CompSOC: Virtual execution platforms for mixed-criticality systems

1

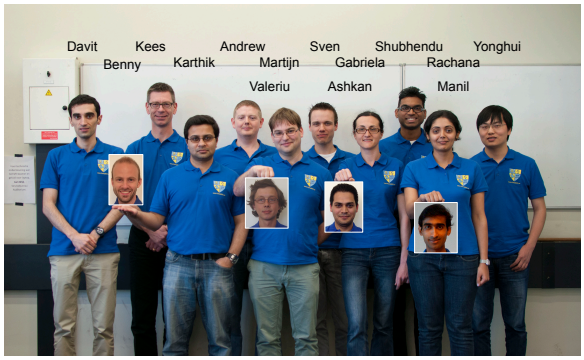
- Cyber-physical, embedded real-time systems usually contain multiple concurrent applications that have different characteristics and requirements, and are often designed by different parties. As a result, a single system contains applications designed using different models of computation (MOC), and with different criticalities (e.g. real time, safety critical, adaptive, or not). By offering an independent execution virtual platform to each application, the CompSOC platform enables independent design, verification, and execution of applications with different criticalities and models of computation. We present the underlying concepts, architecture, and formalism of the CompSOC platform. More information can be found on www.compsoc.eu.

the current CompSOC team

2

- Eindhoven university of technology
  - Kees Goossens (team leader)
  - Gabriela Breaban
  - Valeriu Codreanu
  - Sven Goossens
  - Manil Dev Gomony
  - Martijn Koedam
  - Reinier van Kampenhout
  - Rachana Kumar
  - Yonghui Li
  - Andrew Nelson
  - Shubhendu Sinha
  - Rasool Tavakoli
  - Juan Valencia
- Prague University
  - Benny Akesson
- close collaboration with the dataflow research (SDF3) at TU/e
  - Marc Geilen
  - Sander Stuijk
  - Dip Goswami
  - Majid Nabi
- and many others

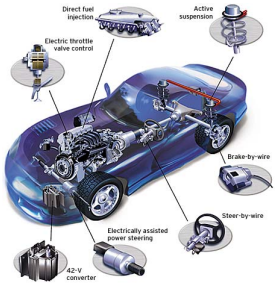
This research is supported by EU grants FP7 288008 **T-CREST**, FP7 288248 **FlexTiles**, CA505 **BENEFIC,** CA703 **OpenES**, ARTEMIS-2013-1 621429 **EMC2** and 621353 **DEWI.** Parts of the platform were developed with support from COMCAS, Cobra, Scalopes, TSAR, NEST, NEVA, MESA.



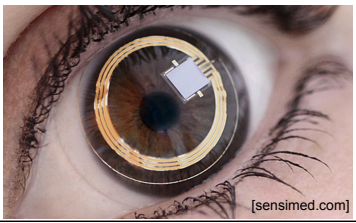
trend: embedded systems

3

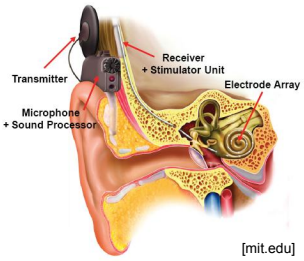
- phones, game consoles, cars, refrigerators, buildings, ...
- interaction with physical world ➔ real-time requirements



[wikipedia.org]



[sensimed.com]

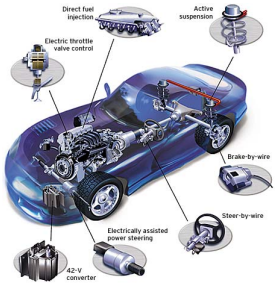


[mit.edu]

trend: embedded systems

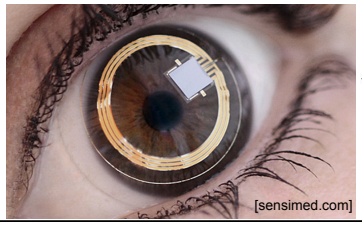
4

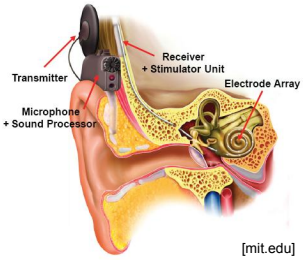
- phones, game consoles, cars, refrigerators, buildings, ...
- interaction with physical world ➔ real-time requirements



[wikipedia.org]

[gadgetsin.com]



[sensimed.com]

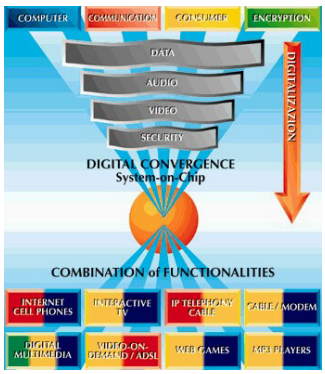


[mit.edu]

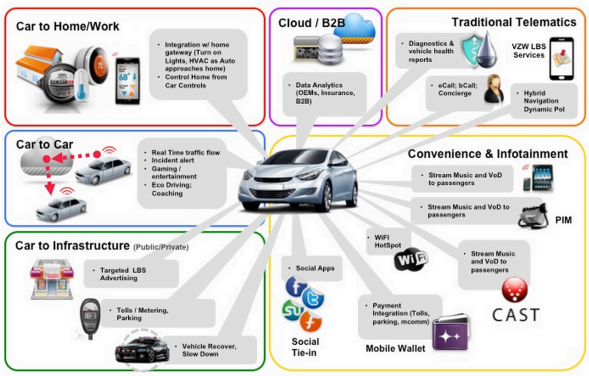
trend: multiple applications on one embedded device

5

- different application domains
  - infotainment (streaming → video analysis), games
  - real-time control, (wireless) networking stacks



[st.com]

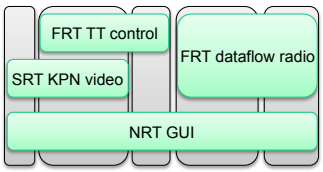


[carvoyant.com]

trend: multiple applications on one embedded connected device

6

- different application domains
  - infotainment (streaming → video analysis), games
  - real-time control, (wireless) networking stacks
- use cases
- sharing resources



observation: different applications

7

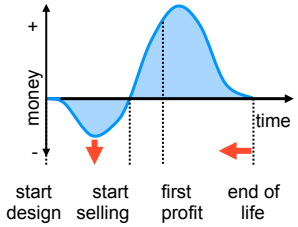
- have different characteristics and requirements
  - FRT, SRT, NRT, adaptive, ...
- are designed by different parties
- use different models of computation (MOC) & verification approaches
  - FRT → worst case & formal verification
  - NRT → average case & simulation
  - adaptive SRT → actual case & simulation
- use different resource-management policies
  - data / event / time, schedulers, power management

problem: design time

8

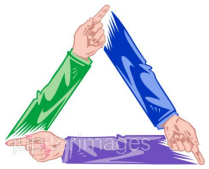
- systems are complex
- time is short

- it takes too long
- getting worse



- monolithic verification after integration
  - hardware, multiple applications

- circular verification
  - who to blame for errors?



goal & approach

9

- reduce SOC design & verification effort

- independent design, verification, and deployment per application
  - application as the unit of verification

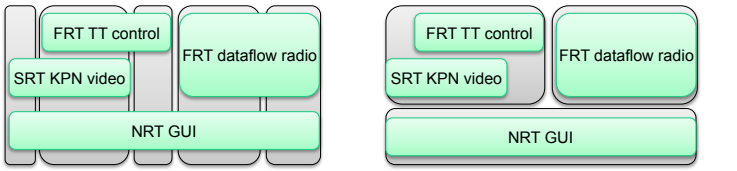
- two complexity-reducing techniques
  - composability
  - predictability

- applied to multi-processor system with distributed shared memory

composability

10

- virtual execution platform per application
  - defined by its space, time, energy budgets
- design, verify, deploy in a virtual platform
- no interference
  - when integrating, executing, or switching use cases

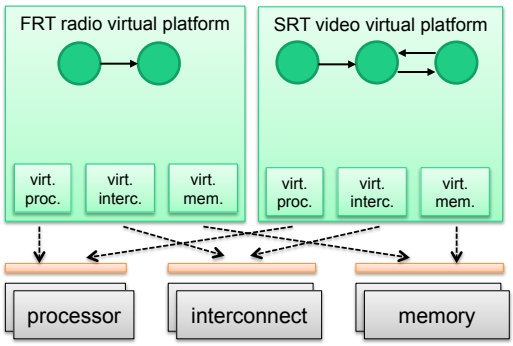


N! combinations & interleavings	➔	N virtual platforms
inter-dependent state $2^S \times 2^S \times 2^S \times 2^S$	➔	independent states $2^S + 2^S + 2^S + 2^S$

composability

11

- time-division-multiplex virtual execution platforms
  - space, time, energy budgets
  - no interference

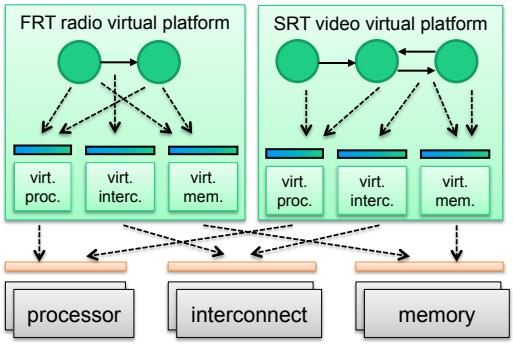


inter-application scheduling
µkernel

composability & predictability

12

- each application can have its own
  - programming model, scheduling, power management
  - (CS)DF, KPN, TT, DSM
- for RT use predictable schedulers, and associated real-time formalism

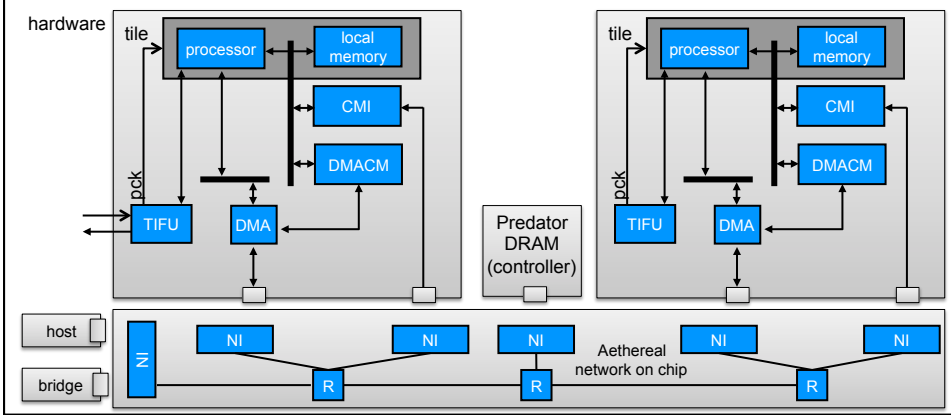


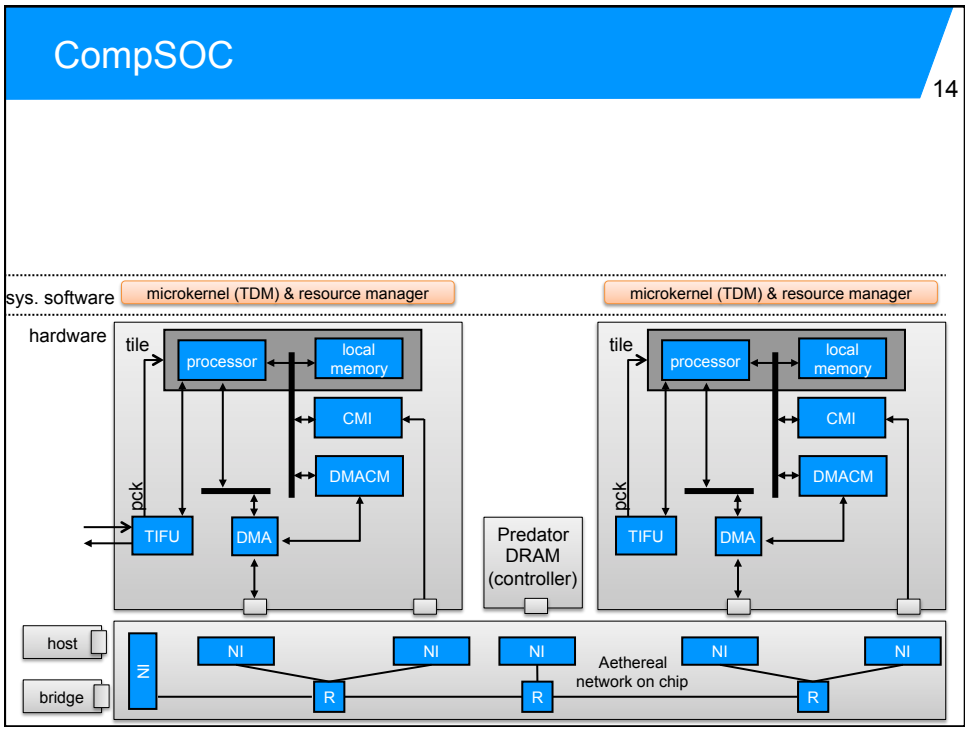
intra-application scheduling & power management

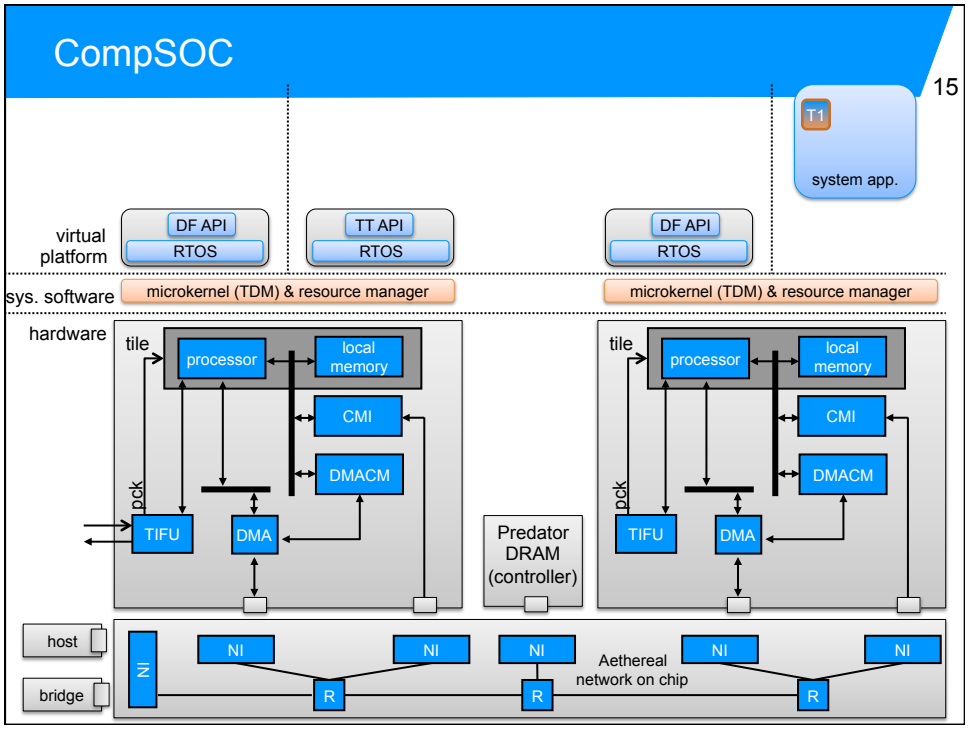
inter-application scheduling

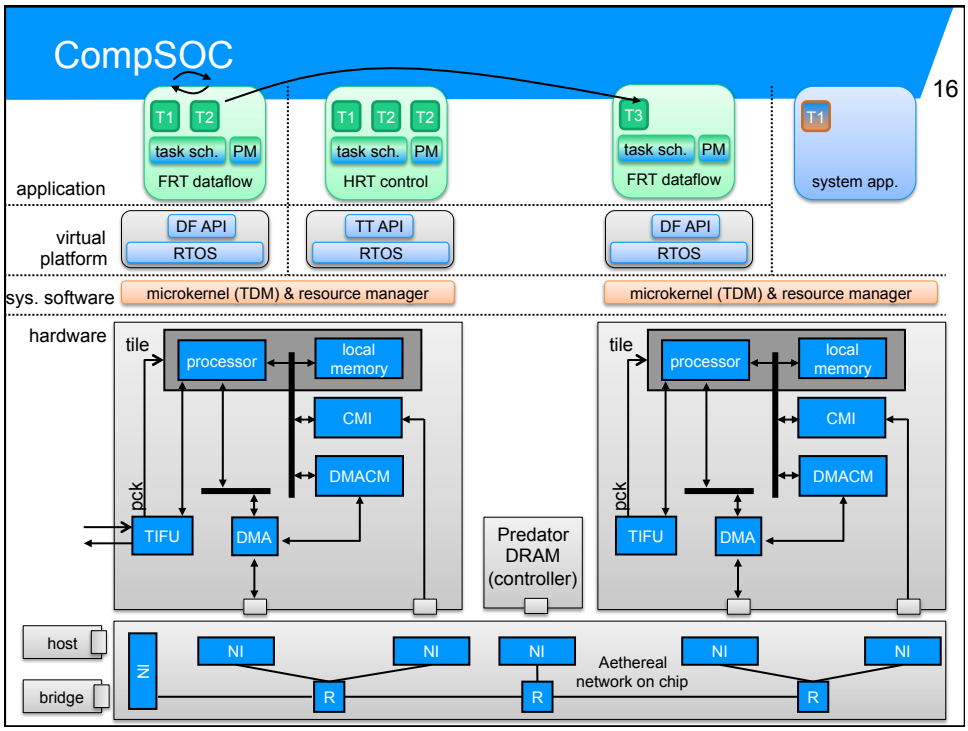
CompSOC

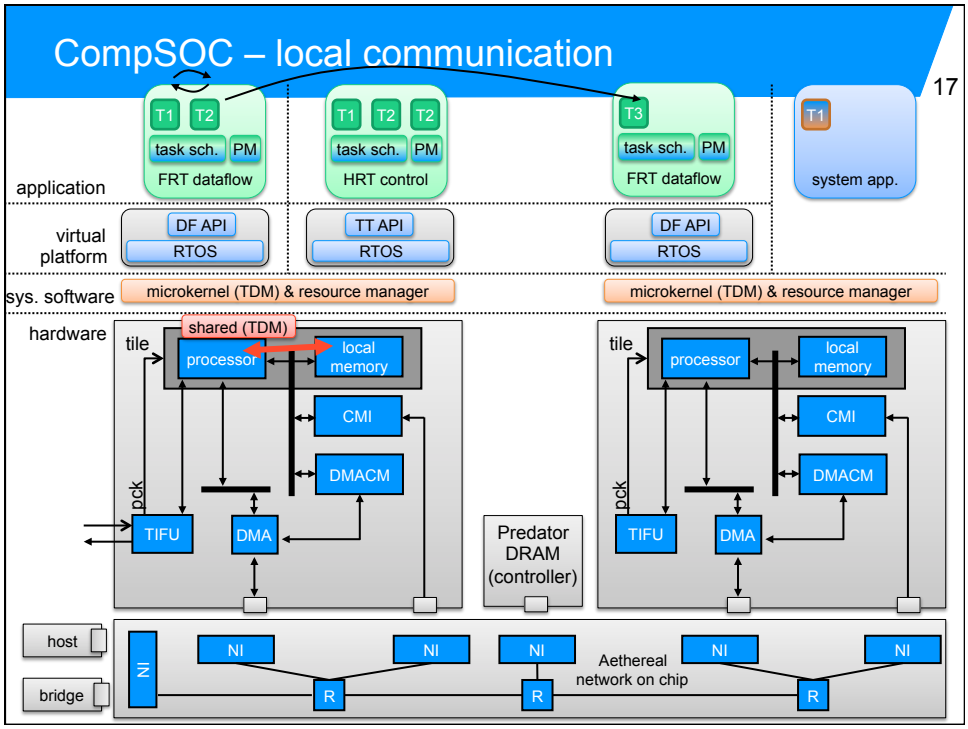
13

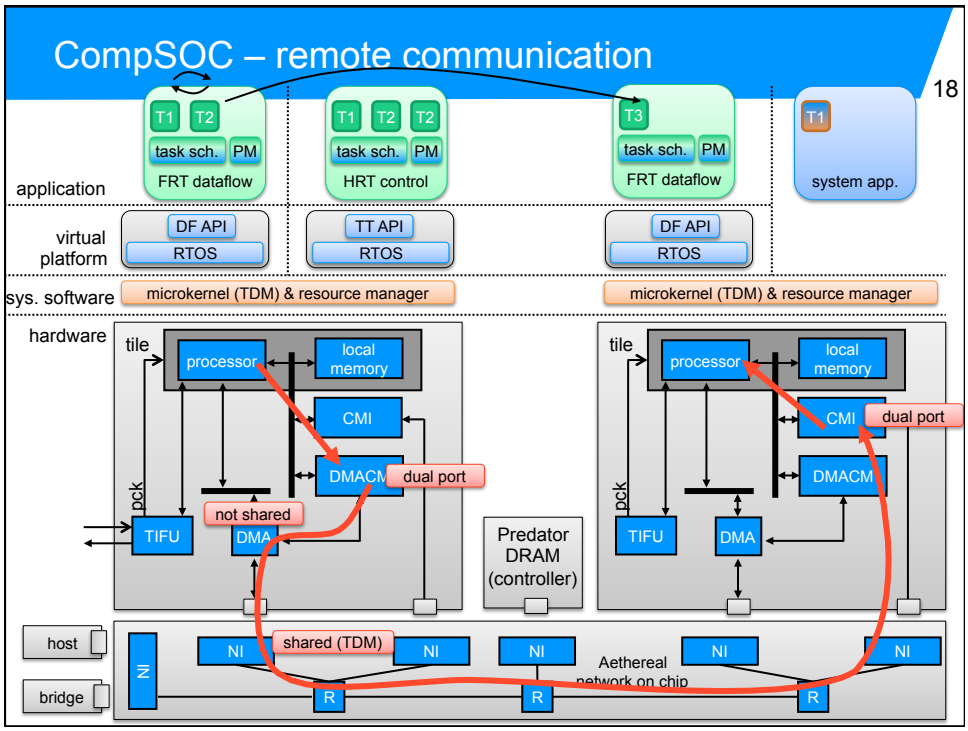


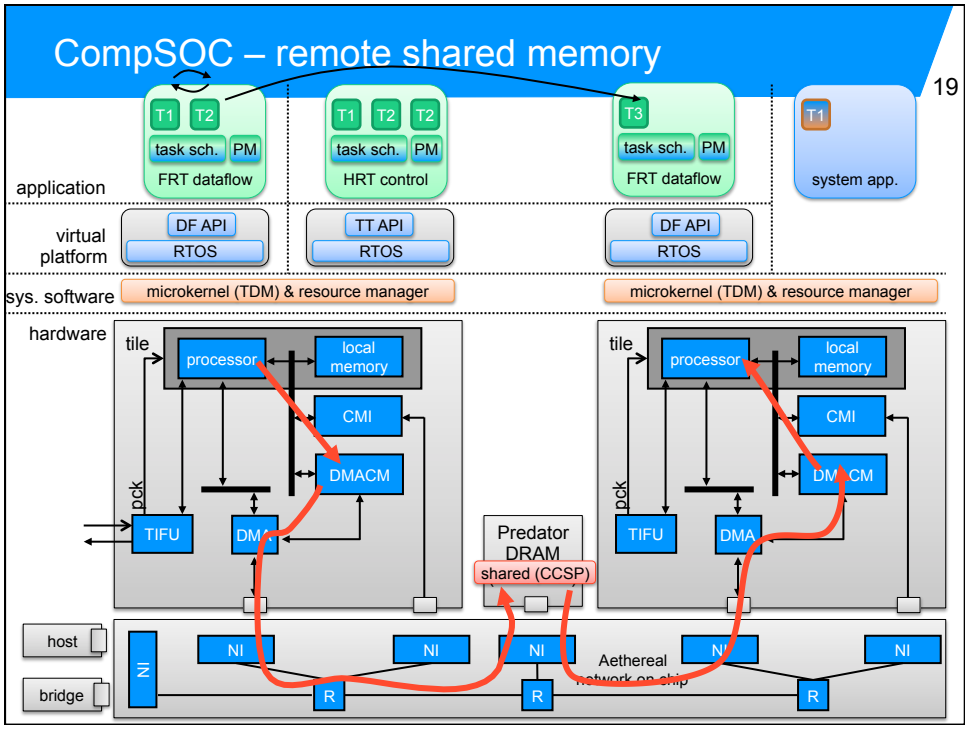

CompSOC
14
sys. software
microkernel (TDM) & resource manager
microkernel (TDM) & resource manager
hardware
tile
processor
local memory
CMI
DMACM
pck
TIFU
DMA
Predator DRAM (controller)
tile
processor
local memory
CMI
DMACM
pck
TIFU
DMA
host
bridge
NI
NI
NI
NI
NI
NI
Aethereal network on chip
R
R
R


CompSOC
15
T1
system app.
virtual platform
DF API
RTOS
TT API
RTOS
DF API
RTOS
sys. software
microkernel (TDM) & resource manager
microkernel (TDM) & resource manager
hardware
tile
processor
local memory
CMI
DMACM
pck
TIFU
DMA
Predator DRAM (controller)
tile
processor
local memory
CMI
DMACM
pck
TIFU
DMA
host
bridge
NI
NI
NI
NI
NI
NI
Aethereal network on chip
R
R
R


CompSOC
16
application
virtual platform
sys. software
hardware
T1
T2
task sch.
PM
FRT dataflow
HRT control
T3
system app.
DF API
TT API
RTOS
microkernel (TDM) & resource manager
tile
processor
local memory
CMI
DMACM
pck
TIFU
DMA
Predator DRAM (controller)
host
bridge
NI
R
Aethereal network on chip

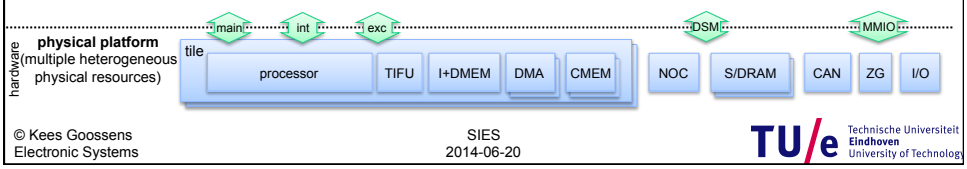

CompSOC – local communication
17
application
virtual platform
sys. software
hardware
shared (TDM)
T1
T2
task sch.
PM
FRT dataflow
HRT control
T3
system app.
DF API
TT API
RTOS
microkernel (TDM) & resource manager
tile
processor
local memory
CMI
DMACM
pck
TIFU
DMA
Predator DRAM (controller)
host
bridge
NI
R
Aethereal network on chip


CompSOC – remote communication
18
application
virtual platform
sys. software
hardware
T1
T2
task sch.
PM
FRT dataflow
HRT control
T3
system app.
DF API
TT API
RTOS
microkernel (TDM) & resource manager
tile
processor
local memory
CMI
DMACM
dual port
not shared
pck
TIFU
DMA
Predator DRAM (controller)
host
bridge
NI
shared (TDM)
R
Aethereal network on chip


CompSOC – remote shared memory
19
application
virtual platform
sys. software
hardware
T1
T2
task sch.
PM
FRT dataflow
HRT control
T3
system app.
DF API
TT API
RTOS
microkernel (TDM) & resource manager
tile
processor
local memory
CMI
DMACM
pck
TIFU
DMA
Predator DRAM
shared (CCSP)
host
bridge
NI
R
Aethereal

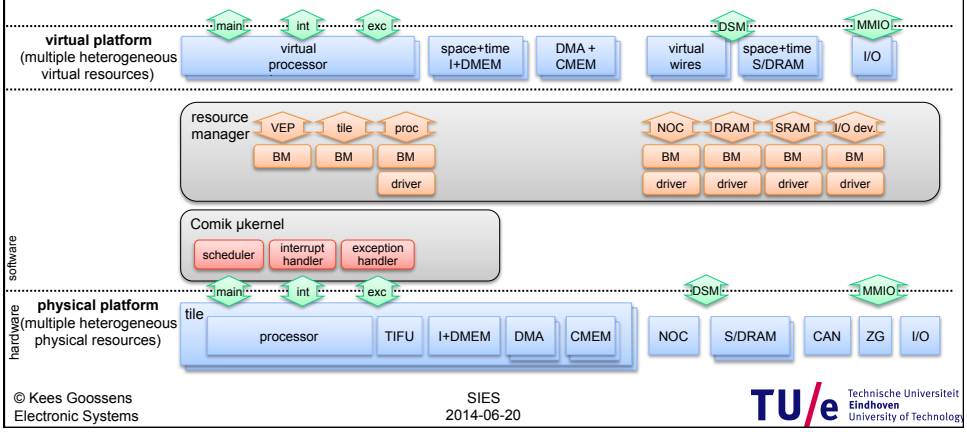
CompSOC software architecture

20



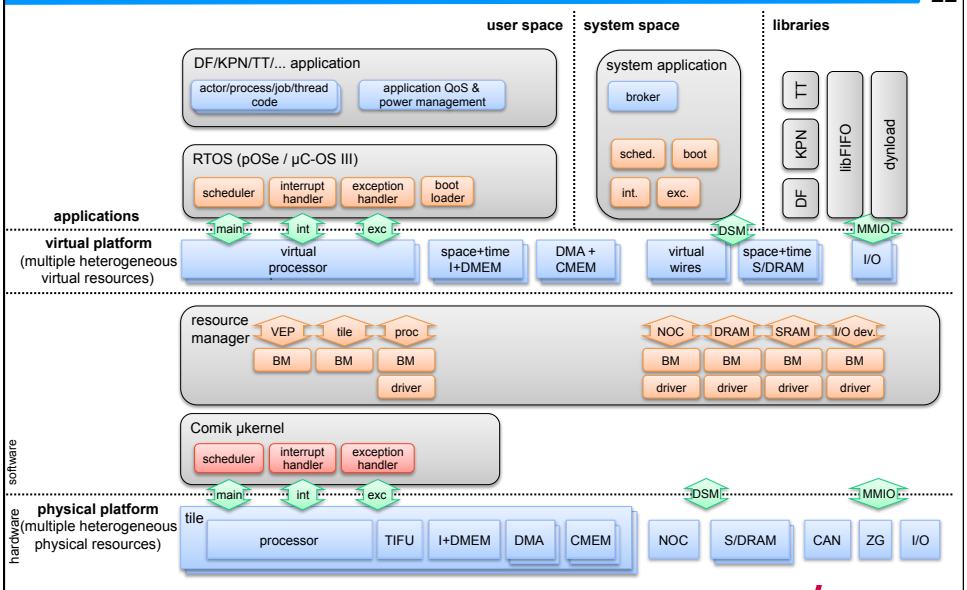
CompSOC software architecture

21



CompSOC software architecture

22



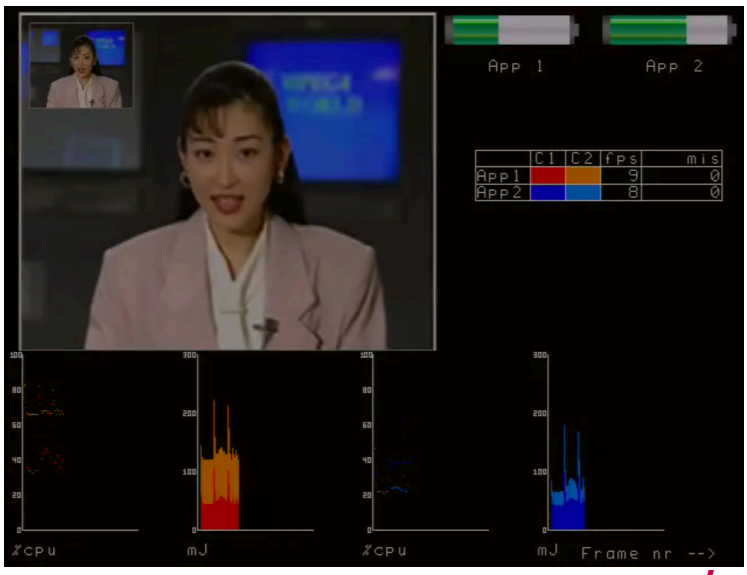
composability

23

- applications can use any programming model
  - cyclo-static dataflow
  - variable-rate dataflow
  - KPN
  - time-triggered
  - distributed-shared memory, with restrictions (see later)
- mixed-criticality platform

demo: 4-battery-run-short.avi [DATE'12]

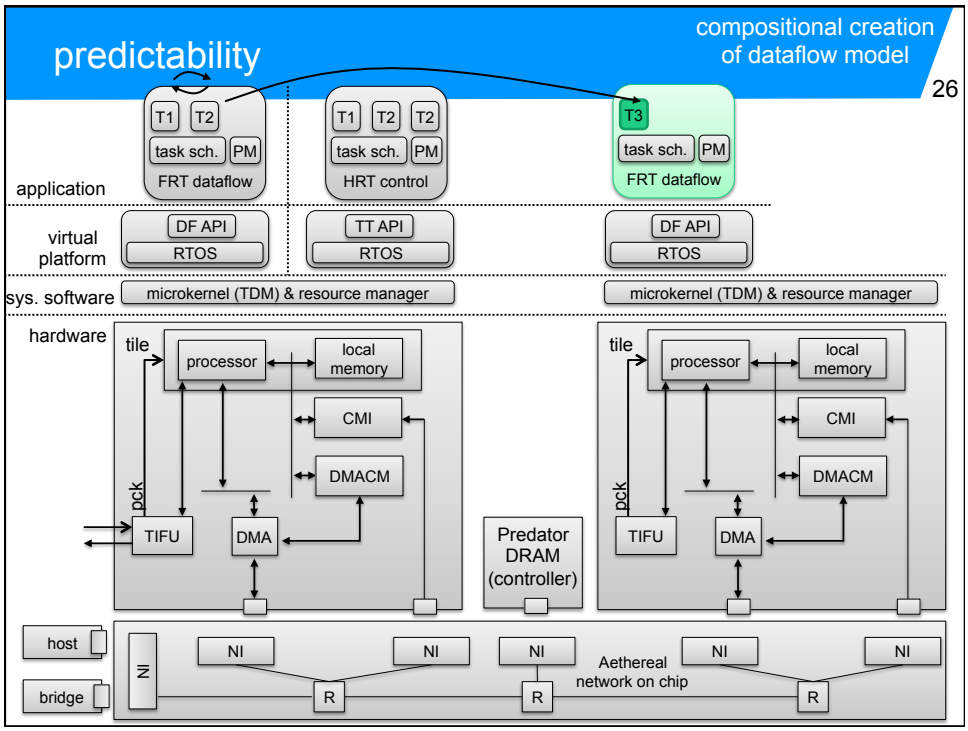
24

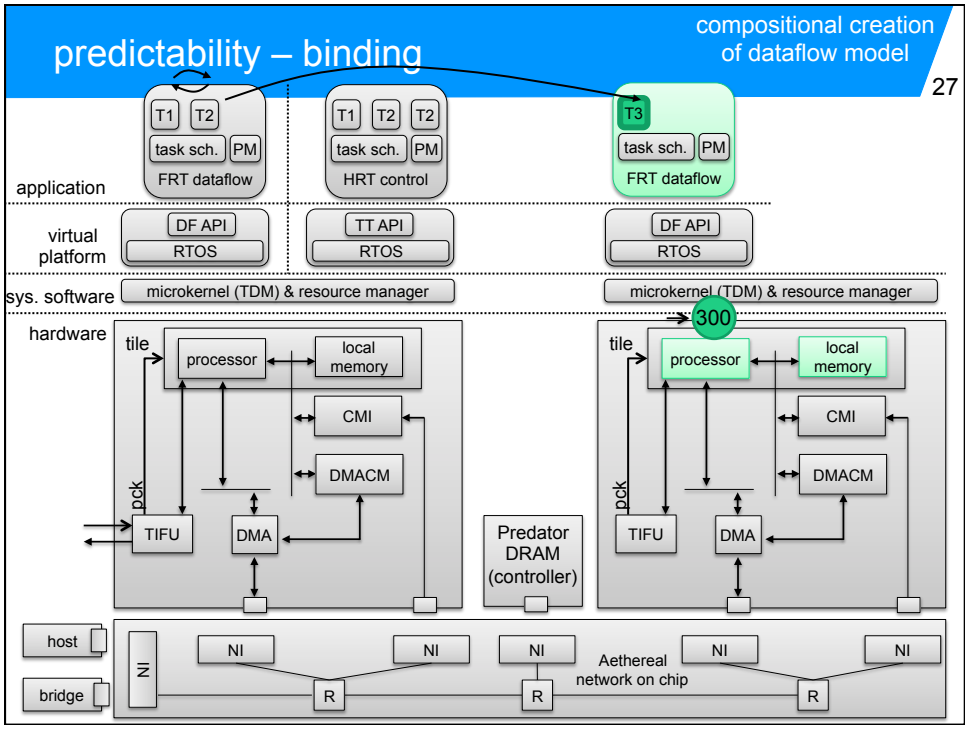


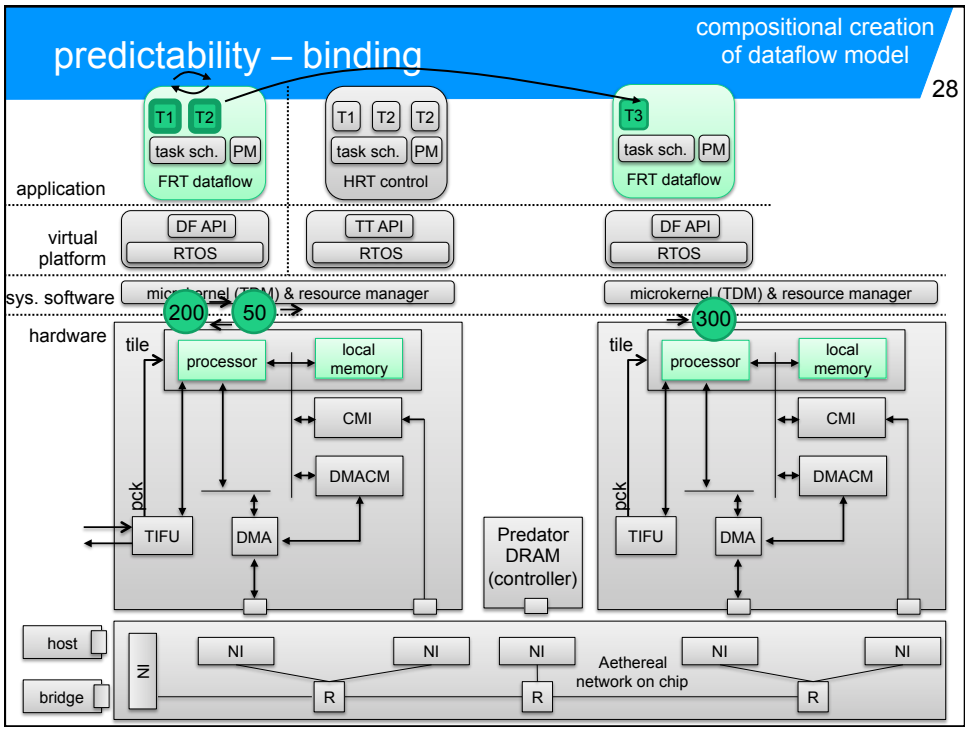
predictability

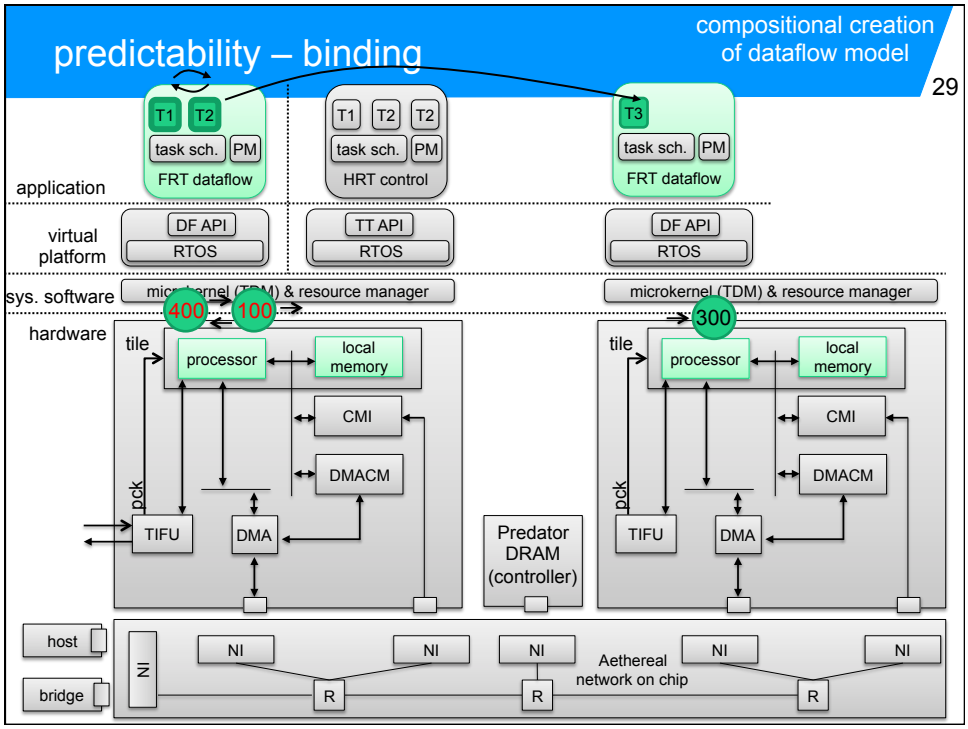
25

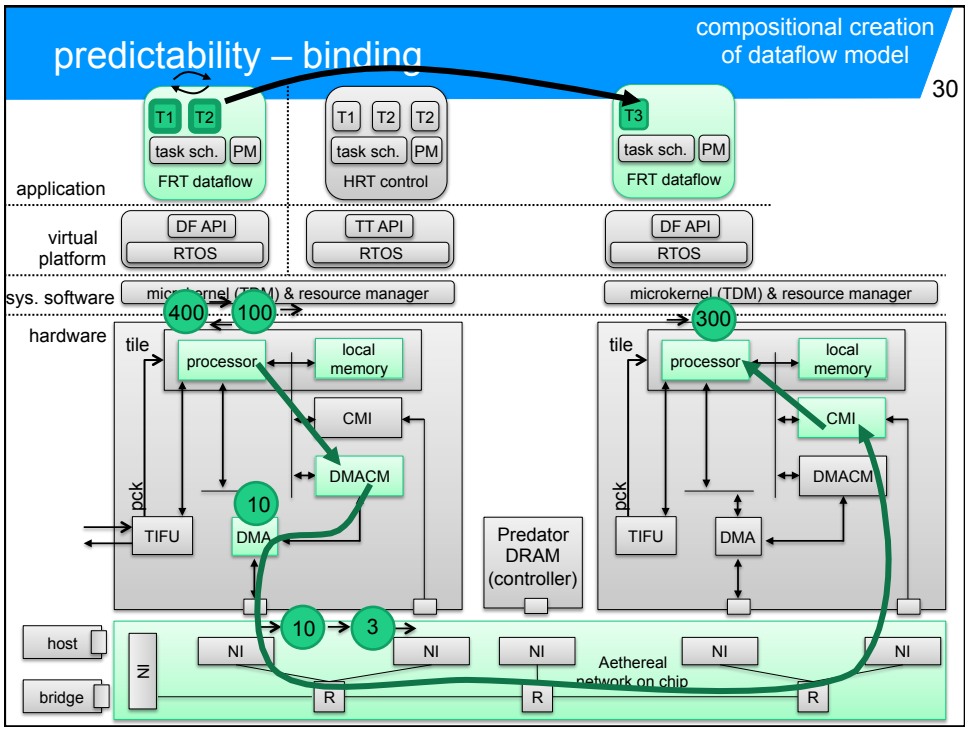
- cyclo-static dataflow (CSDF) is
  - the programming model
  - used to describe the binding of application to platform
    - unshared resources
    - shared resources
- extending to scenario/mode-aware dataflow (Stuijk, Moreira)


predictability
compositional creation of dataflow model
26
application
virtual platform
sys. software
hardware
T1
T2
task sch.
PM
FRT dataflow
HRT control
T3
DF API
TT API
RTOS
microkernel (TDM) & resource manager
tile
processor
local memory
CMI
DMACM
pck
TIFU
DMA
Predator DRAM (controller)
host
bridge
NI
R
Aethereal network on chip


predictability – binding
compositional creation of dataflow model
27
application
virtual platform
sys. software
hardware
T1
T2
task sch.
PM
FRT dataflow
HRT control
T3
DF API
TT API
RTOS
microkernel (TDM) & resource manager
300
tile
processor
local memory
CMI
DMACM
pck
TIFU
DMA
Predator DRAM (controller)
host
bridge
NI
R
Aethereal network on chip

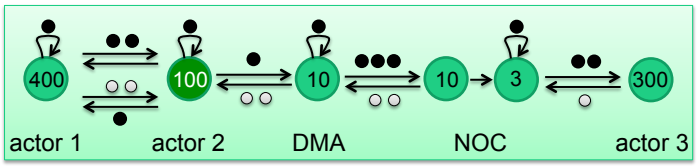

predictability – binding
compositional creation of dataflow model
28
application
virtual platform
sys. software
hardware
T1
T2
task sch.
PM
FRT dataflow
HRT control
T3
DF API
TT API
RTOS
microkernel (TDM) & resource manager
200
50
300
tile
processor
local memory
CMI
DMACM
pck
TIFU
DMA
Predator DRAM (controller)
host
bridge
NI
R
Aethereal network on chip


predictability – binding
compositional creation of dataflow model
29
application
virtual platform
sys. software
hardware
T1
T2
task sch.
PM
FRT dataflow
HRT control
T3
DF API
TT API
RTOS
microkernel (TDM) & resource manager
400
100
300
tile
processor
local memory
CMI
DMACM
pck
TIFU
DMA
Predator DRAM (controller)
host
bridge
NI
R
Aethereal network on chip



predictability with CSDF

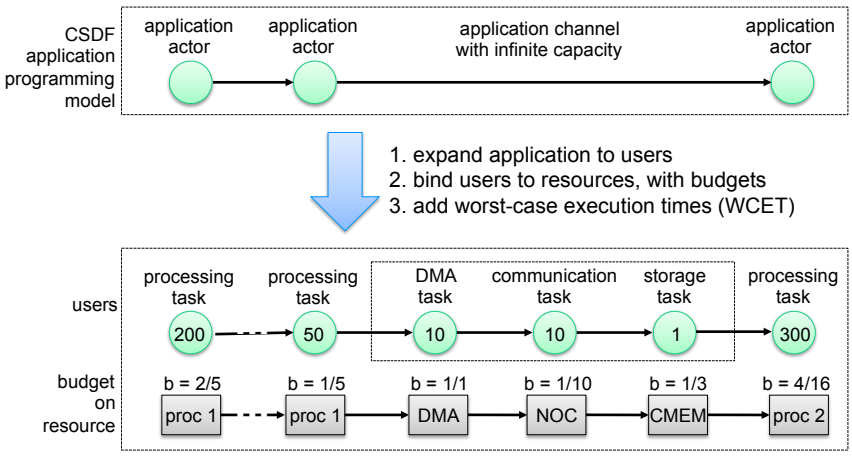
31



- CSDF graph captures
  - application functionality & data dependencies
  - mapping of application on unshared resources (WCET)
  - resource sharing (SO, TDM, RR, LR, ...)
  - finite buffer sizes, flow control / backpressure
- compositional method
  - CSDF graph is constructed of independent smaller subgraphs
  - budgets within application, and between applications

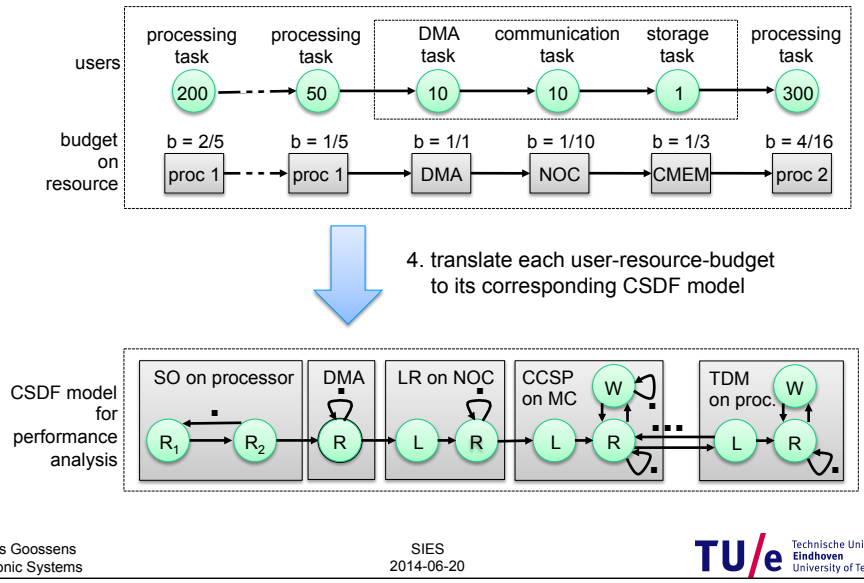
predictability with CSDF

32



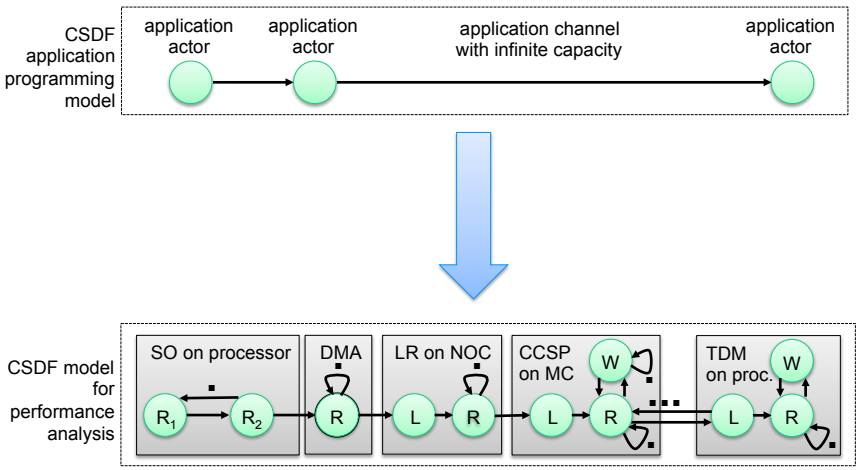
predictability with CSDF

33



predictability with CSDF

34



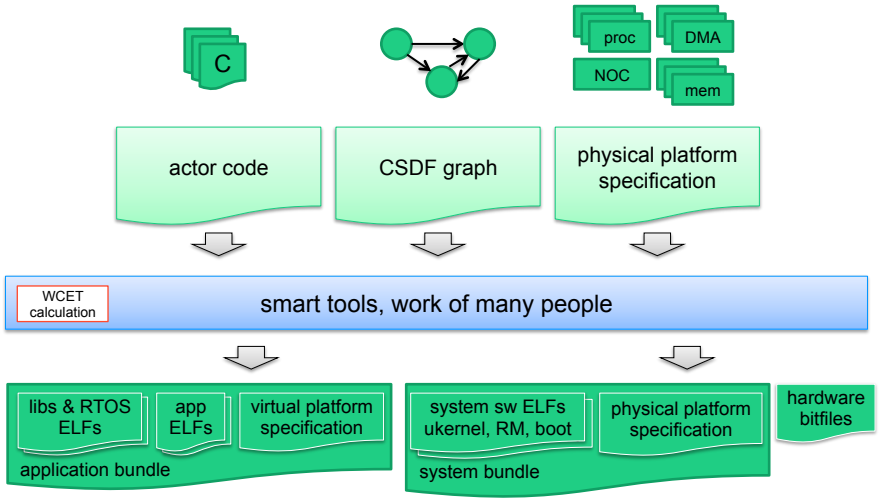
design flow

35

- the CompSOC & SDF3 dataflow framework
- automatic generation of
  - hardware: processor tiles, NOC, memory controllers
  - software: RTOS, resource management, drivers, MOC libraries
- for applications
  - application bundles
    - ELFs
    - virtual platform specification (actor-resource binding, buffer sizes, RTOS scheduling interval, scheduler settings (SO schedule, TDM slots, CCSP priorities, etc.))
    - dynamically loaded on the hardware at run time
- for CSDF applications
  - end-to-end application throughput and latency analysis
- FPGA prototype

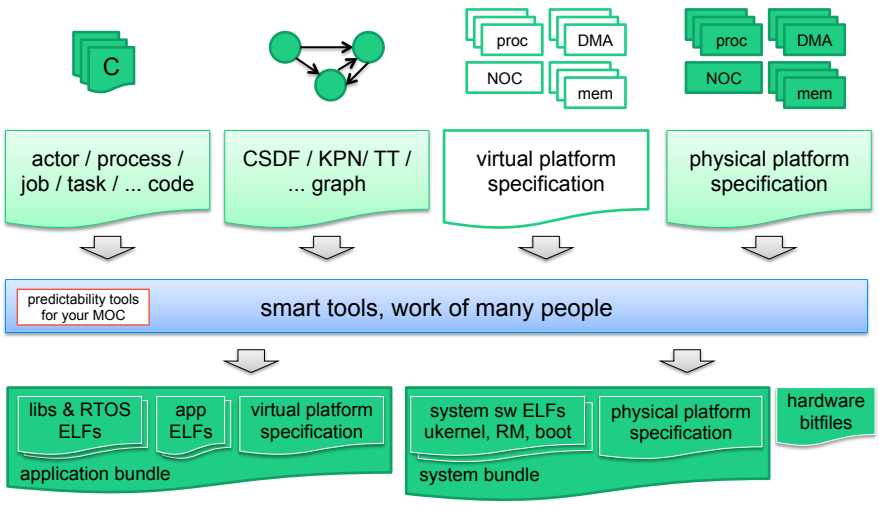
CompSOC & SDF³ design flow

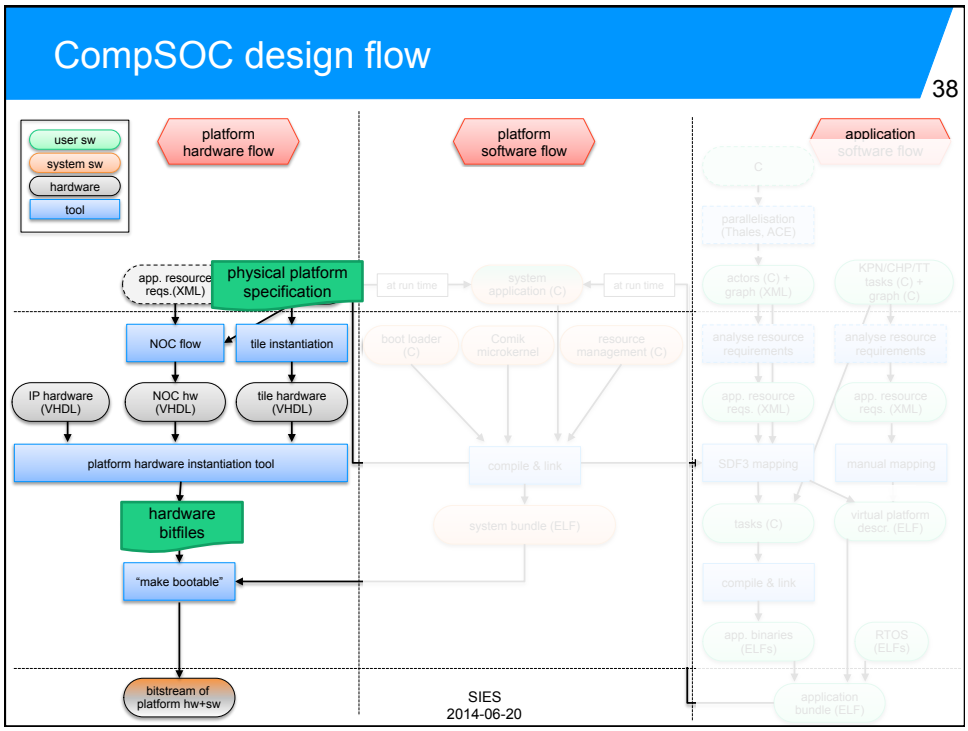
36

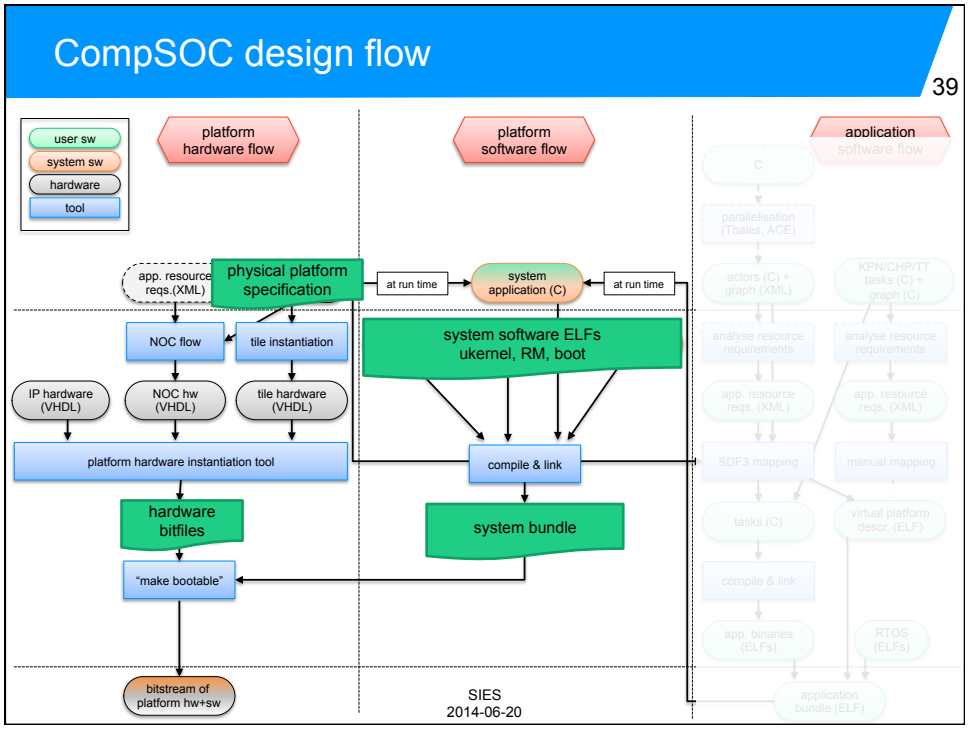


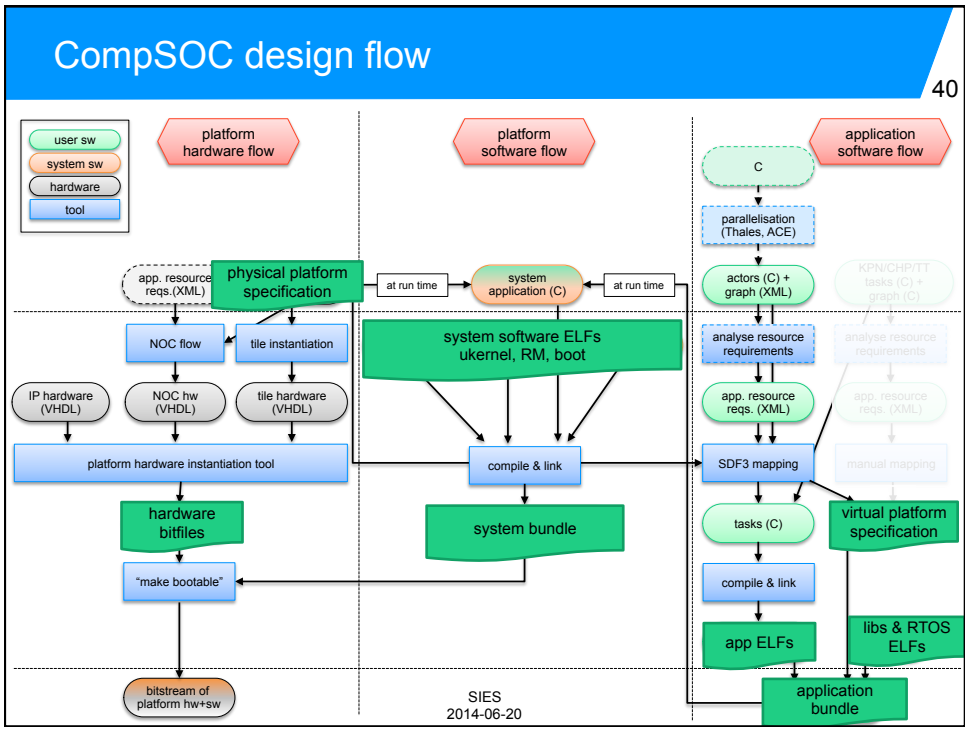
CompSOC design flow

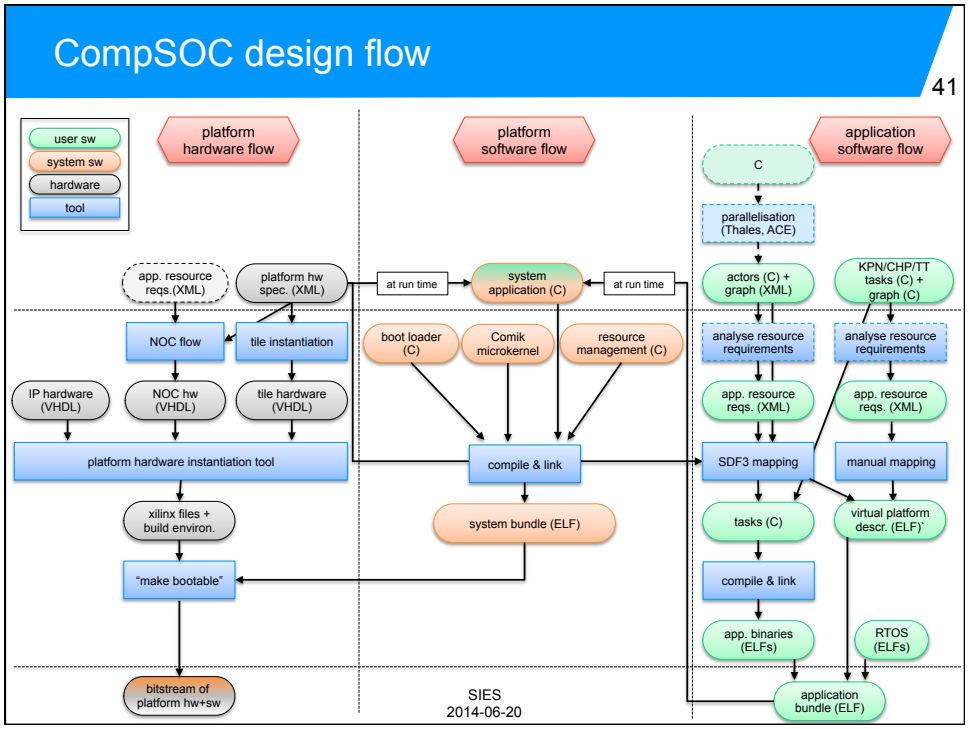
37




CompSOC design flow
38
user sw
system sw
hardware
tool
platform hardware flow
platform software flow
application software flow
app. resource reqs.(XML)
physical platform specification
at run time
system application (C)
at run time
NOC flow
tile instantiation
boot loader (C)
Comik microkernel
resource management (C)
IP hardware (VHDL)
NOC hw (VHDL)
tile hardware (VHDL)
platform hardware instantiation tool
compile & link
hardware bitfiles
system bundle (ELF)
"make bootable"
bitstream of platform hw+sw
SIES
2014-06-20
C
parallelisation (Thales, ACE)
actors (C) + graph (XML)
KPN/CHP/TT tasks (C) + graph (C)
analyse resource requirements
app. resource reqs. (XML)
SDF3 mapping
manual mapping
tasks (C)
virtual platform descr. (ELF)
compile & link
app. binaries (ELFs)
RTOS (ELFs)
application bundle (ELF)

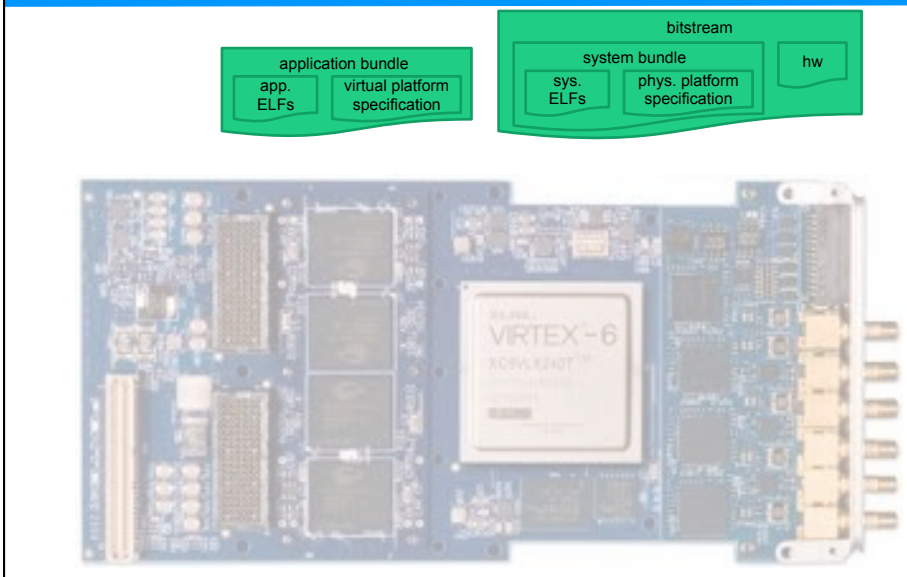

CompSOC design flow
39
user sw
system sw
hardware
tool
platform hardware flow
platform software flow
application software flow
app. resource reqs.(XML)
physical platform specification
at run time
system application (C)
at run time
NOC flow
tile instantiation
system software ELFs ukernel, RM, boot
IP hardware (VHDL)
NOC hw (VHDL)
tile hardware (VHDL)
platform hardware instantiation tool
compile & link
hardware bitfiles
system bundle
"make bootable"
bitstream of platform hw+sw
SIES
2014-06-20
C
parallelisation (Thales, ACE)
actors (C) + graph (XML)
KPN/CHP/TT tasks (C) + graph (C)
analyse resource requirements
app. resource reqs. (XML)
SDF3 mapping
manual mapping
tasks (C)
virtual platform descr. (ELF)
compile & link
app. binaries (ELFs)
RTOS (ELFs)
application bundle (ELF)


CompSOC design flow
40
user sw
system sw
hardware
tool
platform hardware flow
platform software flow
application software flow
C
parallelisation (Thales, ACE)
app. resource reqs.(XML)
physical platform specification
at run time
system application (C)
at run time
actors (C) + graph (XML)
KPN/CHP/TT tasks (C) + graph (C)
NOC flow
tile instantiation
system software ELFs ukernel, RM, boot
analyse resource requirements
analyse resource requirements
IP hardware (VHDL)
NOC hw (VHDL)
tile hardware (VHDL)
app. resource reqs. (XML)
app. resource reqs. (XML)
platform hardware instantiation tool
compile & link
SDF3 mapping
manual mapping
hardware bitfiles
system bundle
tasks (C)
virtual platform specification
"make bootable"
compile & link
app ELFs
libs & RTOS ELFs
bitstream of platform hw+sw
SIES
2014-06-20
application bundle


CompSOC design flow
41
user sw
system sw
hardware
tool
platform hardware flow
platform software flow
application software flow
C
parallelisation (Thales, ACE)
app. resource reqs.(XML)
platform hw spec. (XML)
at run time
system application (C)
at run time
actors (C) + graph (XML)
KPN/CHP/TT tasks (C) + graph (C)
NOC flow
tile instantiation
boot loader (C)
Comik microkernel
resource management (C)
analyse resource requirements
analyse resource requirements
IP hardware (VHDL)
NOC hw (VHDL)
tile hardware (VHDL)
app. resource reqs. (XML)
app. resource reqs. (XML)
platform hardware instantiation tool
compile & link
SDF3 mapping
manual mapping
xilinx files + build environ.
system bundle (ELF)
tasks (C)
virtual platform descr. (ELF)'
"make bootable"
compile & link
app. binaries (ELFs)
RTOS (ELFs)
bitstream of platform hw+sw
SIES
2014-06-20
application bundle (ELF)

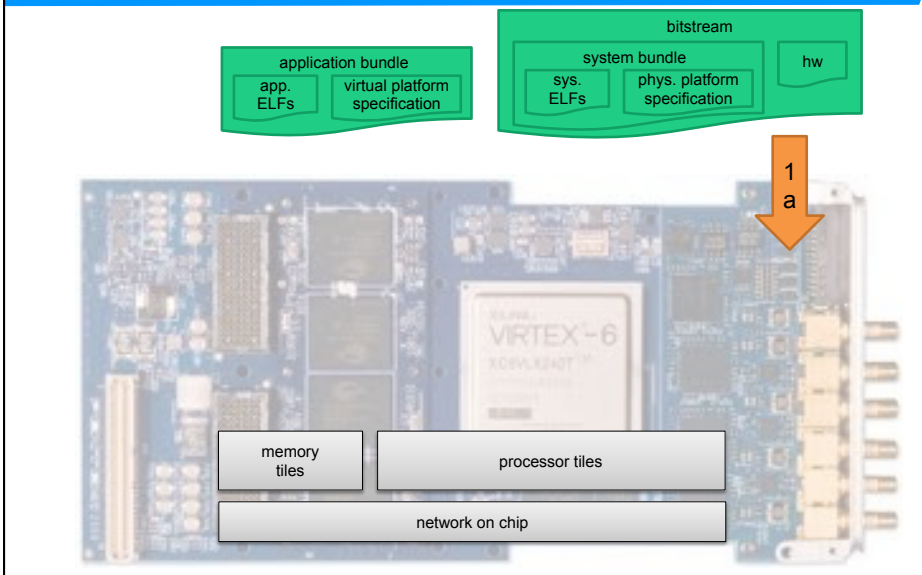
CompSOC – run-time loading

42



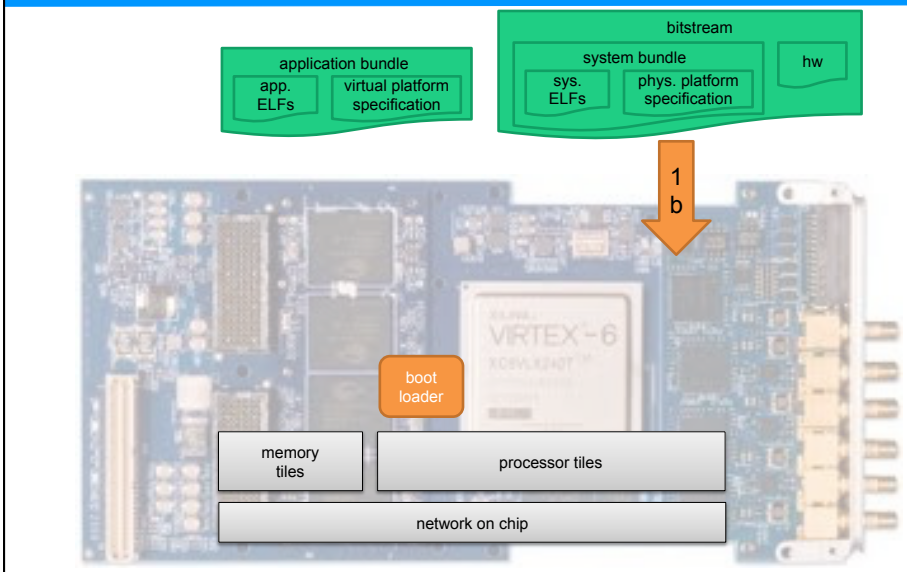
CompSOC – run-time loading

43



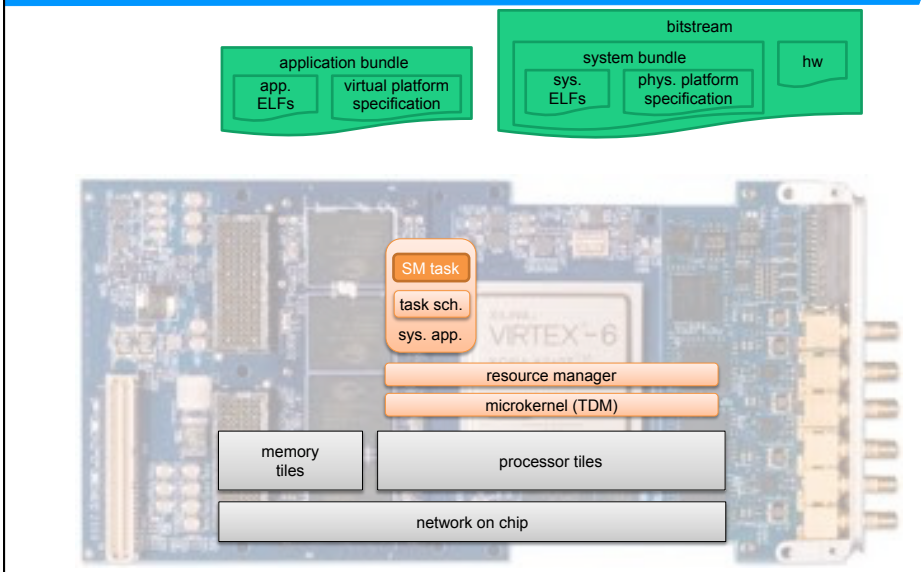
CompSOC – run-time loading

44



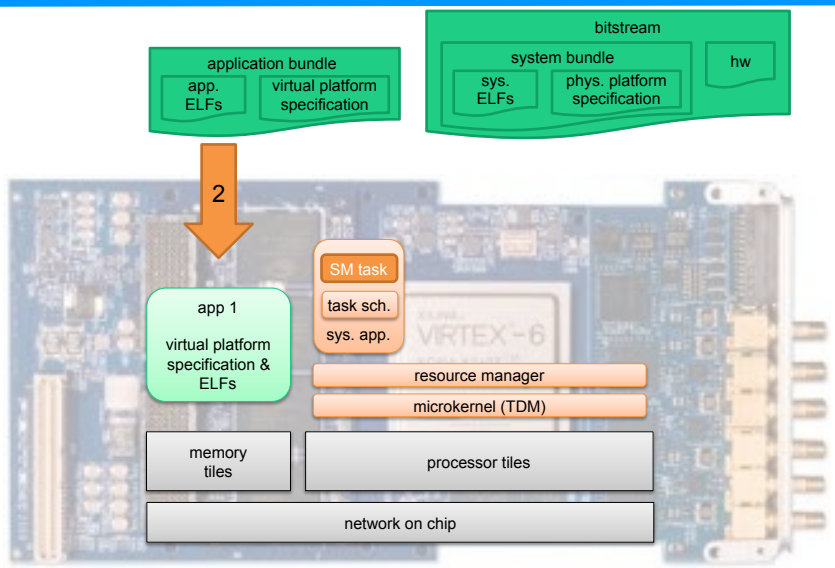
CompSOC – run-time loading

45



CompSOC – run-time loading

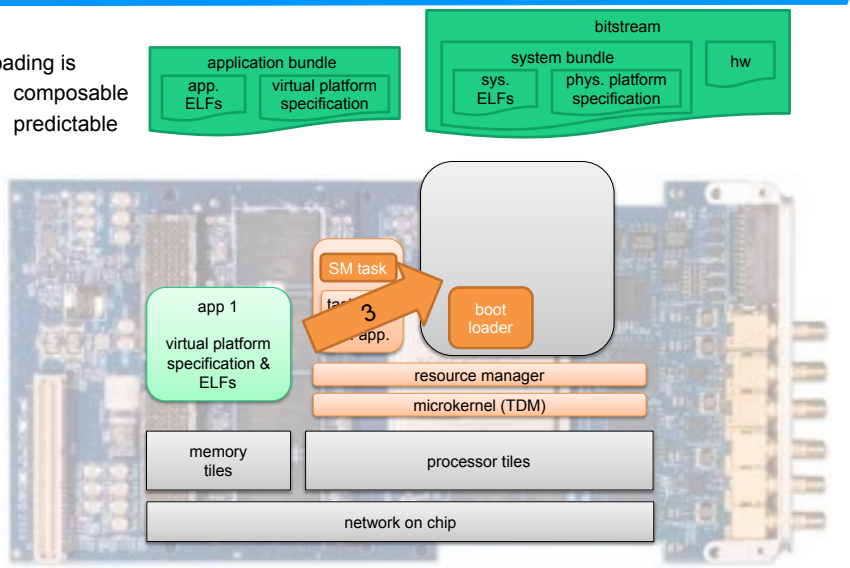
46



CompSOC – run-time loading

47

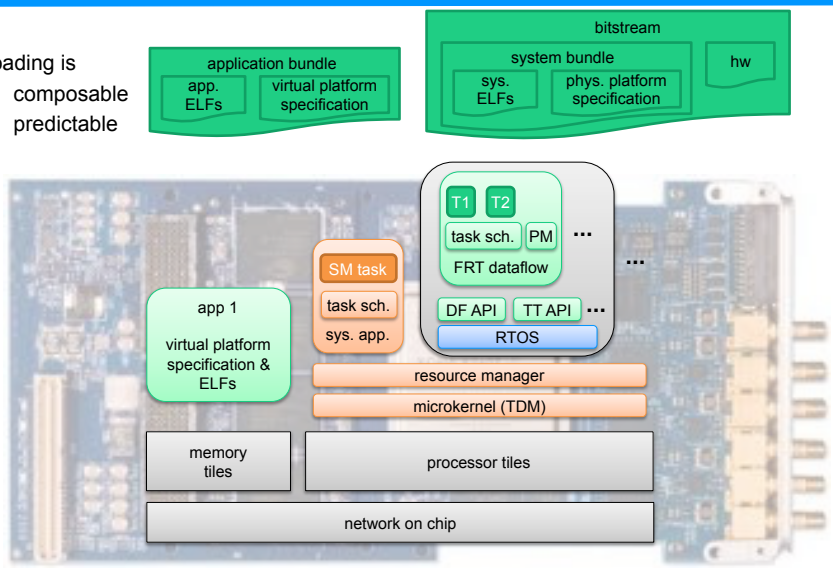
- loading is
  - composable
  - predictable



CompSOC – run-time loading

48

- loading is
  - composable
  - predictable



(current) restrictions

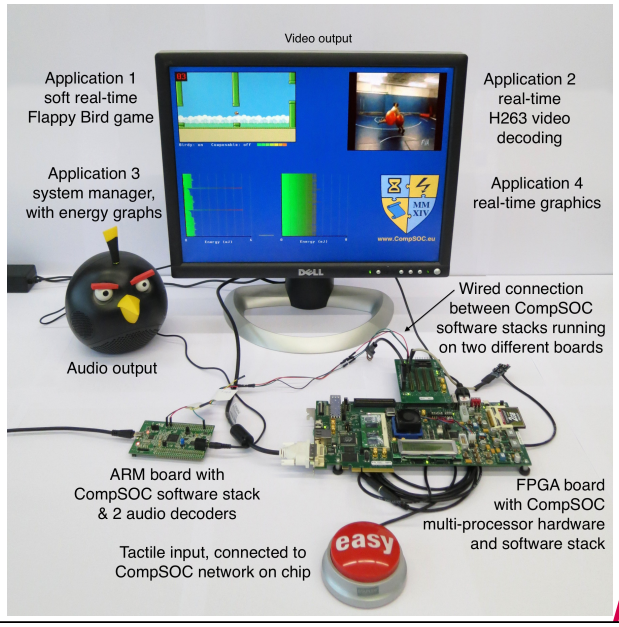
49

- data and code must fit in local tile memories
  - no caches, or else flush on preemption
- no I/O virtualisation
- DVFS is simulated on FPGA
- external interrupts – supported
- preemptive intra-application scheduling – supported
- memory protection – supported

- limited support for multiple use cases in the design flow
- currently supported programming models
  - cyclo-static dataflow (actors)
  - Kahn process networks (processes)
  - time-triggered (jobs) – limited

demo: playback-dynamic-load.mp4 [DATE'14]

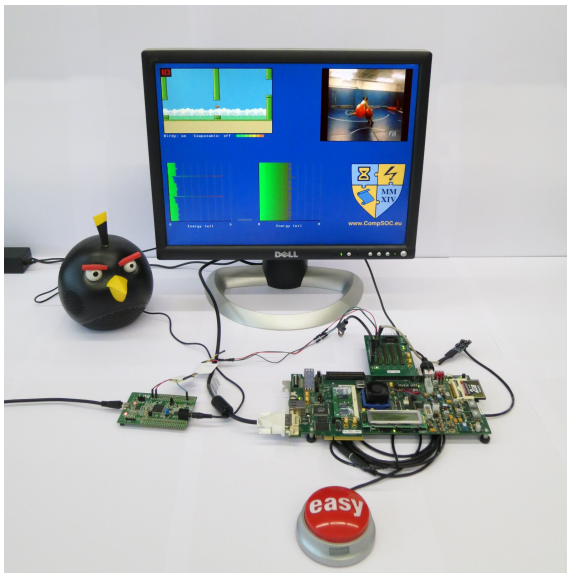
50



demo: playback-dynamic-load.mp4 [DATE'14]

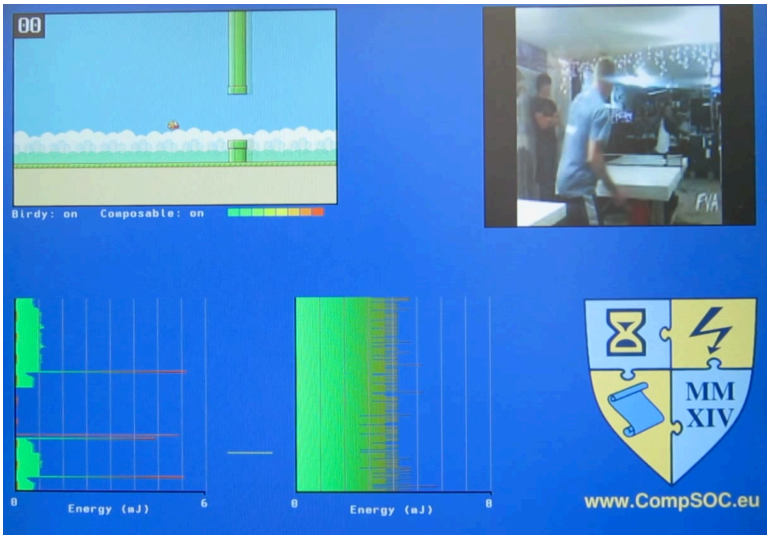
51

- applications
  - SRT flappy bird
  - RT adaptive H263 decoder
  - RT graphics
  - system application
  - 2x audio decoding
- FPGA board
  - MPSOC
  - for button input
  - for video output
- ARM board
  - for audio mixing & output
- linked with serial I/O



demo: playback-dynamic-load.mp4 [DATE'14]

52



conclusions

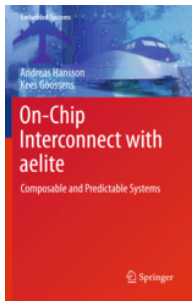
53

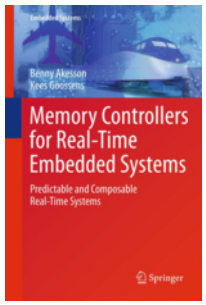
- reduce SOC design effort
- independent design, verification, and execution per application
- composability
- predictability
- virtual execution platforms with
  application-specific MOC, scheduling, power management
  – any mix of NRT, SRT, FRT
- design flow
- FPGA prototype
- used in teaching MSc embedded systems lab

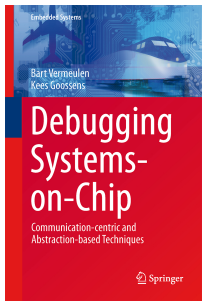
more information: www.compsoc.eu

54

- CompSOC overview in SIGBED'13 and in Multiprocessor System-on-Chip. Huebner (ed), Springer, 2010
- Aethereal real-time NOC, DAC'10
- Real-time DRAM memory controller, DATE'13, ECRTS'14
- CompOSe RTOS, MICPRO'11
- CoMiK microkernel, DATE'14
- Composable power management, SAMOS'11
- Design flow, FPGA World'13
- SDF3, DAC'06 http://www.es.ele.tue.nl/sdf3/







acknowledgements

55

- Eindhoven university of technology
  - Kees Goossens (team leader)
  - Gabriela Breaban
  - Sven Goossens
  - Martijn Koedam
  - Reinier van Kampenhout
  - Yonghui Li
  - Andrew Nelson
  - Shubhendu Sinha
  - Rasool Tavakoli
  - Juan Valencia
- Prague University
  - Benny Akesson
- close collaboration with the dataflow research (SDF3) at TU/e
  - Sander Stuijk
  - Marc Geilen
- and many others

This research is supported by
EU grants CATRENE CA505 **BENEFIC,**
CA703 **OpenES**, CT217 **RESIST**,
ARTEMIS-2013-1 621429 **EMC2** and 621353 **DEWI.**
Parts of the platform were developed with support from Flextiles, T-CREST, COMCAS, Cobra, Scalopes, TSAR, NEST, NEVA, MESA.

end

56

for further information

www.compsoc.eu

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