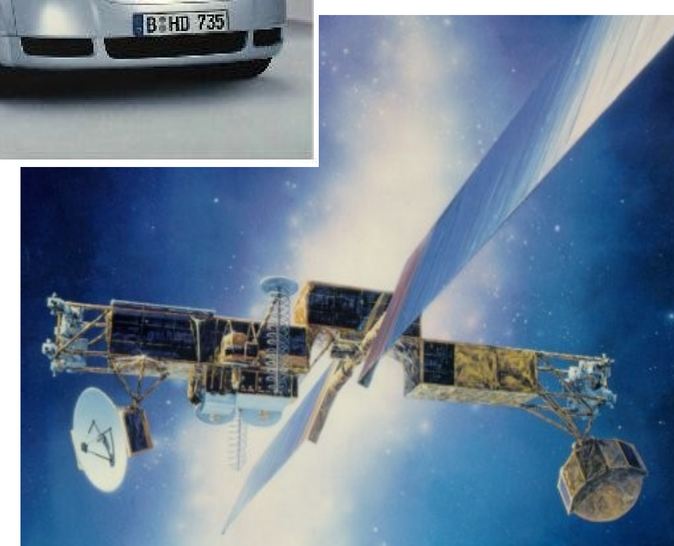
Scheduling and Voltage Scaling for Energy/Reliability Trade-offs in Fault-Tolerant Time-Triggered Embedded Systems

Kåre Harbo Poulsen,

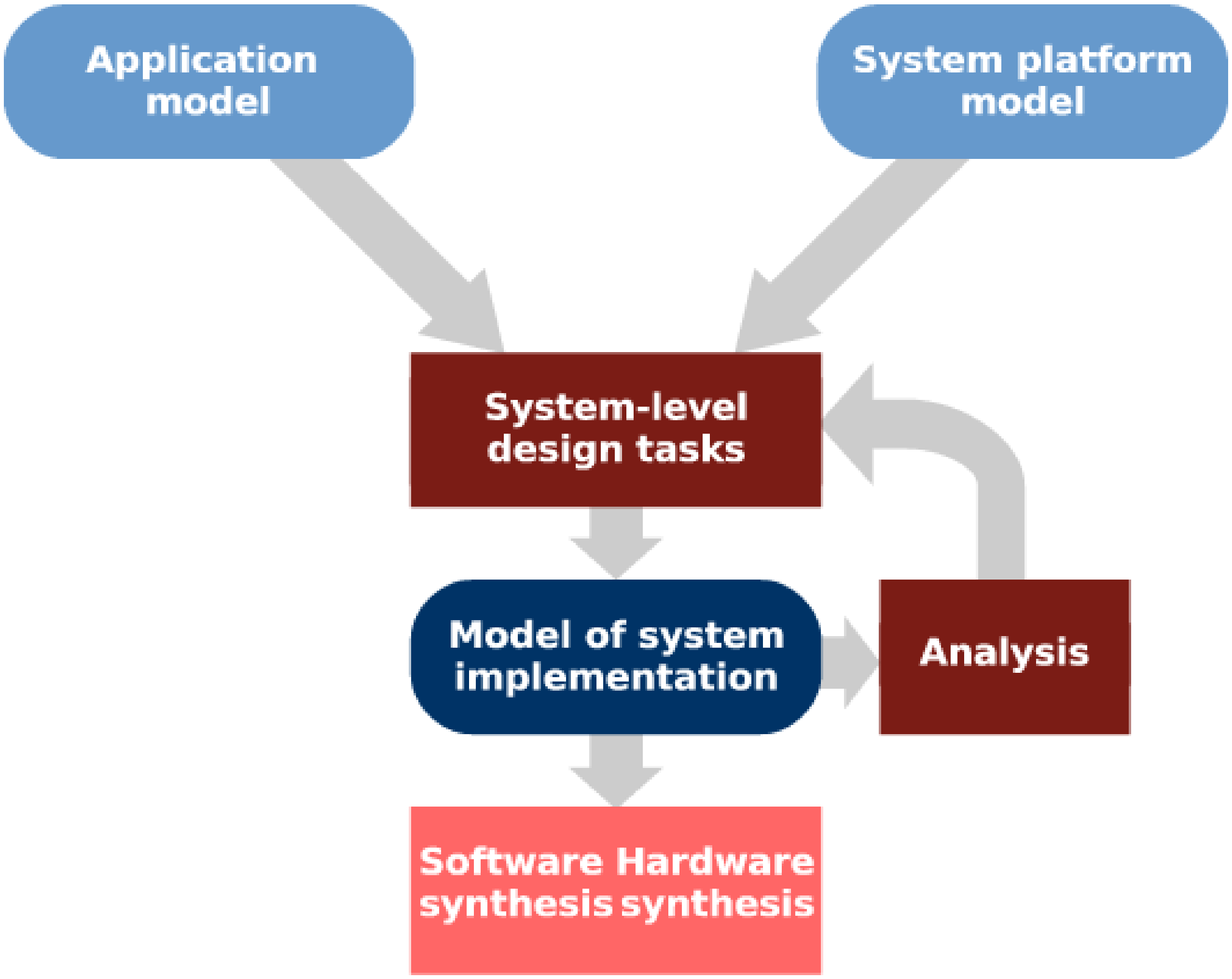
Paul Pop, Viacheslav Izosimov

August 23, 2007

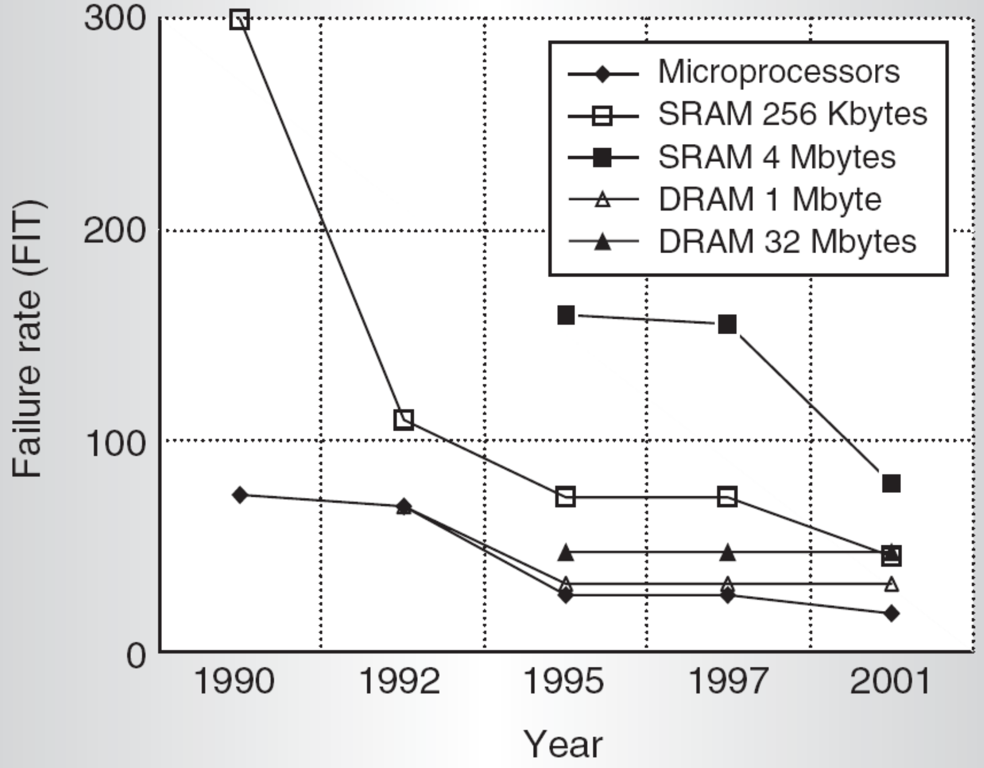
Embedded Systems

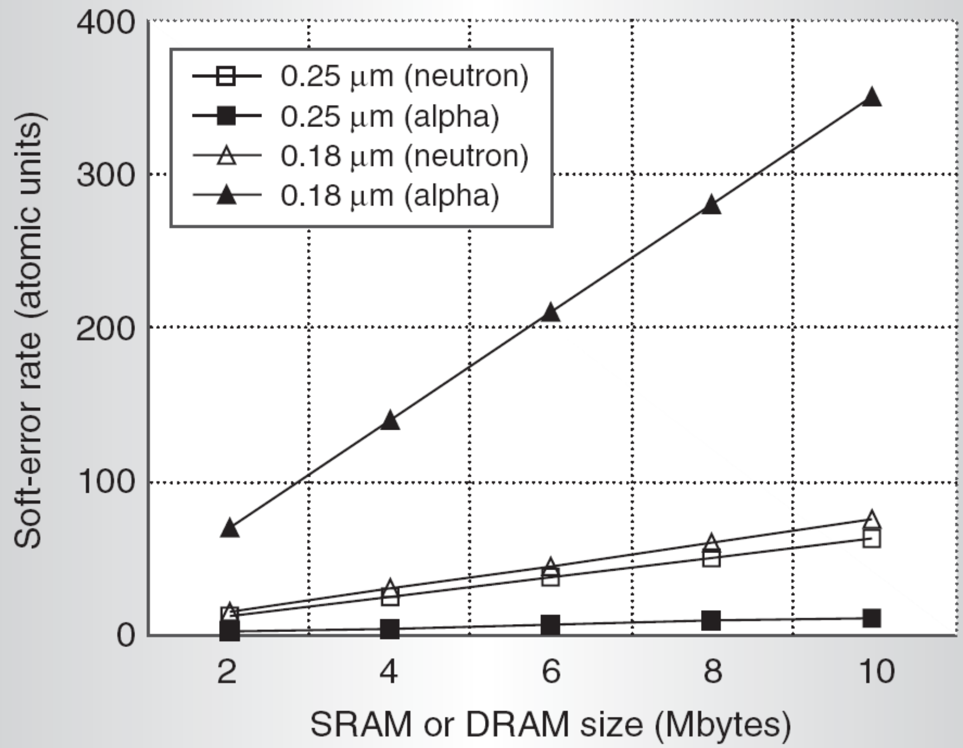


Design of Embedded Systems



Faults



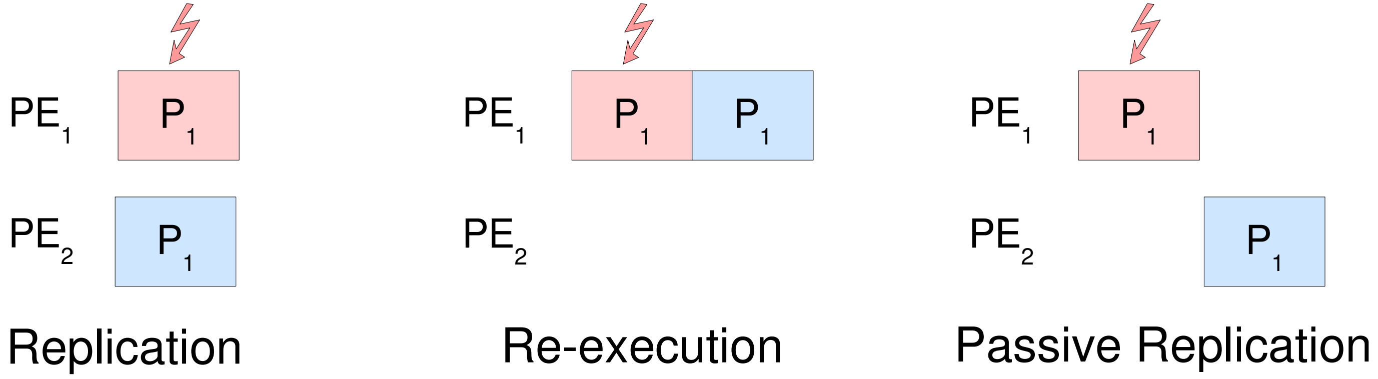


- Permanent faults are decreasing
- Transient faults are increasing

From: Cristian Continescu, Trends and challenges in VLSI circuit reliability, 2003

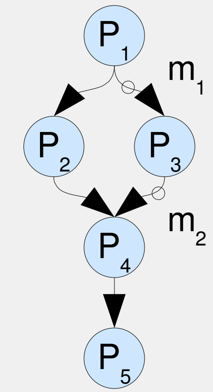
Fault-Tolerance

- Tolerate faults gracefully
- Expressions for reliability for fault-tolerance

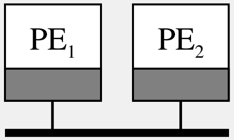


Embedded Systems Model

- Input
  - Application
  - Architecture
  - Reliability goal: 0.999 999 999

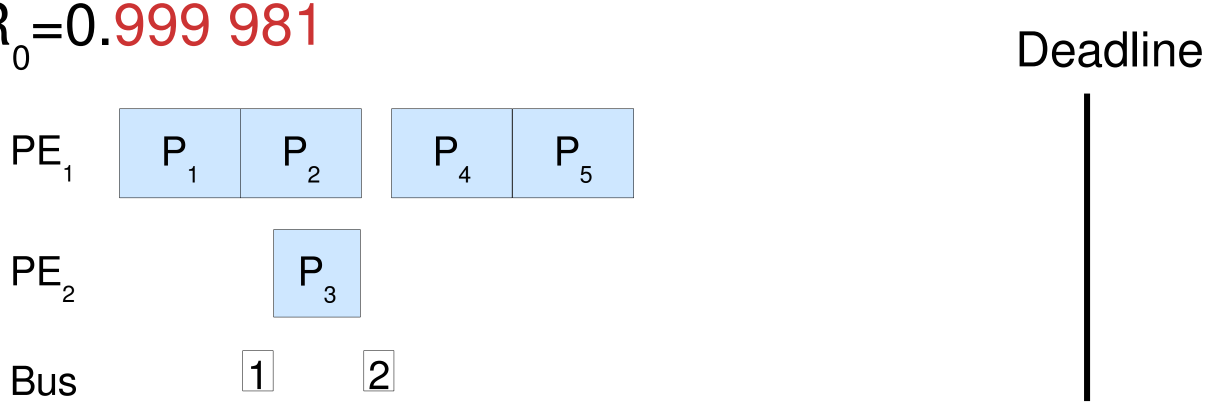


	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4



$k=1$

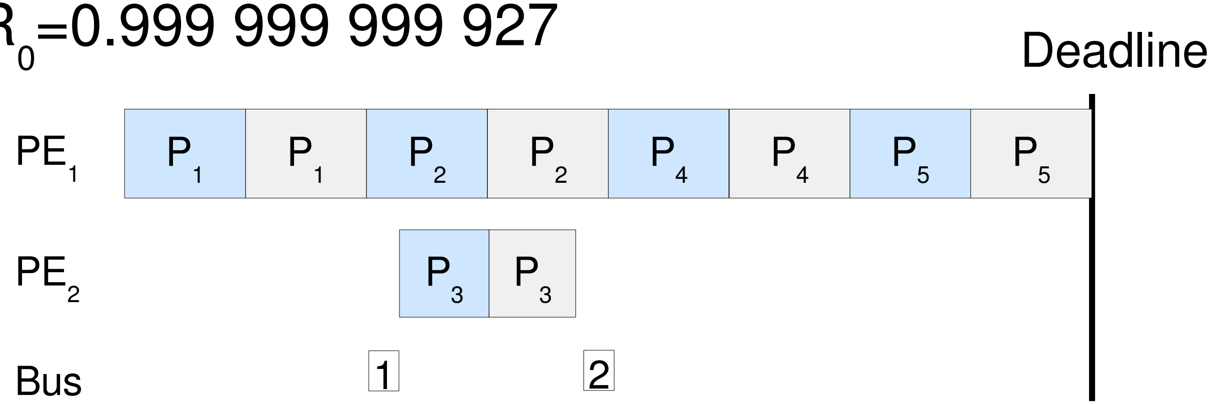
$R_0=0.999\ 981$

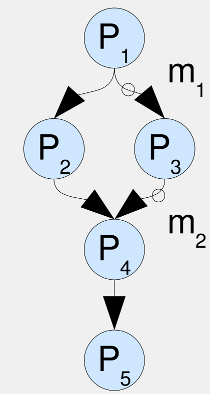


Fault-Tolerant Scheduling

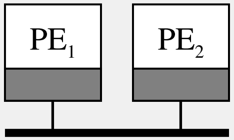
- Input
  - Application
  - Architecture
  - Reliability goal: 0.999 999 999
- Fault-tolerance for *k=1* faults

R_0=0.999 999 999 927





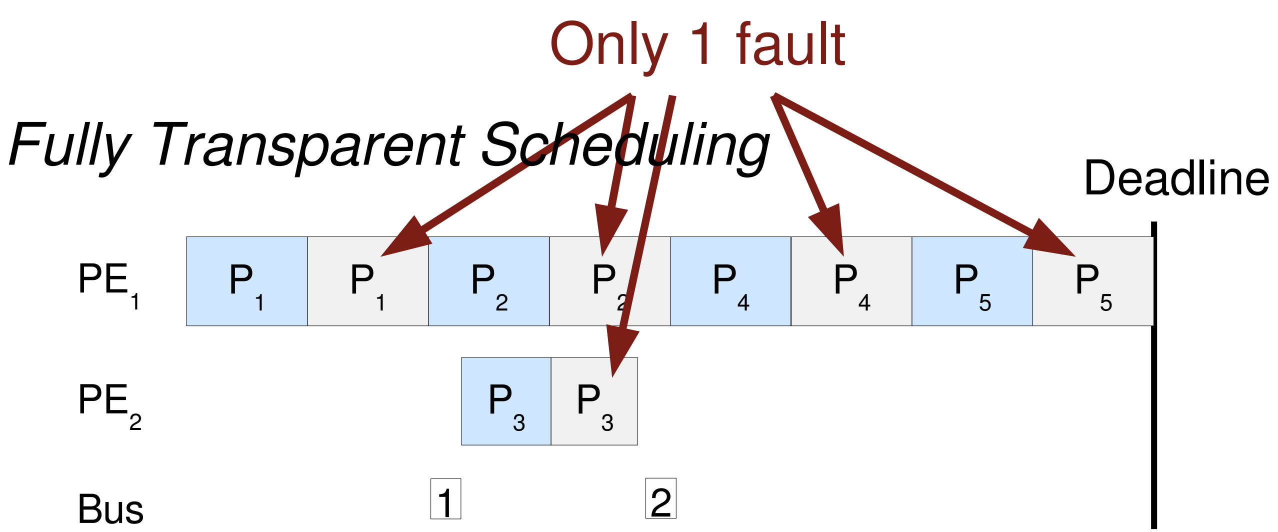
	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4

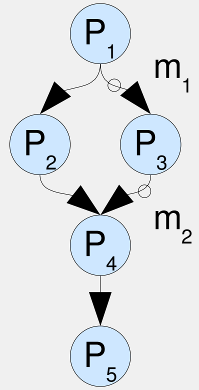


k=1

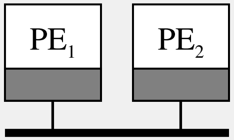
Fault-Tolerant Scheduling

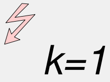
- Fault tolerant scheduler
  - Full transparency
    - Good debugability
    - Little memory





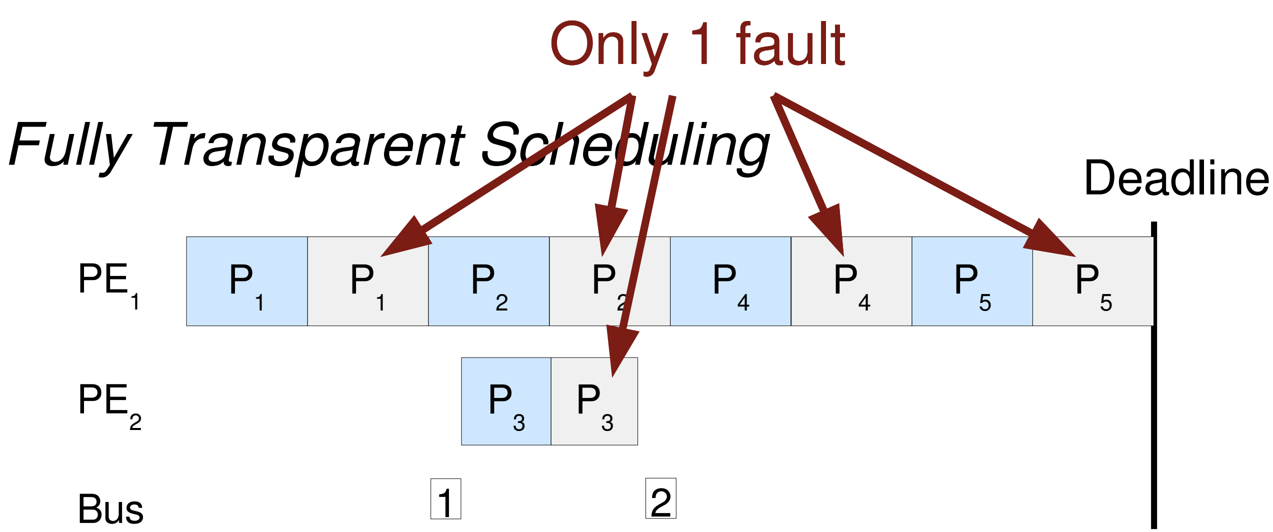
	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4

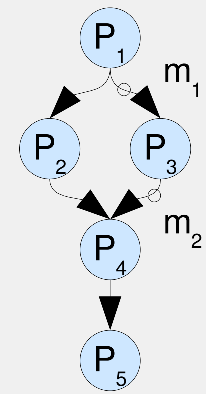




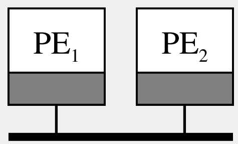
Fault-Tolerant Scheduling

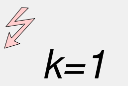
- Can be done faster
  - Sacrifice local transparency





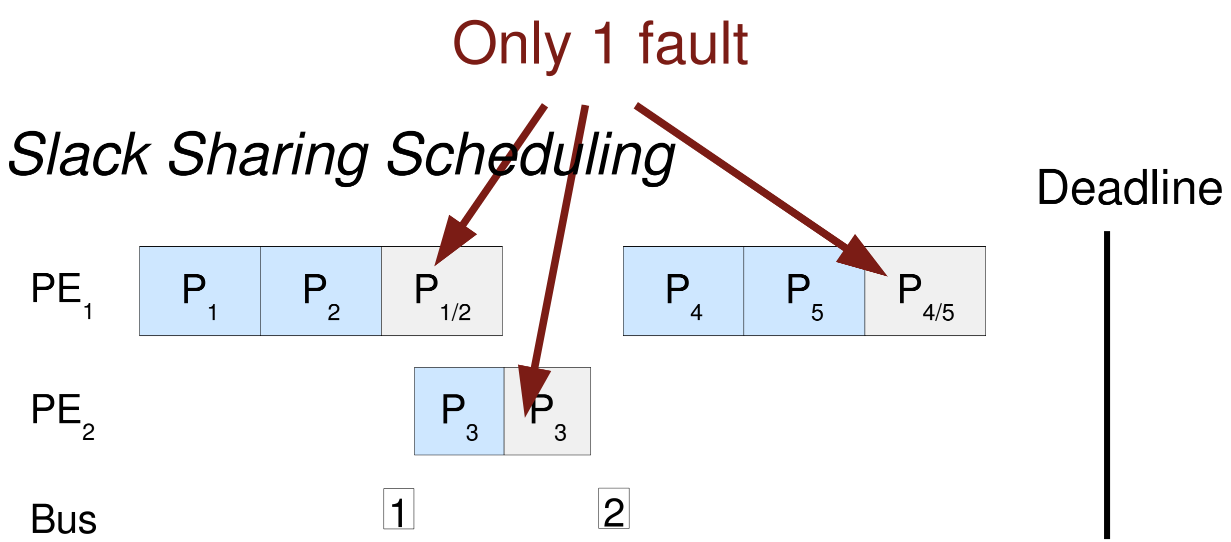
	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4

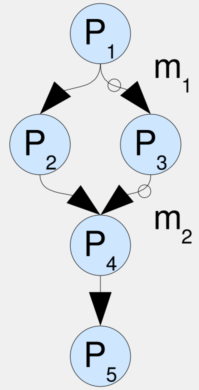




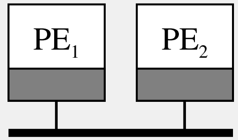
Fault-Tolerant Scheduling

- Can be done faster
  - Sacrifice local transparency
  - More complex online scheduler





	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4

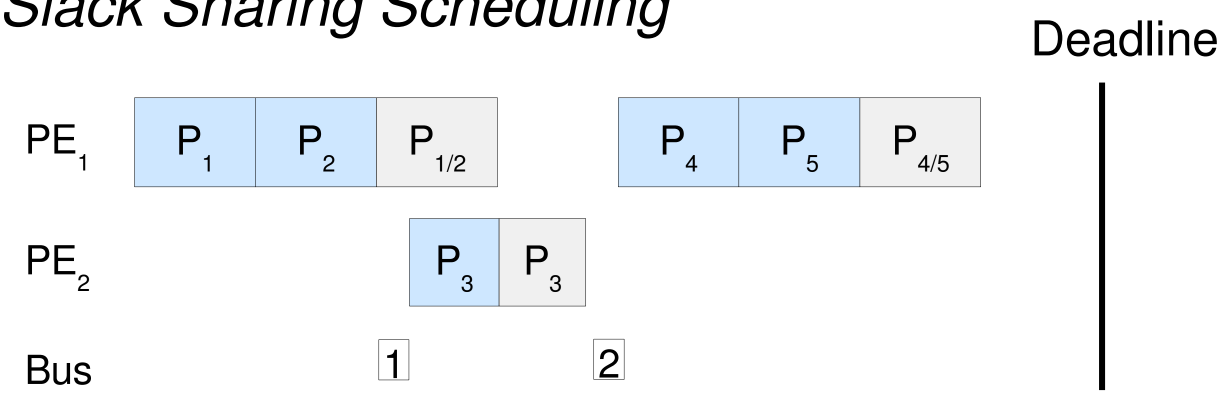


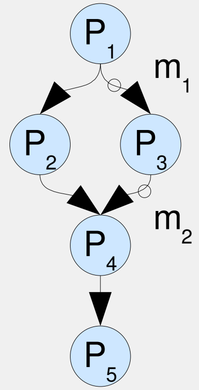
k=1

Fault-Tolerant Scheduling

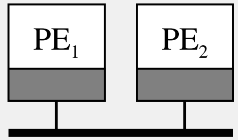
- Even faster
  - Sacrifice all transparency
  - Schedule for each fault scenario

Slack Sharing Scheduling





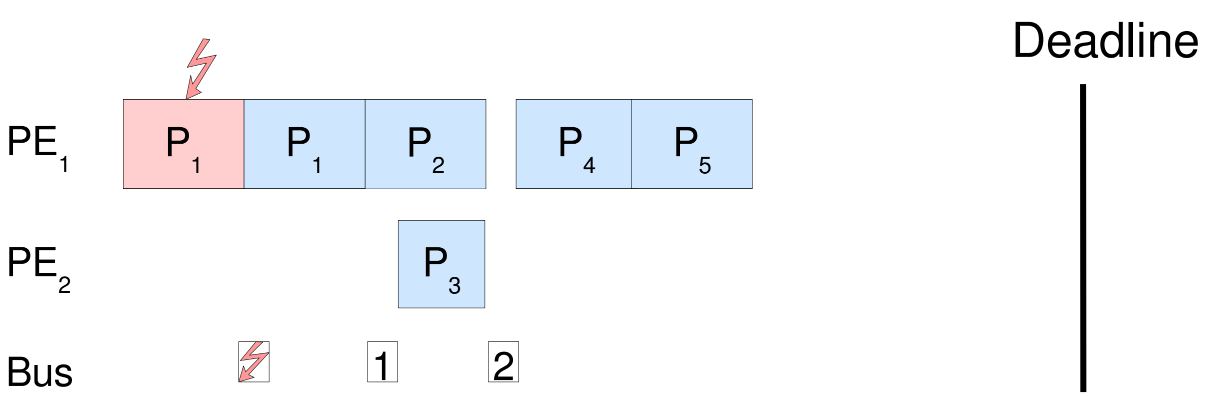
	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4

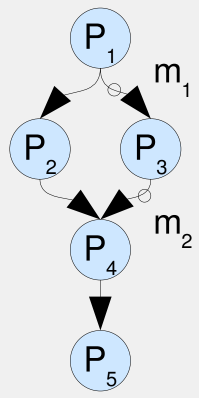


k=1

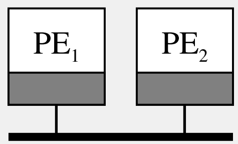
Fault-Tolerant Scheduling

- Even faster
  - Sacrifice all transparency
  - Schedule for each fault scenario
  - At most *k* re-executions





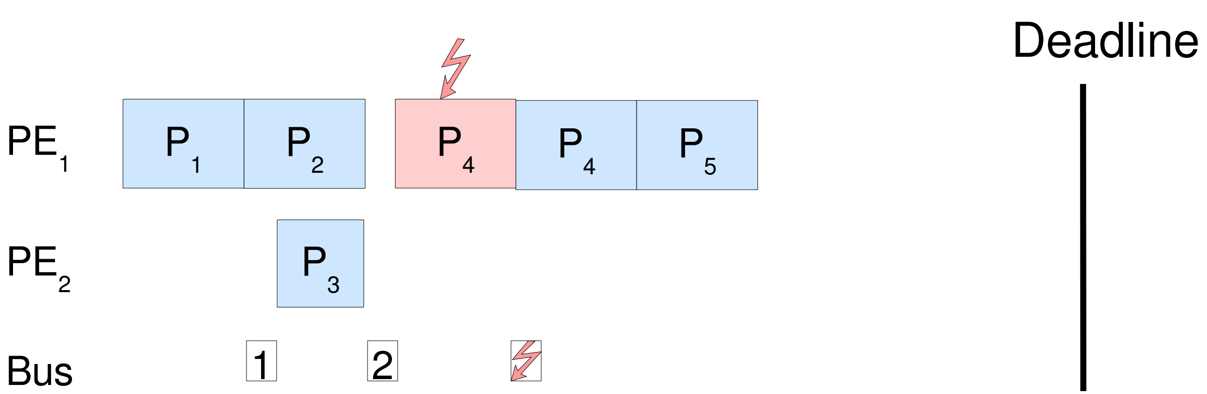
	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4

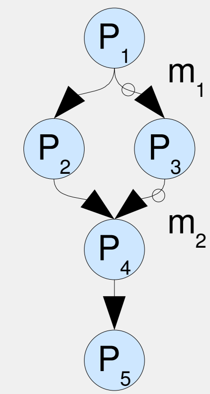


k=1

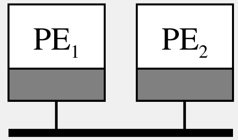
Fault-Tolerant Scheduling

- Even faster
  - Sacrifice all transparency
  - Schedule for each fault scenario
  - At most *k* re-executions





	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4

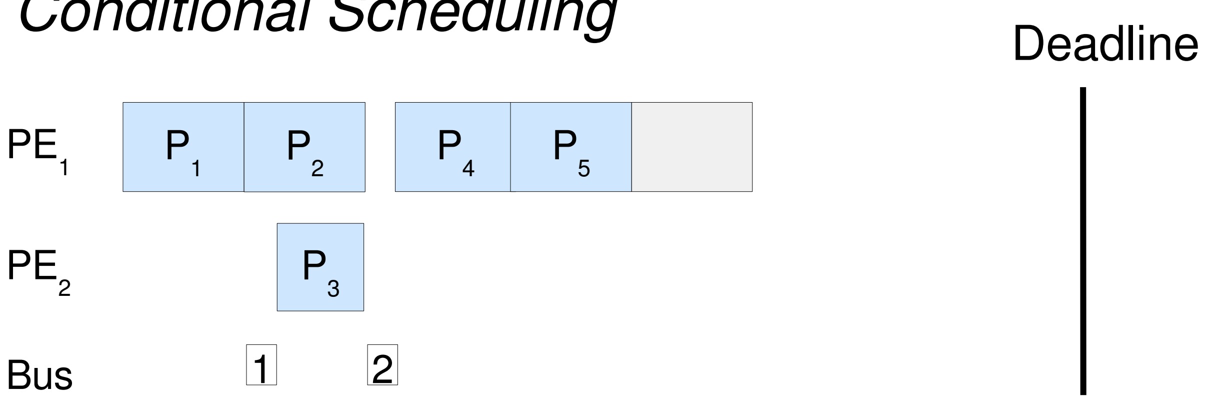


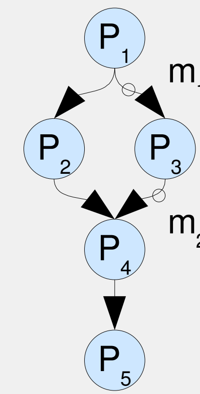
k=1

Fault-Tolerant Scheduling

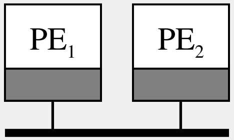
- Even faster
  - Sacrifice all transparency
  - Schedule for each fault scenario
  - At most *k* re-executions
  - All faults information is shared

Conditional Scheduling





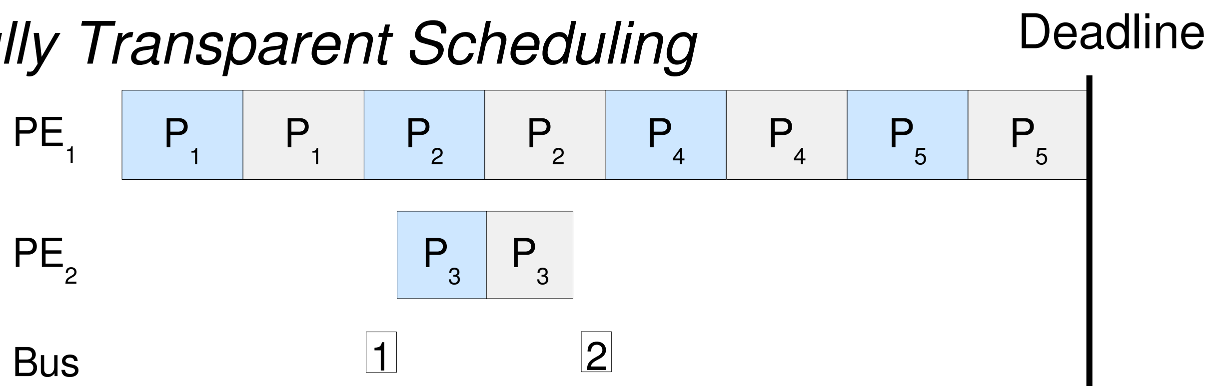
	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4



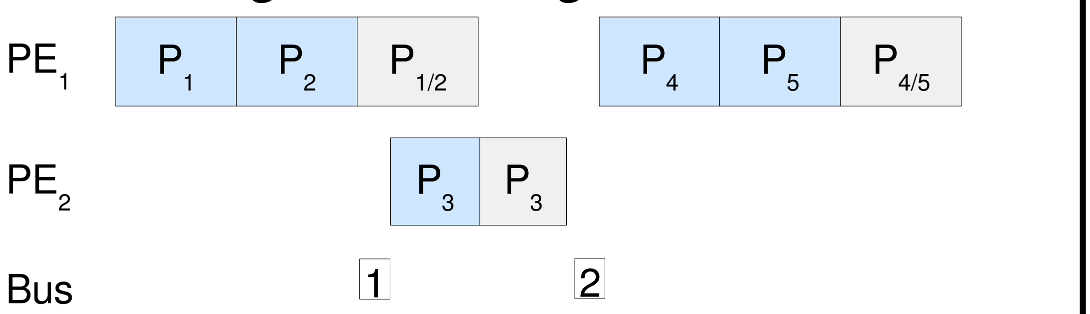
k=1

Fault-Tolerant Schedulings

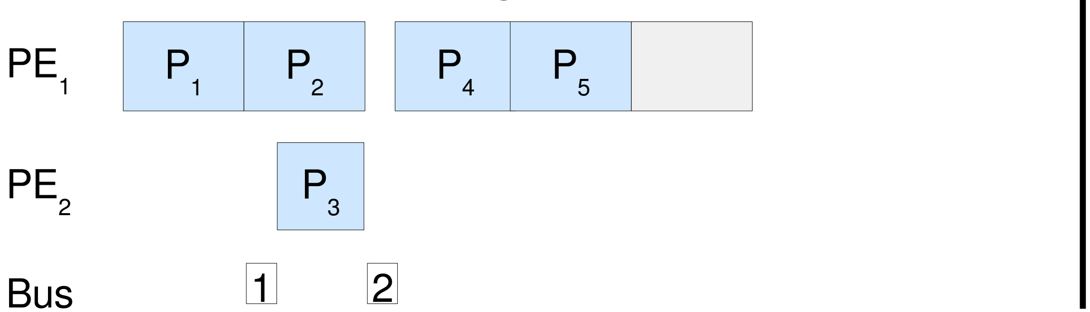
Fully Transparent Scheduling

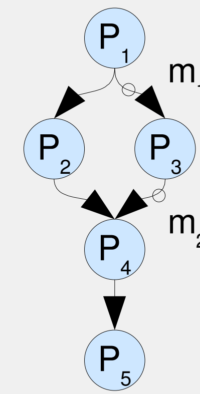


Slack Sharing Scheduling

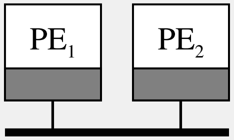


Conditional Scheduling



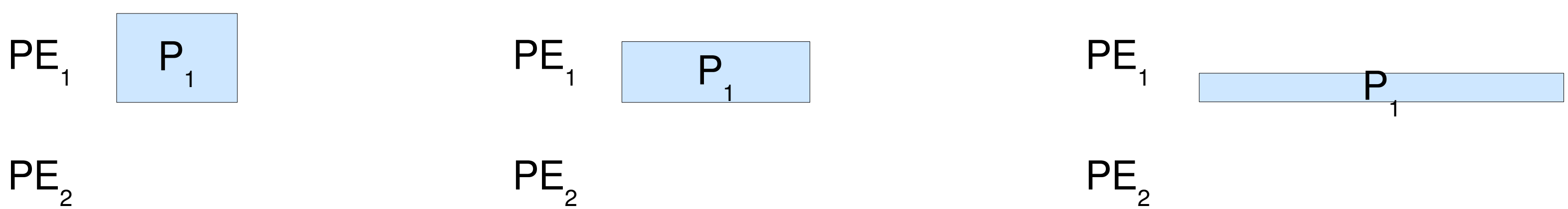


	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4



k=1

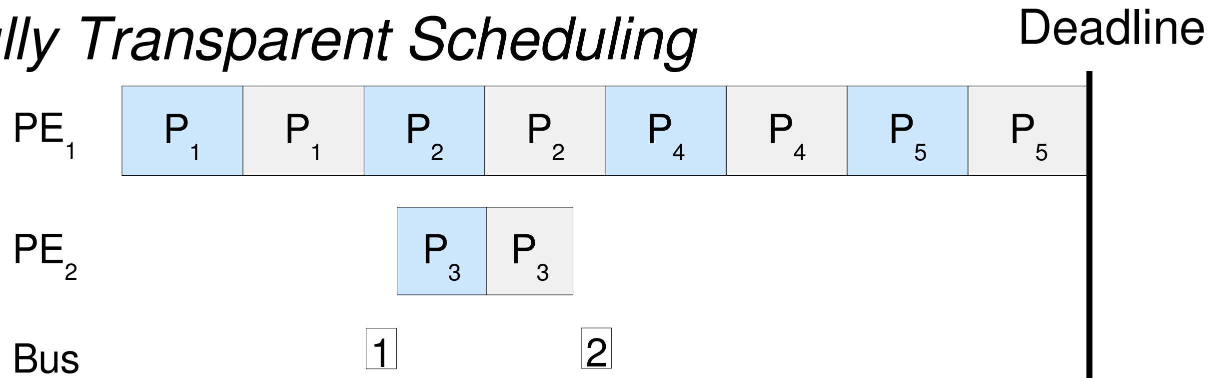
- Goal: minimise energy consumption
  - Dynamic voltage scaling



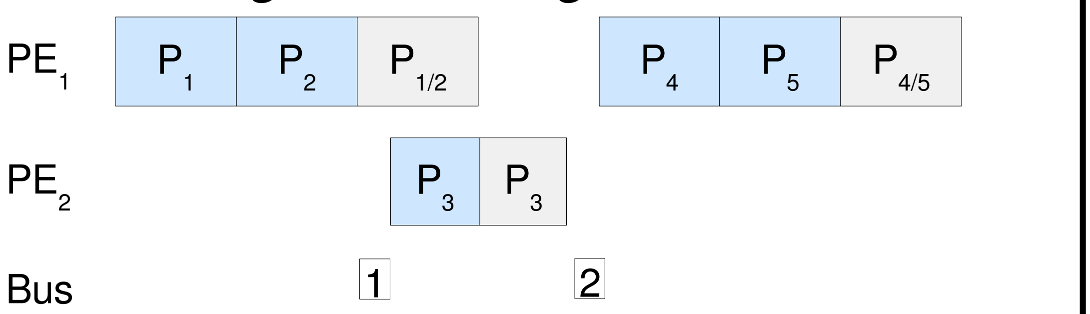
100% V_{ss}	66% V_{ss}	33% V_{ss}
100% E_0	44% E_0	11% E_0

Energy Management

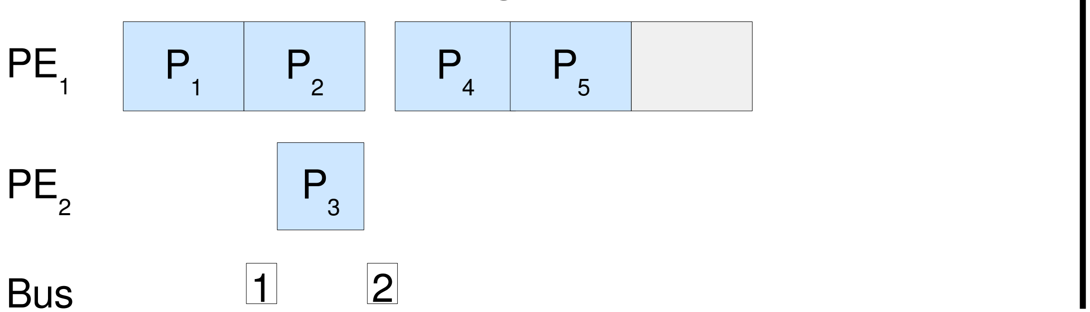
Fully Transparent Scheduling

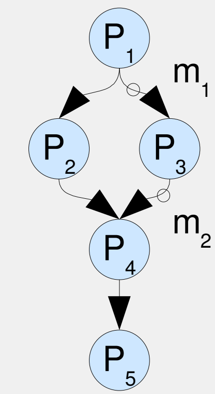


Slack Sharing Scheduling

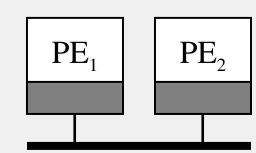


Conditional Scheduling





	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4

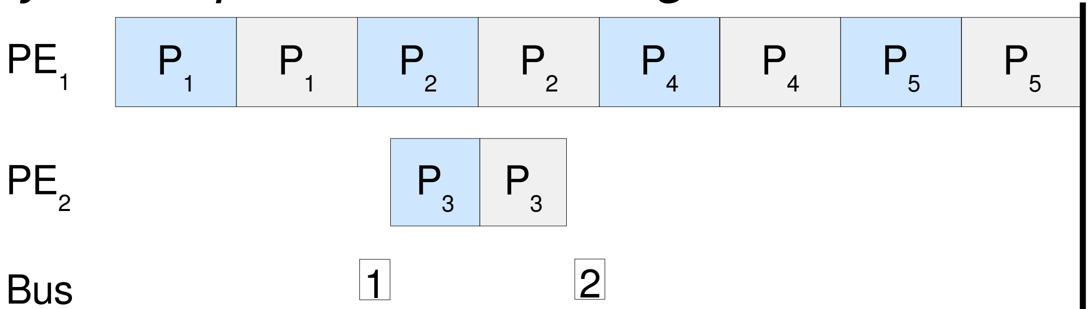


k=1

Energy Management

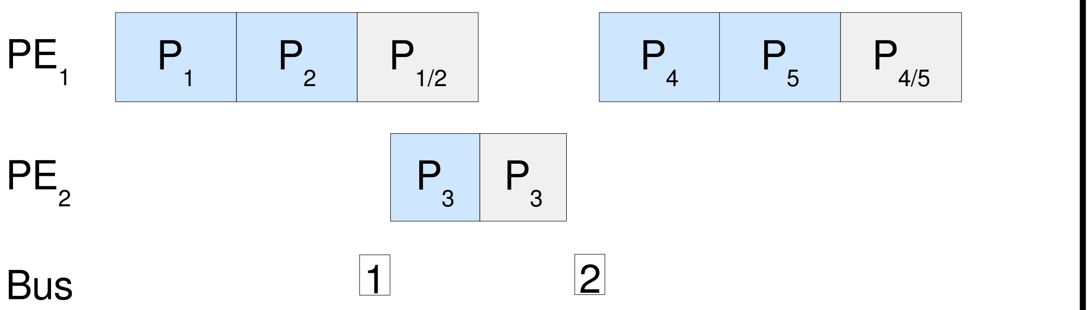
Fully Transparent Scheduling

Deadline

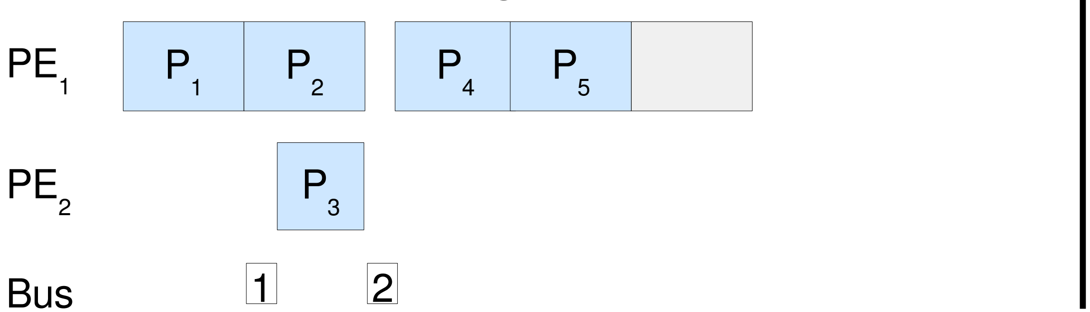


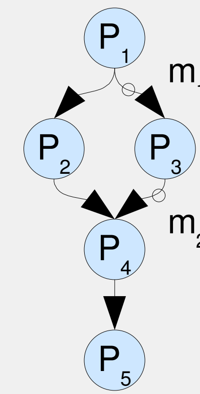
100% E_0

Slack Sharing Scheduling

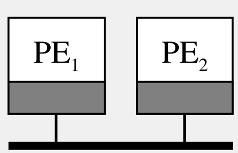


Conditional Scheduling



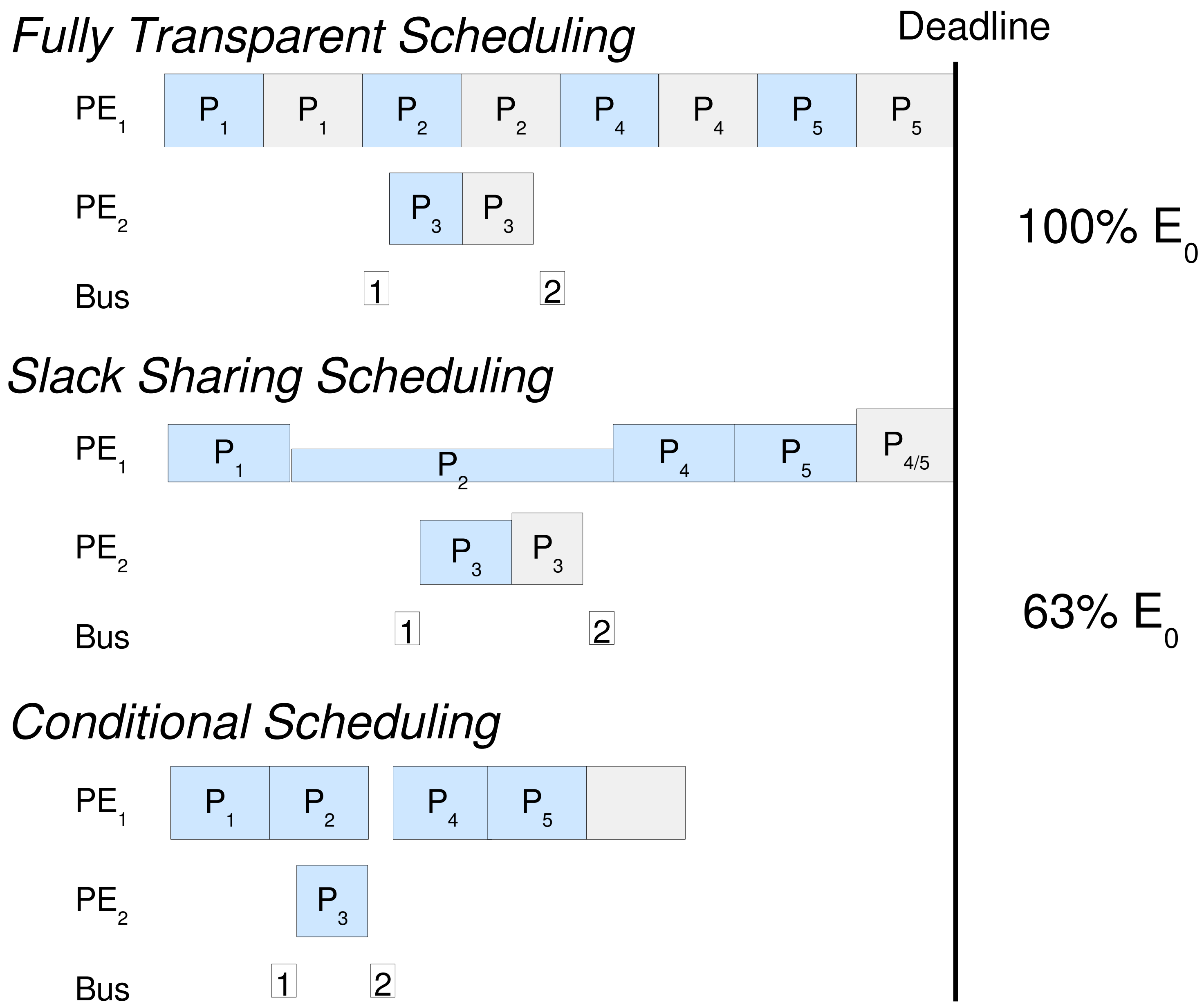


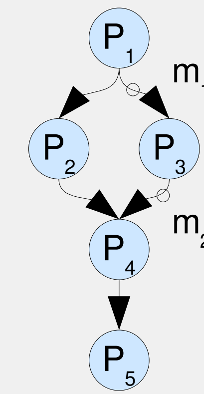
	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4



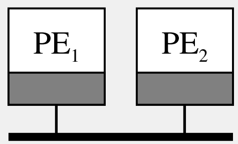
k=1

Energy Management



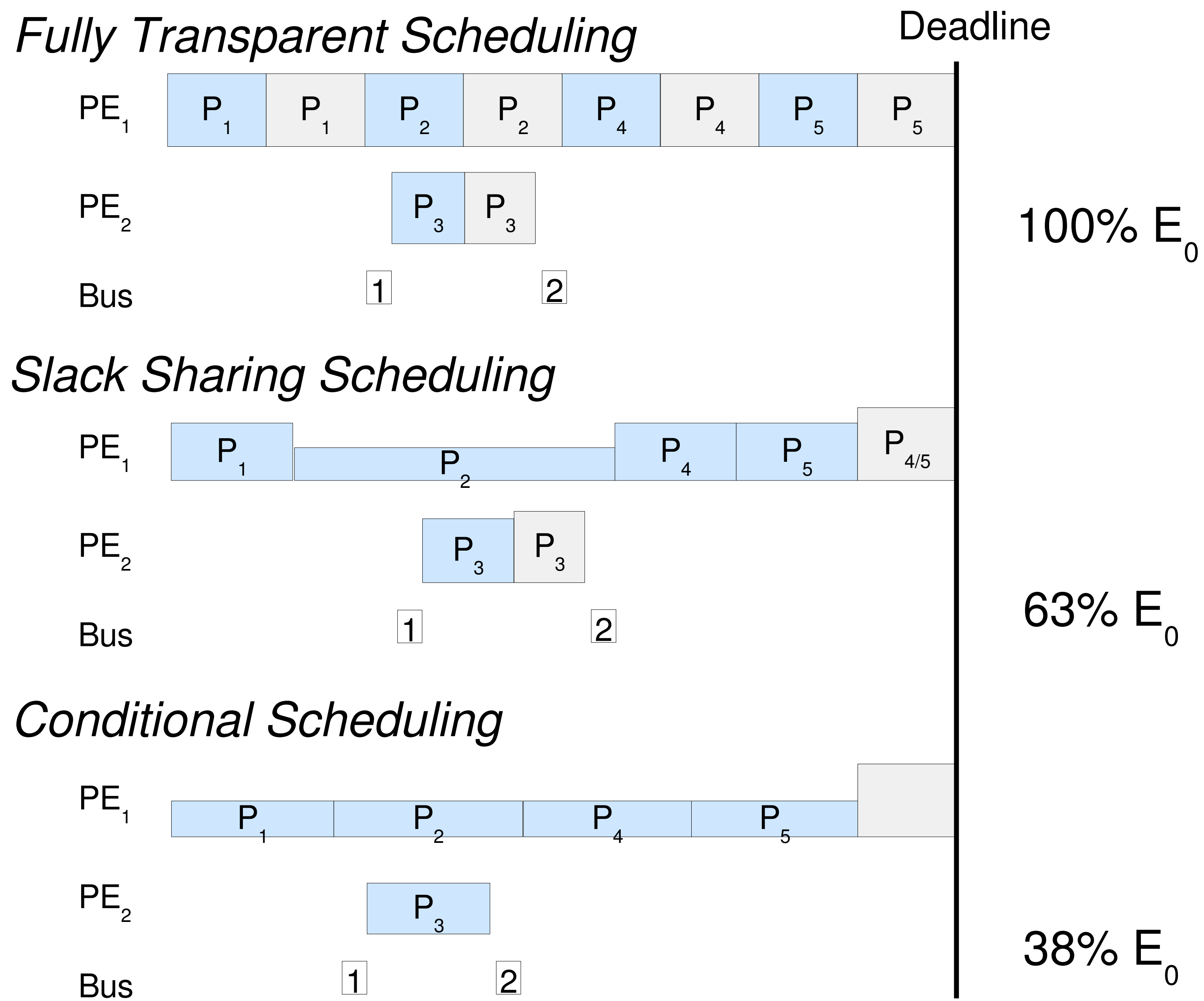


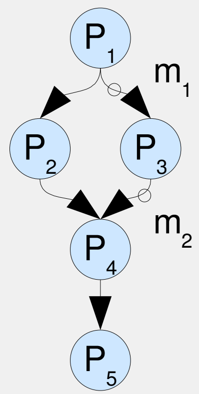
	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4



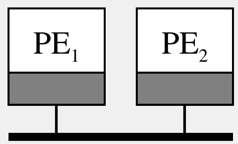
k=1

Energy Management



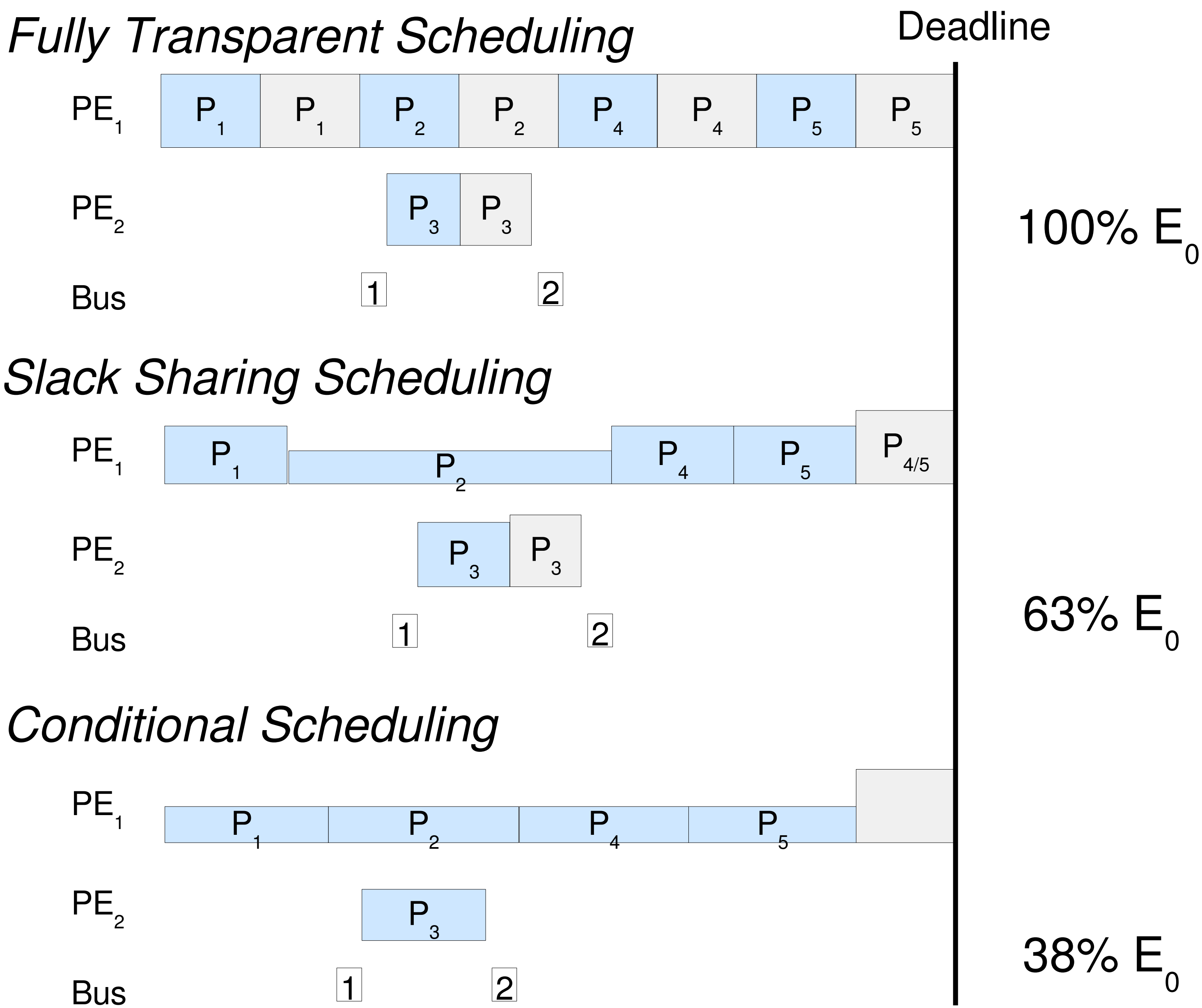


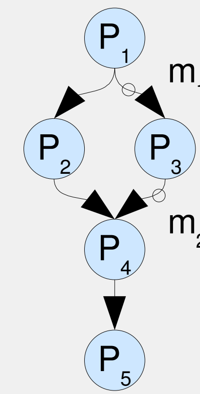
	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4



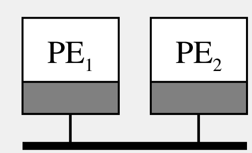
k=1

Energy Management





	PE_1	PE_2
P_1	4	4
P_2	4	4
P_3	3	3
P_4	4	4
P_5	4	4



k=1

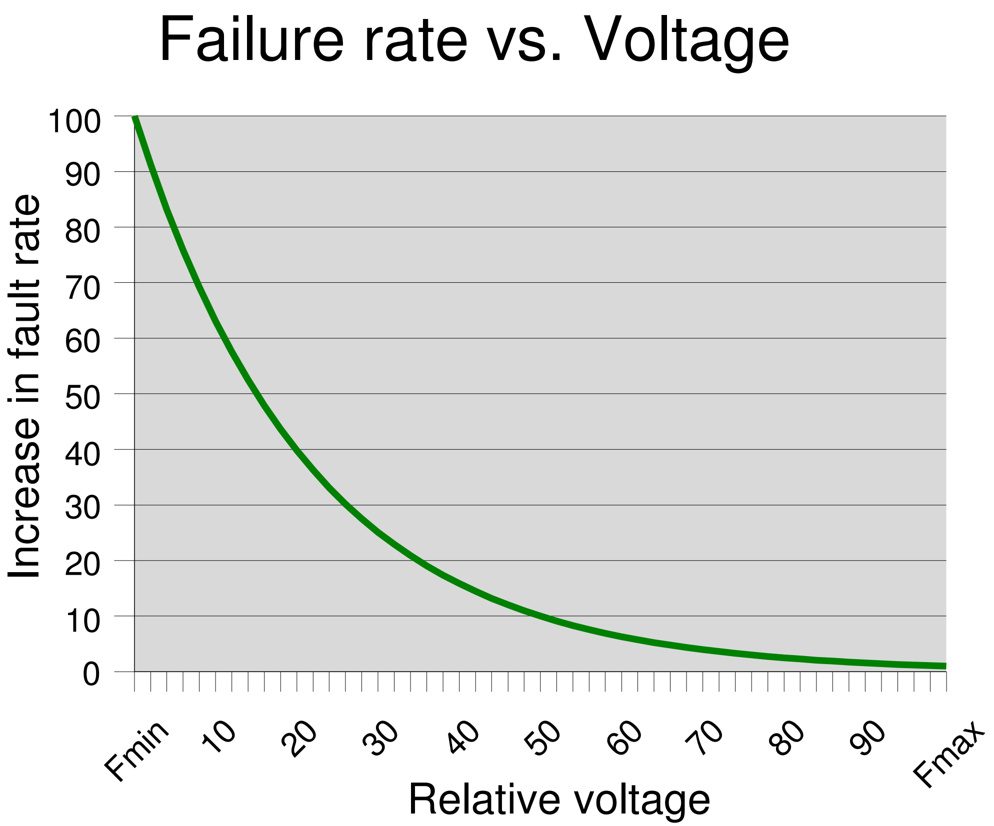
Reliability and Energy

- Lower voltage
  - Critical energy is lowered
  - Probability of faults increases
  - Circuit operates slower
- Lower frequency
  - Longer execution time
  - Probability of faults increases

Reliability and Energy

- Exponential model

$$\lambda(f) = \lambda_0 10^{\frac{d(1-f)}{1-f_{min}}}$$

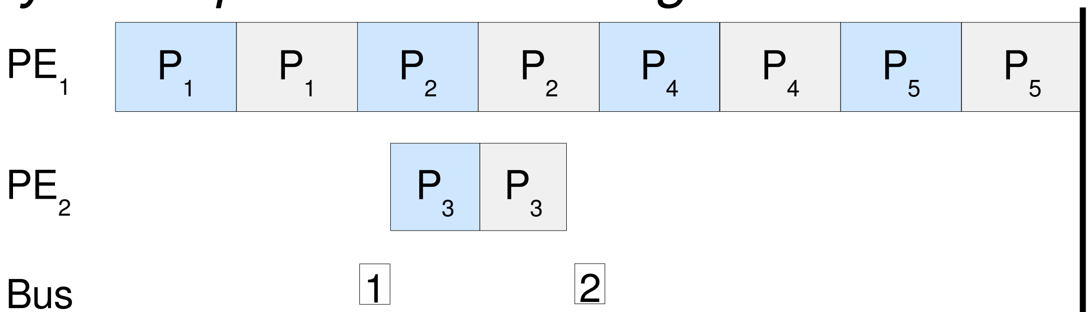


Dakai Zhu, Reliability-Aware Dynamic Energy Management in Dependable Embedded Real-Time Systems, 2006

Energy Management

Fully Transparent Scheduling

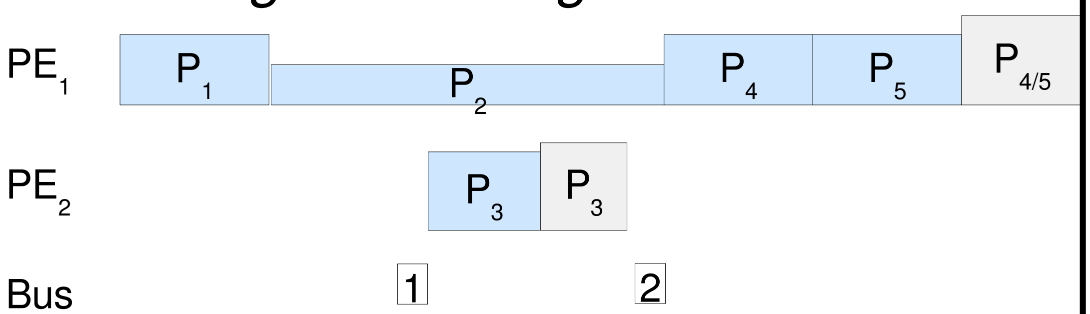
Deadline



R=0.999 999 999 93

100% E_0

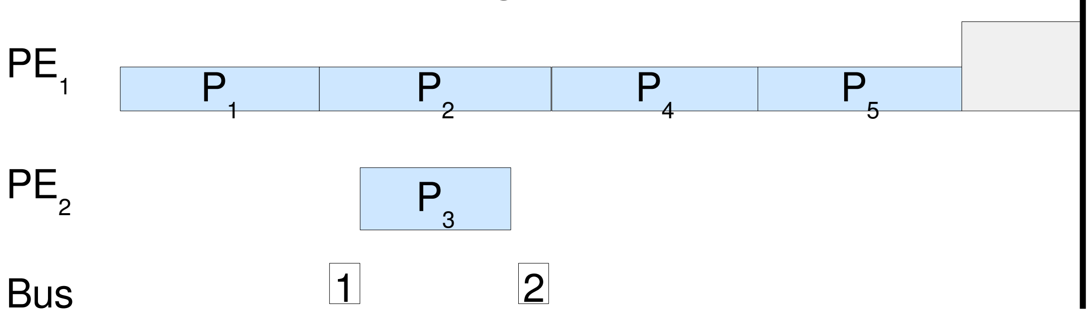
Slack Sharing Scheduling



R=0.999 999 999 25

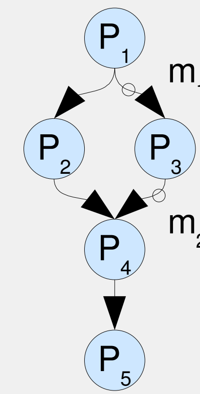
63% E_0

Conditional Scheduling

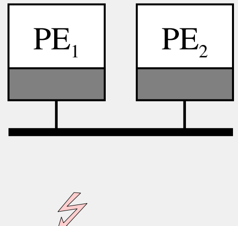


R=0.999 999 958 208

38% E_0



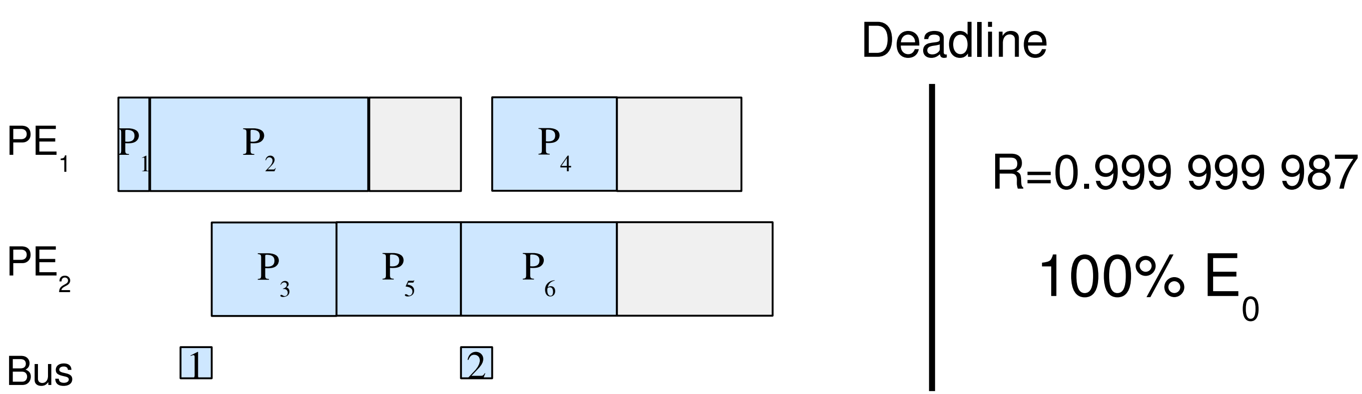
	PE$_1$	PE$_2$
P$_1$	4	4
P$_2$	4	4
P$_3$	3	3
P$_4$	4	4
P$_5$	4	4

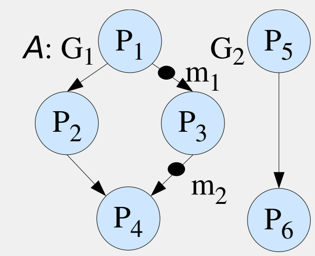


k=1

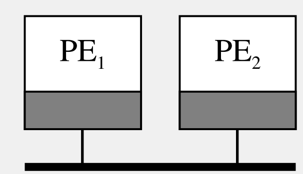
Energy/Reliability Trade-off

- Reliability goal: 0.999 999 9





	N_1	N_2
P_1	10	X
P_2	70	X
P_3	X	40
P_4	40	X
P_5	X	40
P_6	X	50



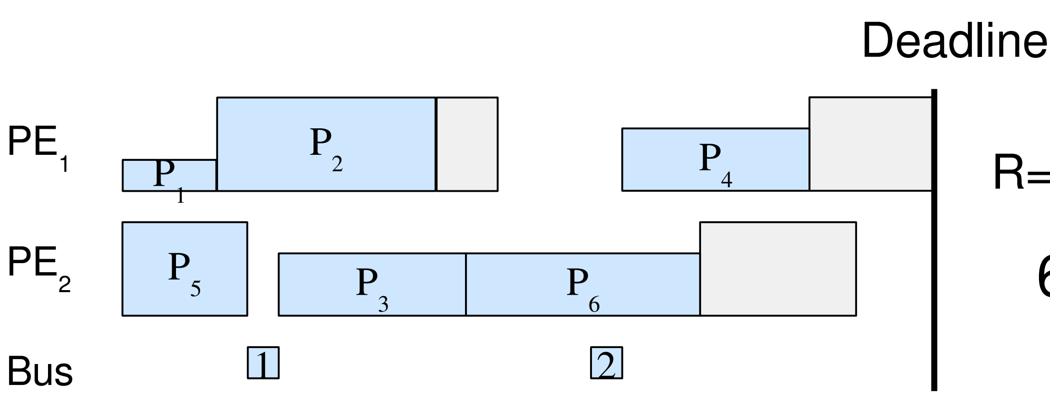
Voltage levels

N_1	100%	66%	33%
N_2	100%	66%	33%

k = 1

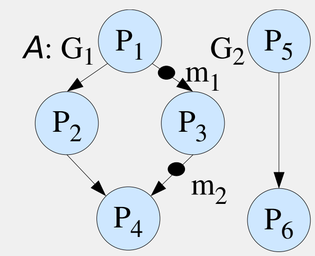
Energy/Reliability Trade-off

- Reliability goal: 0.999 999 9
- Set reliability as hard constraint

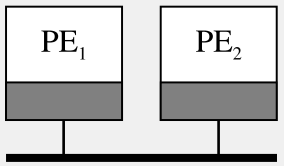


R=0.999 999 878

68% E_0



	N_1	N_2
P_1	10	X
P_2	70	X
P_3	X	40
P_4	40	X
P_5	X	40
P_6	X	50



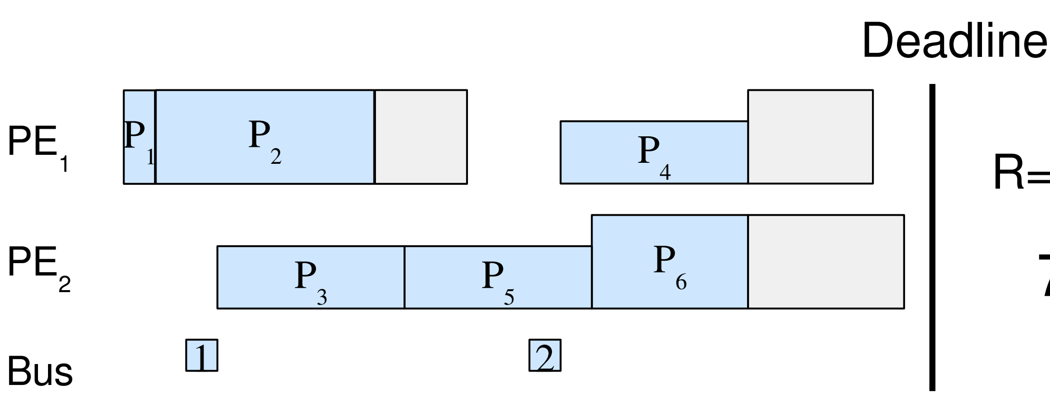
Voltage levels

N_1	100%	66%	33%
N_2	100%	66%	33%

k = 1

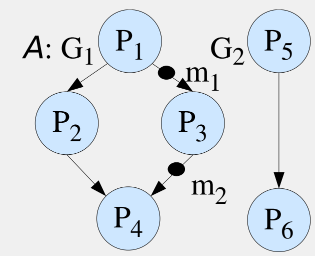
Energy/Reliability Trade-off

- Reliability goal: 0.999 999 9
- Set reliability as hard constraint
- Trade-off 5% energy
- Meets reliability goal

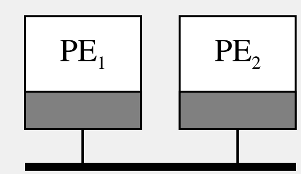


R=0.999 999 920

73% E_0



	N_1	N_2
P_1	10	X
P_2	70	X
P_3	X	40
P_4	40	X
P_5	X	40
P_6	X	50



Voltage levels

N_1	100%	66%	33%
N_2	100%	66%	33%

k = 1

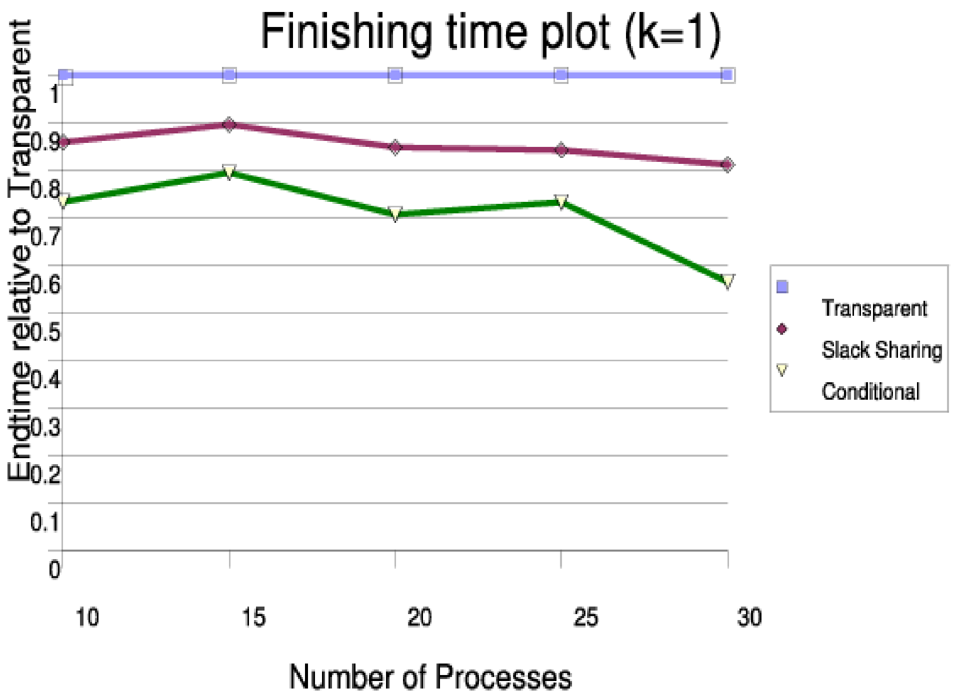
Problem Formulation

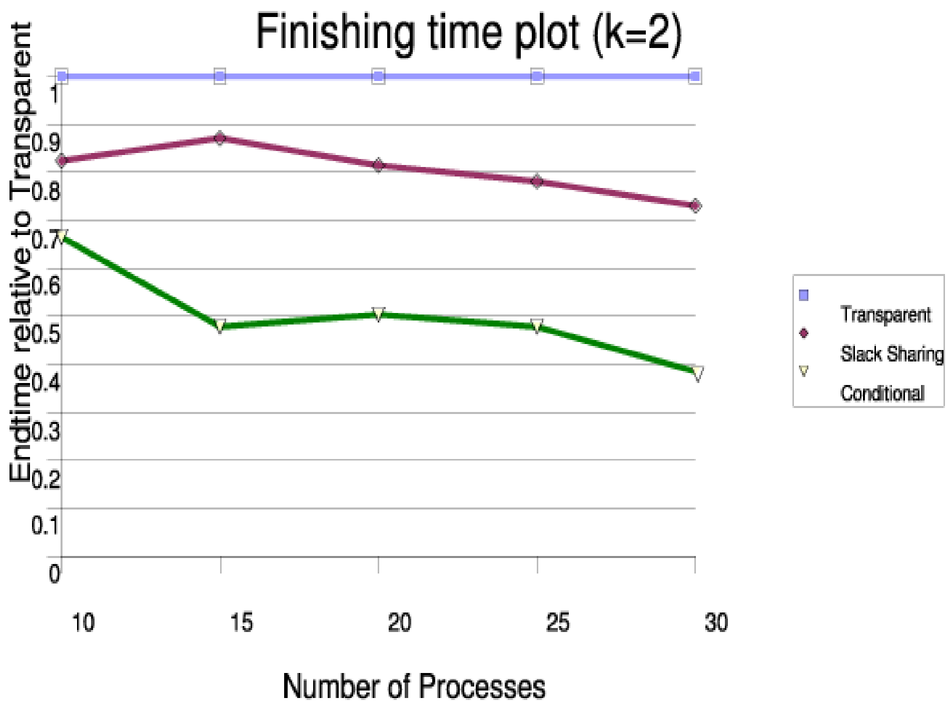
- Input
  - Application
  - Architecture
  - Reliability goal
- Decide
  - Fault-Tolerant Scheduling
  - Mapping
  - Fault-Tolerance Policy
- While optimising for
  - Energy
  - Under hard reliability goal

Implementation

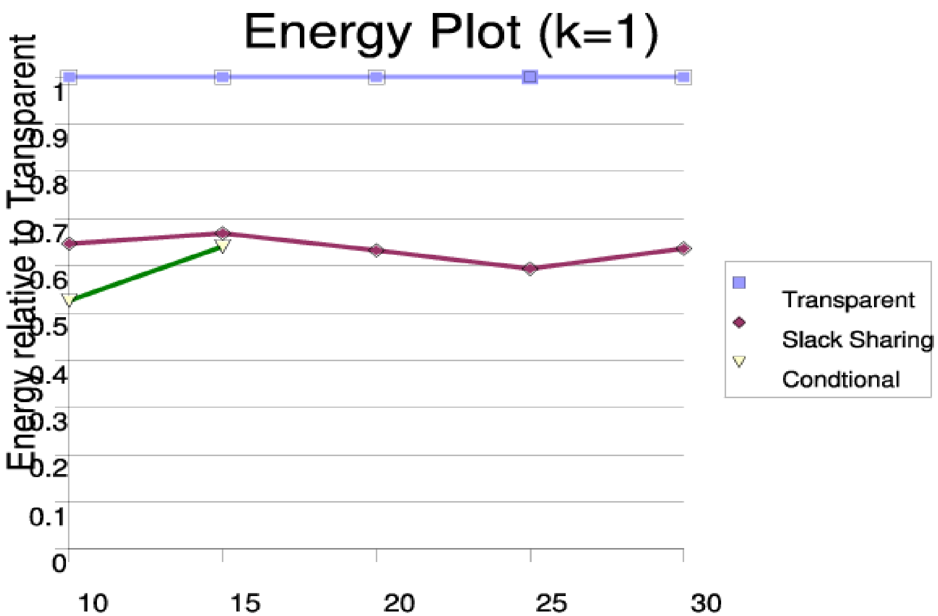
- Problem is NP-Complete
  - Normally solved using “best effort” heuristics
- Use constraint logic programming
  - Good performance with NP-completeness
  - Optimal solutions are feasible
  - Flexible model
  - ECLiPSe-CLP

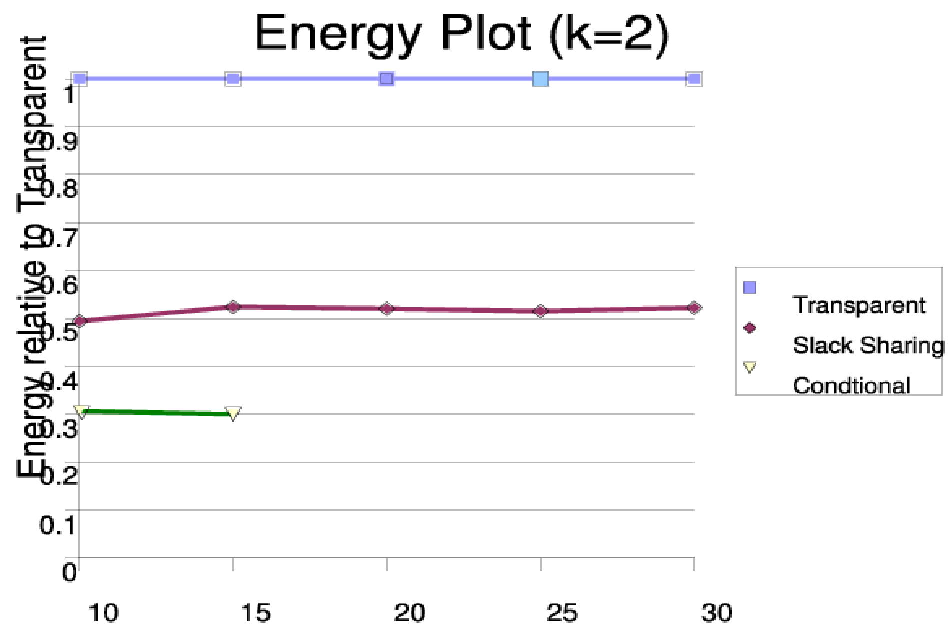
Comparison of Schedulers

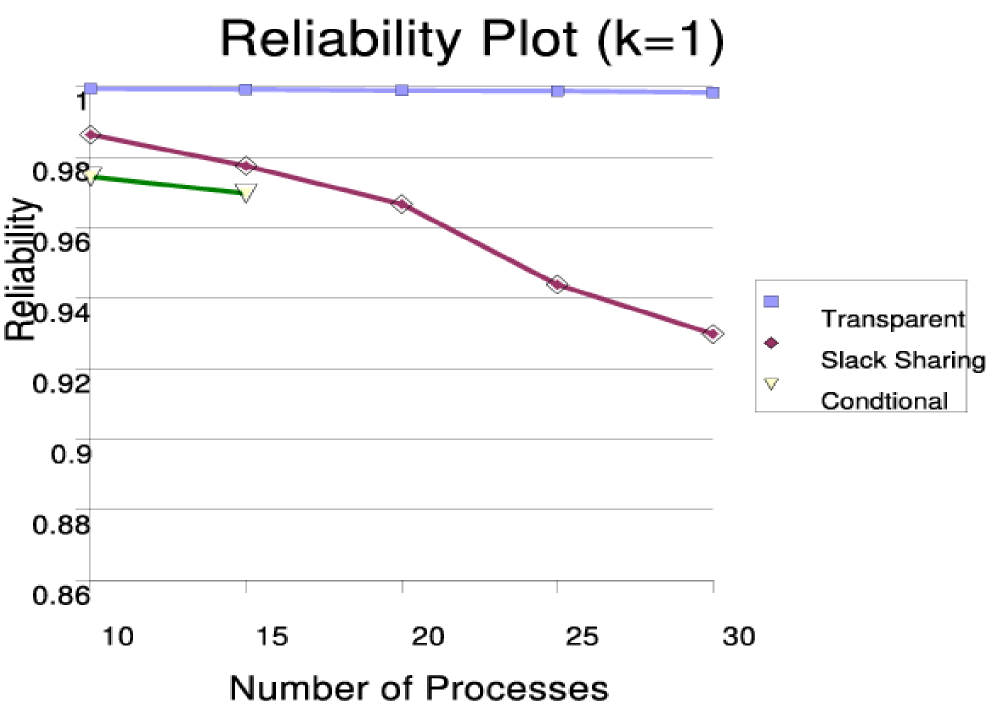


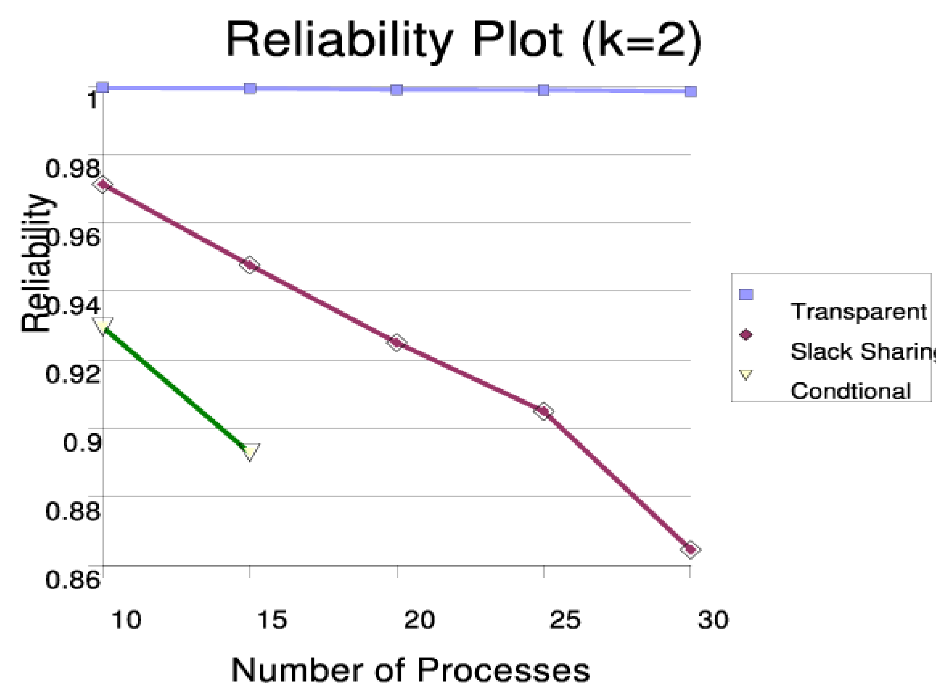


Comparison of Schedulers

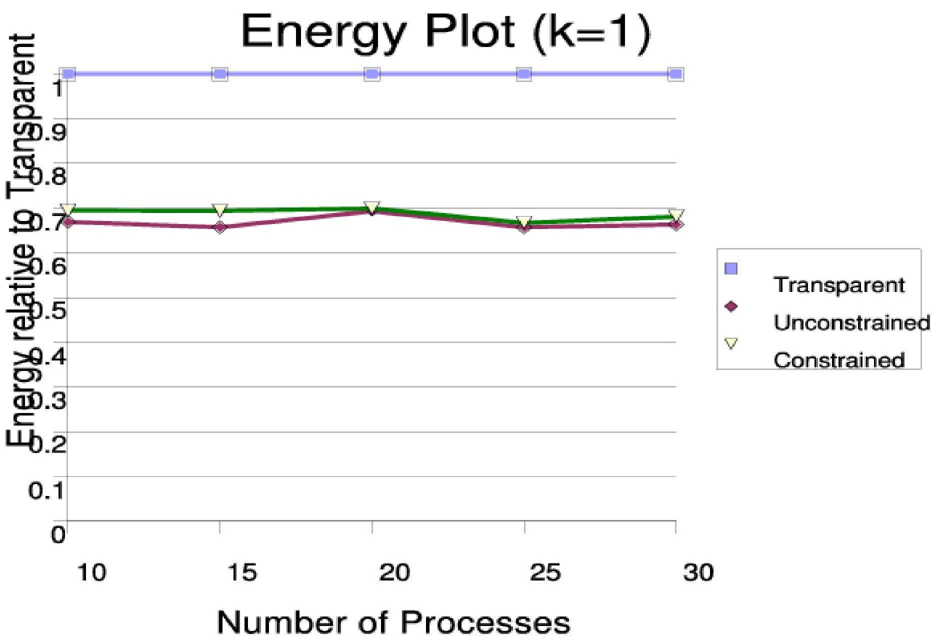


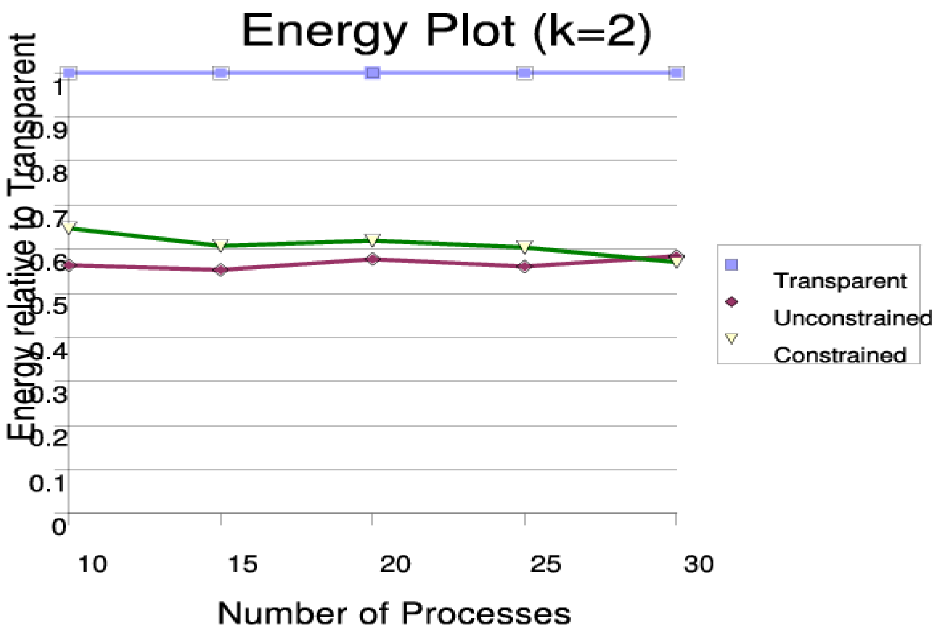


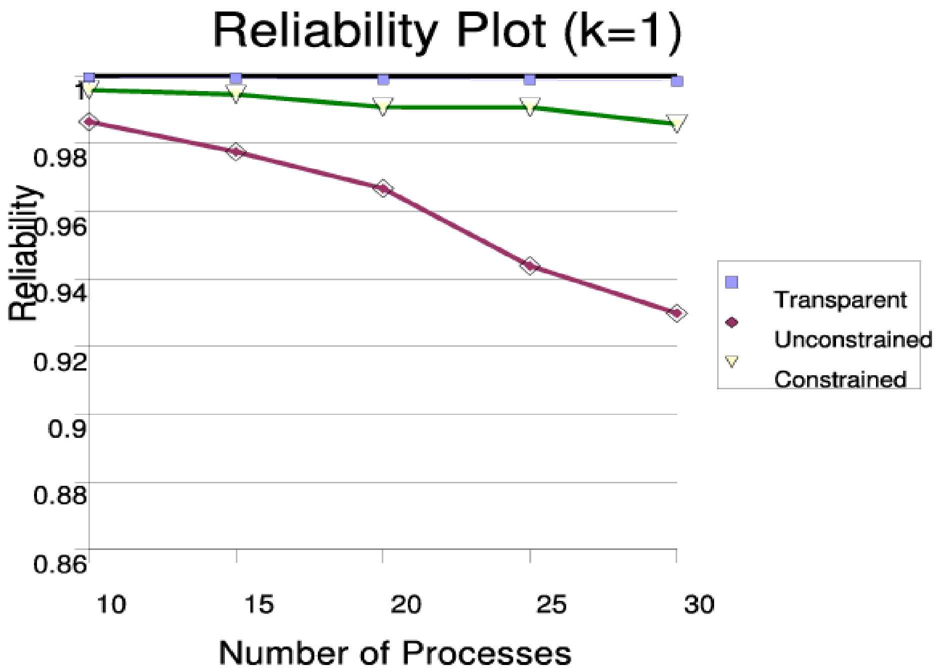


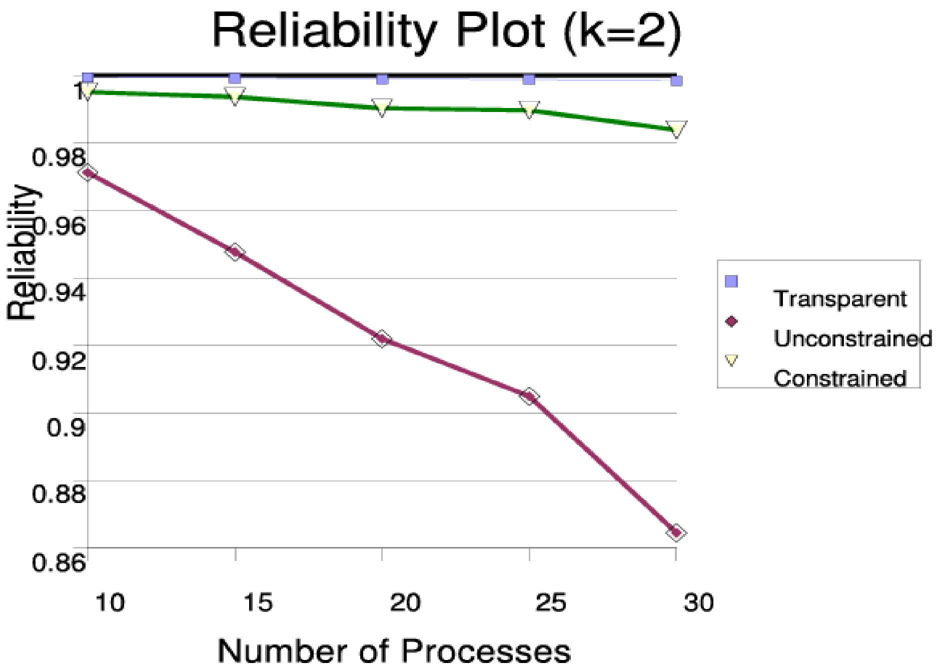


Reliability and Energy Trade-offs









Conclusions

- Design tool for doing
  - Fault tolerant scheduling
  - Mapping
  - Policy assignment
- Optimising for
  - Minimal energy
  - Hard constraints for timing and reliability
- Message:
  - Reliability can be met at little energy cost

Embedded Systems