AARHUS
UNIVERSITY
DEPARTMENT OF ENGINEERING

29. AUGUST 2013

MODELLING DIFFERENT CPU POWER STATES IN VDM-RT

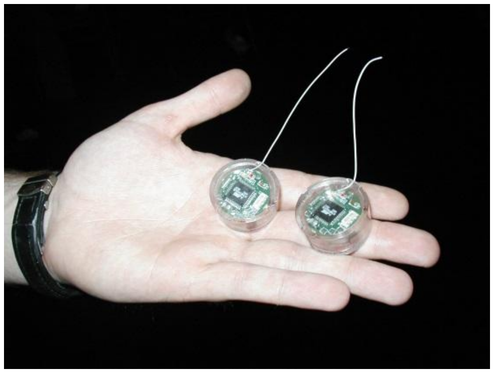
JOSÉ ANTONIO ESPARZA ISASA AND PETER GORM LARSEN

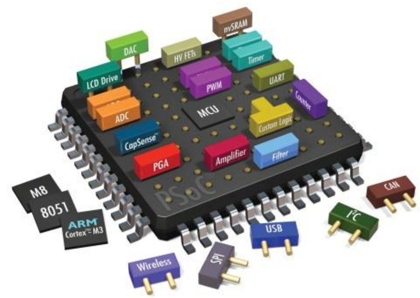
AGENDA

1. Introduction
2. Power modes in commercial microcontrollers
3. CPUs in VDM-RT
4. Modelling power modes for VDM-RT CPUs
5. Scenario 1: functionallity that makes the CPU active
6. Scenario 2: functionallity that runs if the CPU is active
7. Calculating energy consumption
8. Suggested additions to the Overture platform
9. Conclusions

AARHUS UNIVERSITY
DEPARTMENT OF ENGINEERING

INTRODUCTION

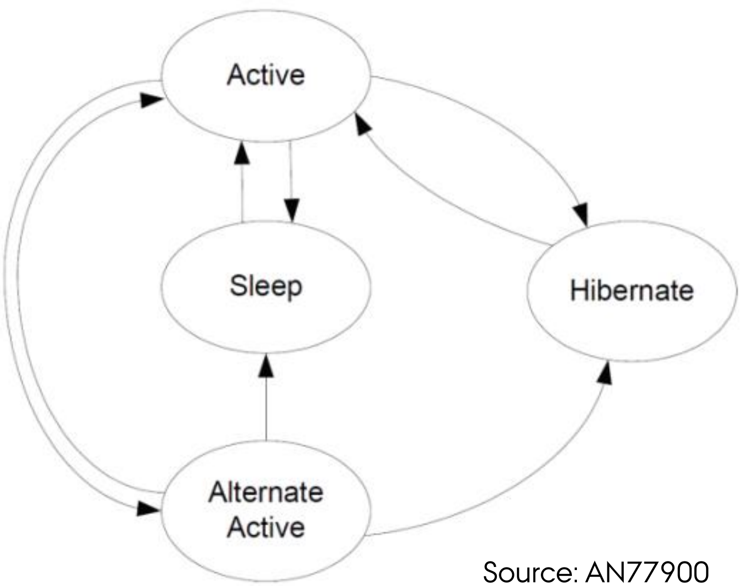




- Implement & Optimize vs. System Level Design
- Power consumption under a model-driven engineering design approach?
- How can we model several CPU power states in VDM-RT?

POWER MODES IN COMMERCIAL MICROCONTROLLERS

- Dynamic Frequency Scaling
- Dynamic Voltage Scaling
- Predefined consumption modes: consuption figures readily available



Source: AN77900
Cypress

- Clock speed vs. Consumption
- Duty cycle of the application?
- Kind of application?

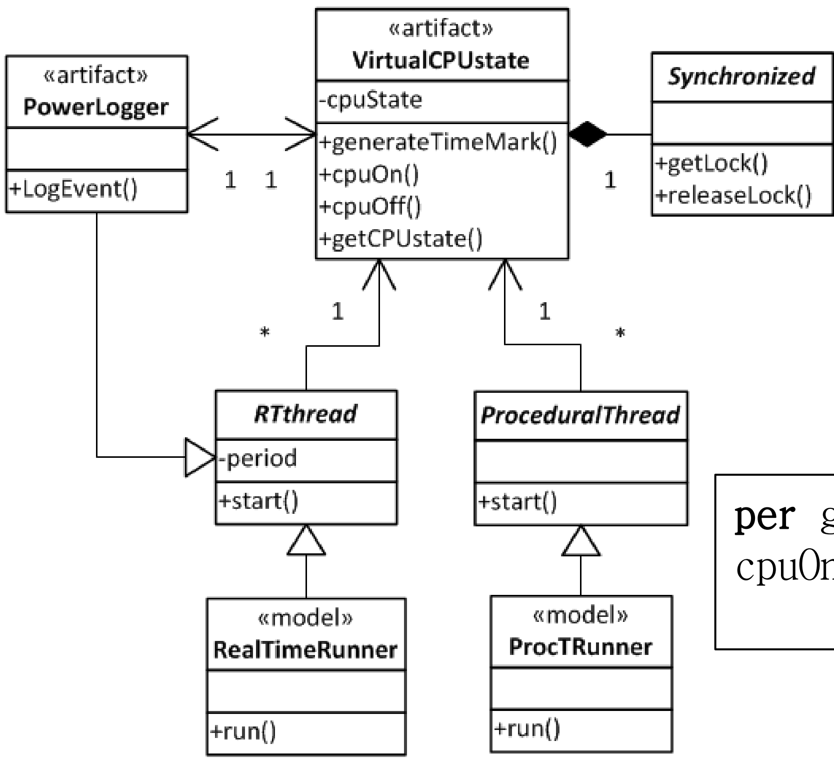
CPUS IN VDM-RT

- CPUs are execution nodes
- Different parts of a model can be deployed

- Constant frequency
- Minimal scheduling policies included

```
mcu : CPU := new CPU(<FP>, 20e6); -- 20 MHz controller
mcu.deploy(model);
```

MODELLING POWER MODES FOR VDM-RT CPUS



- Thread safe access to VirtualCPUstate
- Polling to VirtualCPUstate surrounded by `duration(0)`

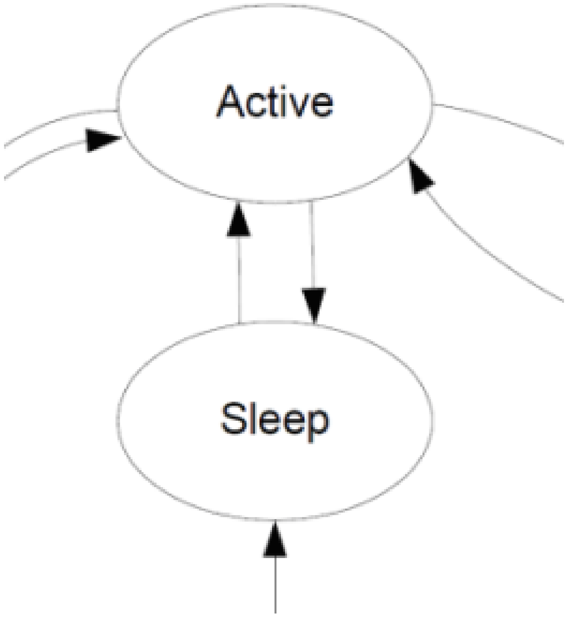
```
per getCPUstate => #active(getCPUstate) = 0  and
cpuOn = true => #fin(turnOff) = #fin(getCPUstate);
```

AARHUS UNIVERSITY
DEPARTMENT OF ENGINEERING

SCENARIO 1: FUNCTIONALLITY THAT MAKES THE CPU ACTIVE

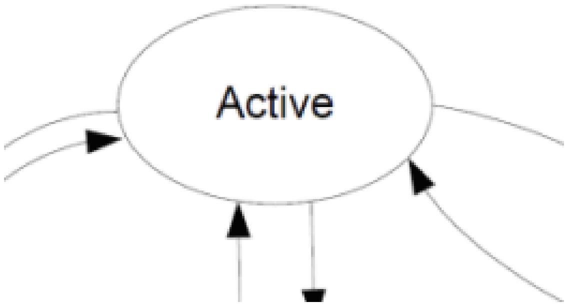
```
duration (0)   state.turnOn();
duration (200) (executeLogic());
duration (0)   state.turnOff();
```

```
public turnOn : () ==> ()
turnOn() ==
(
        cpuOn := true;
        logger.logOn();
);
```



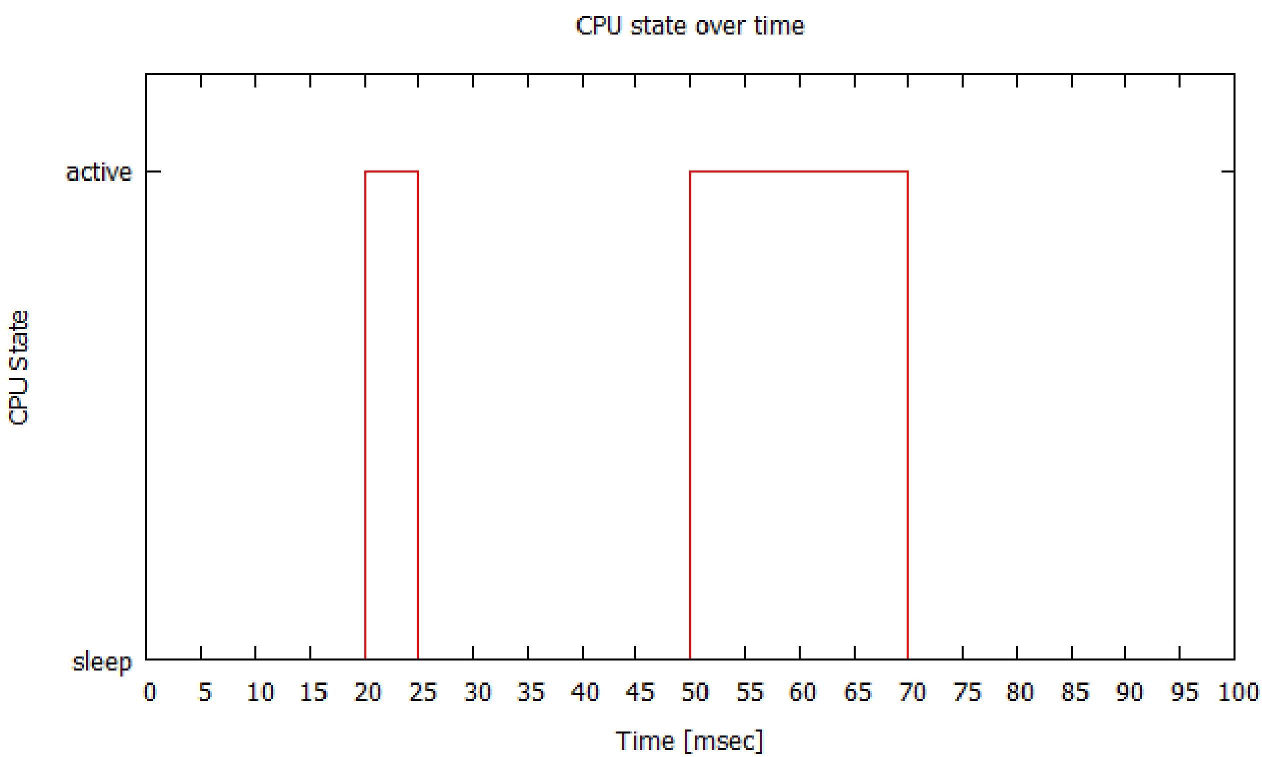

```
public logOn: () ==> ()
logOn() == stateChanges := stateChanges ^ [time];
```

SCENARIO 2: FUNCTIONALLITY THAT RUNS IF THE CPU IS ACTIVE



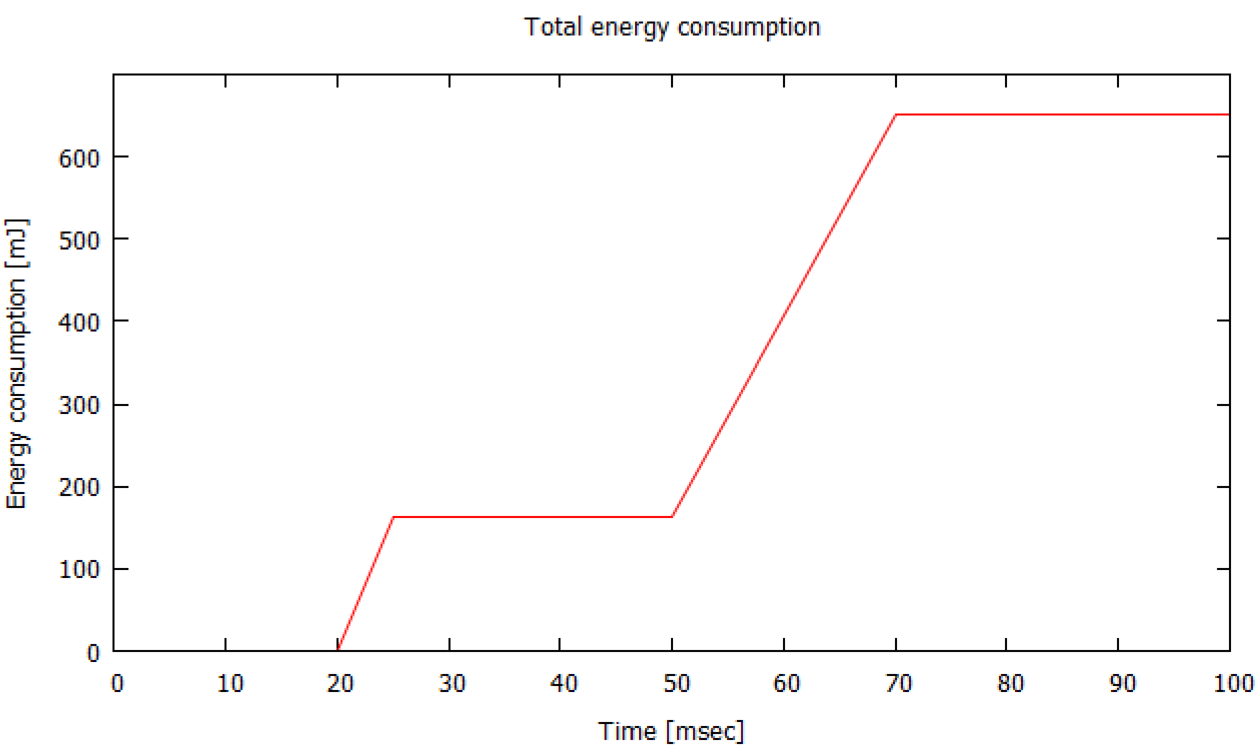

```
duration (0) if not state.getCPUstate() then
(
        IO`print("₩nCPU is off");
)
else (
        duration (200) (IO`print( “Exec Logic"));
        duration (0) state.turnOff();
);
```

CALCULATING ENERGY CONSUMPTION (I)



- @ 24MHz
- Active 6.5mA
- Op. 5 V

CALCULATING ENERGY CONSUMPTION (II)



- Total energy consumption of 650 mJ

SUGGESTED ADDITIONS TO THE OVERTURE PLATFORM (I)

› Incorporating events to VDM-RT
› Periodic internal CPU events
› Periodic/Aperiodic external to CPU events

```
duration (0)if time = eventTime
              then eventGenerator.feed(cpu,event);
```

› Using events to wake up CPU

```
cpu.wakeOn(event);
```

› Sleeping the CPU

```
cpu.threadsActive();
cpu.sleep();
```

SUGGESTED ADDITIONS TO THE OVERTURE PLATFORM

- Implementation in the VDM-RT java engine
- Automatic power consumption graph generation
- Dynamic adjustment of operating frequency
- Reconsider the bus constructor

CONCLUSIONS

- Initial approach to multi-state CPU modelling
- No tool support at the moment
- Time synchronization between model and implementation is critical in order to get accurate estimations.