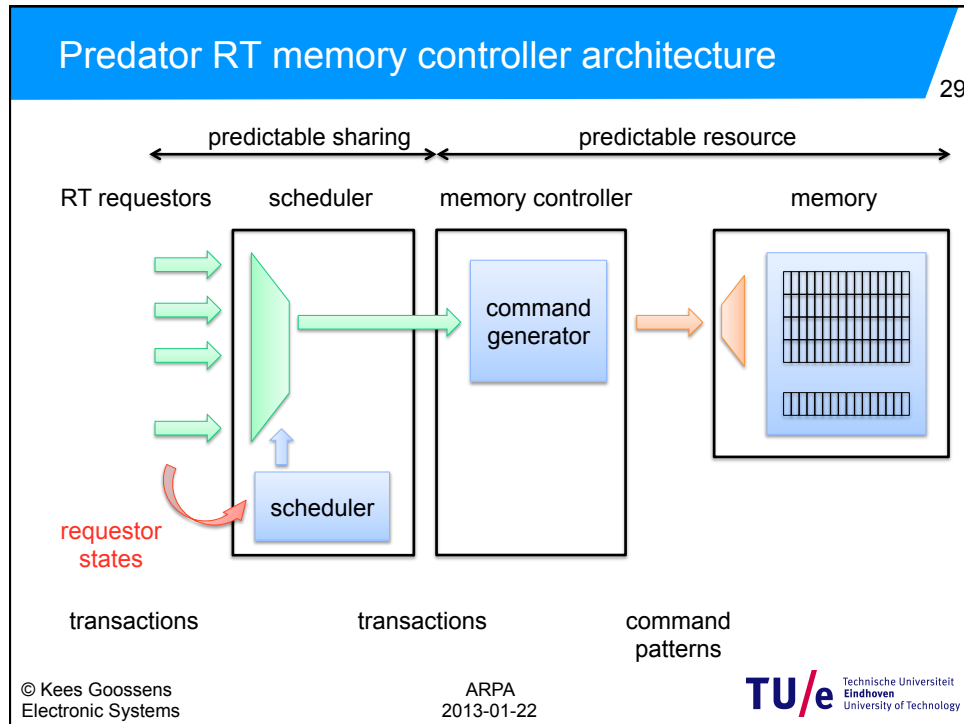
Predator RT memory controller architecture

28



transactions

command
patterns



T-CREST – focussing on real time platform

33

- a RT memory controller is not enough
- RT interconnect (NOC)
- RT processor & RTOS
- programming model
- etc.
- T-CREST focus on real-time:
 - minimise the worst case, not the average case!
- leveraging the experience of CompSOC platform

© Kees Goossens
Electronic Systems

ARPA
2013-01-22

TU/e Technische Universiteit
Eindhoven
University of Technology

conclusions

34

- building a RT or NRT memory controller is fun (except for the PHY)
- but building **RT+NRT** memory controller requires more trade offs:
 - guaranteed – average
 - bandwidth – latency – power
 - burst count, bank interleaving, command patterns, open/close page, static/dynamic (non)-work-conservative scheduling, etc.
- key is to **trade RT efficiency for better NRT locality**
- this is possible when traffic is a mix of RT and NRT
- “just” bolting on RT on a NRT controller is very limited (16%)
- higher RT guarantees require restriction on access patterns
 - be careful to not pay in terms of data efficiency

TUE acknowledgements **bold = memory team**

35

- [Eindhoven university of technology](#)
 - Kees Goossens (team leader)
 - **Sven Goossens**
 - **Manil Dev Gomony**
 - Martijn Koedam
 - **Yonghui Li**
 - Shubhendu Sinha
 - Jun Zhu
- [CISTER, Porto](#)
 - **Benny Akesson**
- [Delft university of technology](#)
 - **Karthik Chandrasekar**
 - Davit Mirzoyan
 - Ashkan Beyranvand Nejad
 - Andrew Nelson
- close collaboration with the [dataflow research \(SDF3\)](#) at TU/e
 - Sander Stuijk
 - Marc Geilen
- and many others

This research is supported by EU grants
FP7 288008 **T-CREST**, FP7 288248 **FlexTiles**,
CA104 **Cobra**, CA505 **BENEFIC**, &
NL grant STW10346 **NEST**.
Parts of the platform were developed with support
from COMCAS, Scalopes, TSAR, NEVA, MESA.



more information: www.compsoc.eu

36

- CompSOC in CRTS'12 and in Multiprocessor System-on-Chip. Huebner (ed), Springer, 2010
- Aethereal real-time NOC, DAC'10
- Predator real-time DRAM memory controller, DATE'11
- CompOSe RTOS, MICPRO'11
- composable power management, SAMOS'11
- SDF3, DAC'06 Stuijk, et al. <http://www.es.ele.tue.nl/sdf3/>

