

Technische Informatik

Universitätsstudium und HTL Ausbildung: Chancen und Herausforderungen

Ulrich Schmid
s@ecs.tuwien.ac.at

HTBLuVA Salzburg, Dezember 2012

Warum bin ich hier ?

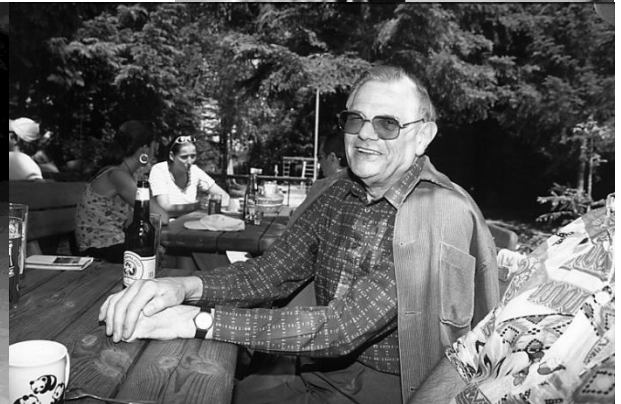
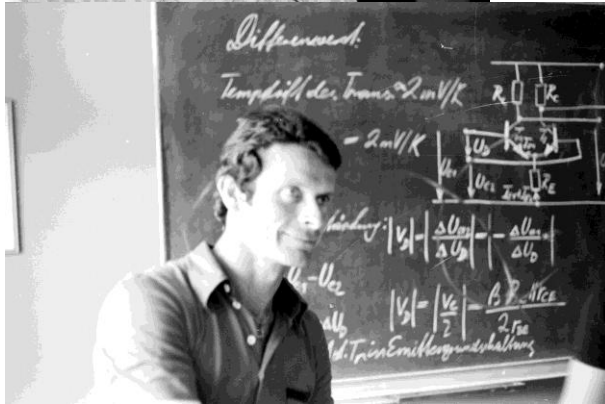
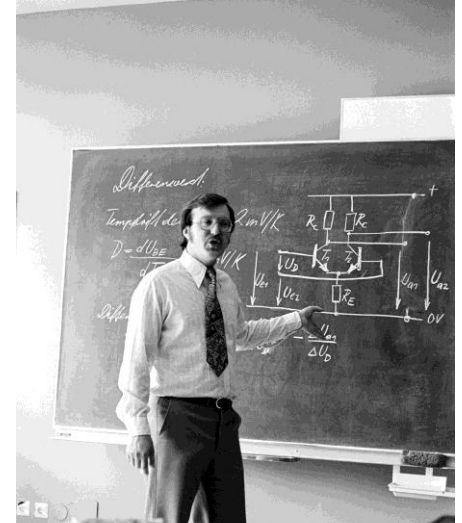
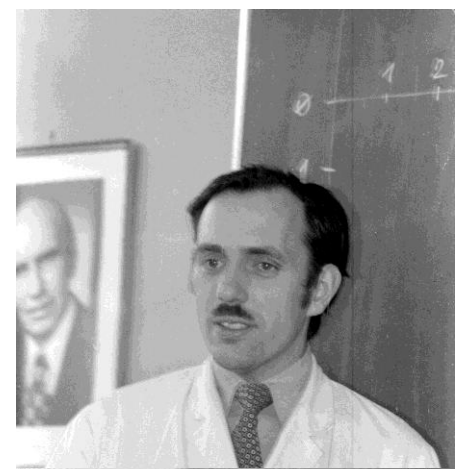
Aus
persönlichen
Gründen

In meiner Funktion als
Koordinator der TI -
Studien an der TU Wien

Aus grauer Vorzeit ...



© alle Fotos: J. Hahn











Warum bin ich hier ?

Aus
persönlichen
Gründen

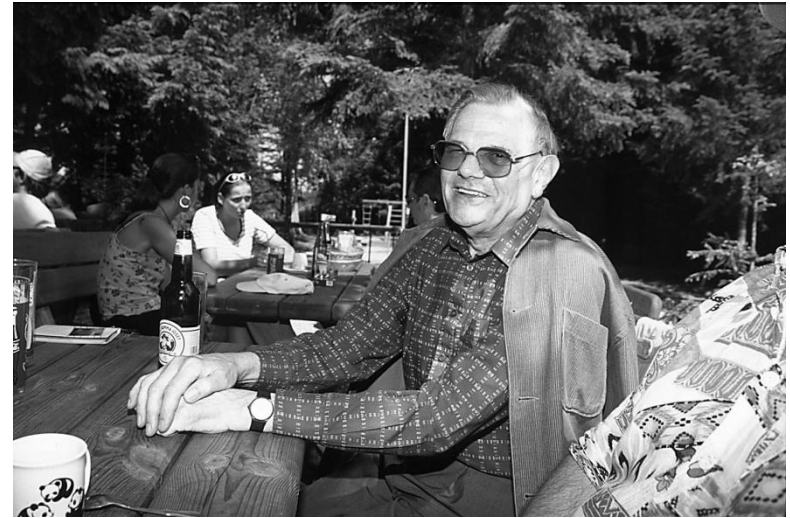
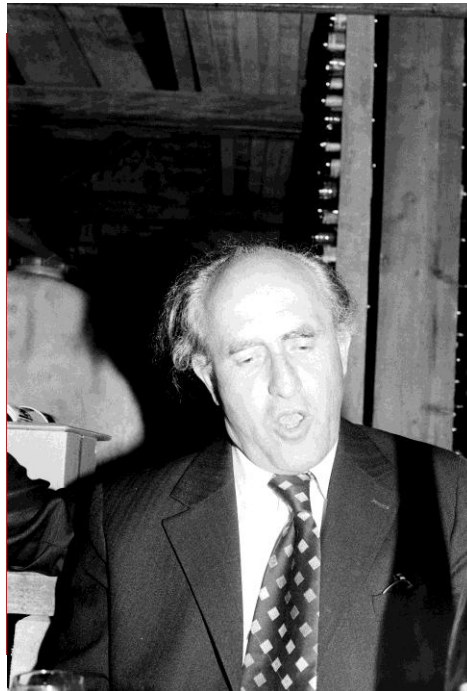
HTL St. Pölten

≡

Extrem vielfältiges Angebot, Wissen
und Fertigkeiten zu erwerben

Warum bin ich hier ?

Aus
persönlichen
Gründen



Warum bin ich hier ?

Aus
persönlichen
Gründen

¬ HTL St. Pölten → ¬ Wissenschaftler
→ ¬ TU-Professor

Warum bin ich hier ?

Ein Ingenieur, ein Physiker und ein Mathematiker übernachten in einem Hotel ...

■ Probleme der TU Wien - Informatik:

- 💣 Generell: Wissenschaft vs. Engineering
- 💣 Falsche Vorstellungen von Zielen und Anforderungen eines akademischen Informatik-Studiums
- 💣 Fehlende Vorkenntnisse und Fertigkeiten der Studienanfänger

In meiner Funktion
als Koordinator der TI
- Studien an der TU
Wien

Warum bin ich hier ?

- Naheliegende Idee: **Abhilfe durch Verbesserung des „Interfaces“ zu den Schulen!**
- Leichter gesagt als getan: Erfordert
 - Herstellung einer „Win-Win“ Situation!
 - Ressourcensparende Realisierung!
 - Zeit ...
 - Ideen ...
 - Initiative ...

In meiner Funktion
als Koordinator der TI
- Studien an der TU
Wien

- ✓ Warum bin ich hier?)
- Versuch eines Einblicks in die TI-Forschung:
DARTS
 - Fault-tolerant distributed algorithms
 - Performance analysis using max,+ algebra
- Technische Informatik an der TU Wien
- Probleme
- Lösungsideen

Versuch eines Einblicks in die TI-Forschung



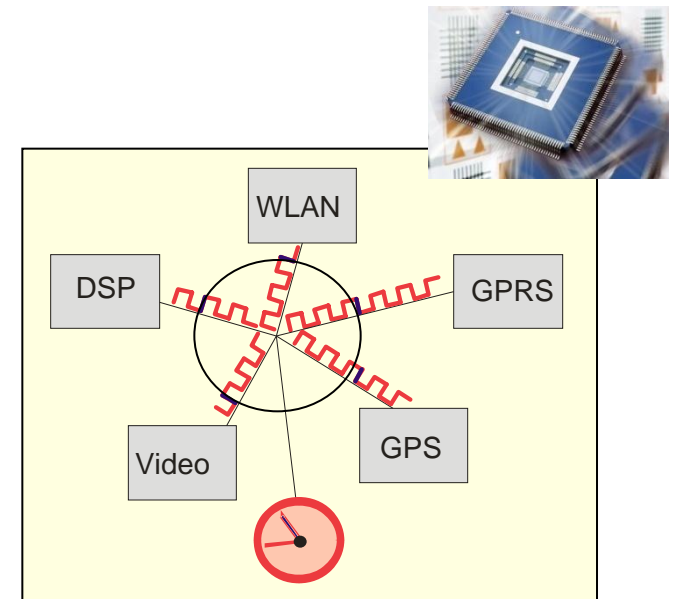
[Switch language: English]

Classic synchronous paradigm

- ❖ **Concept:** Common notion of time for entire chip
- ❖ **Method:** Single crystal oscillator
Global, phase-accurate clock tree

Disadvantages

- Cumbersome clock tree design (physical limits!)
- High power consumption
- Clock is **single point of failure!**



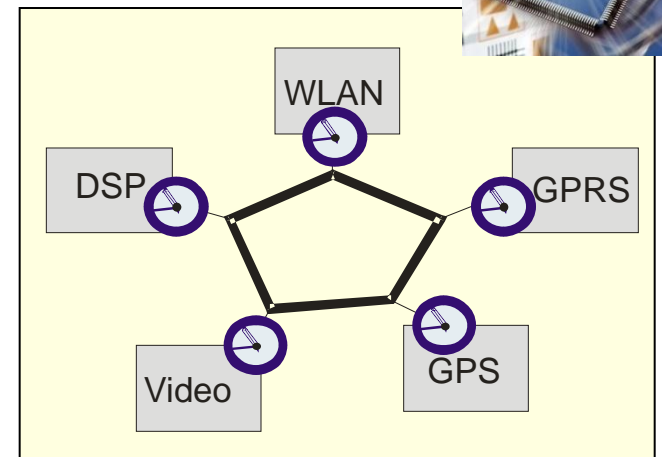
Alternative: DARTS clocks

- ❖ **Concept:** Multiple synchronized tick generators
- ❖ **Method:** Distributed fault-tolerant tick generation algorithm
Implemented in (asynchronous) digital logic

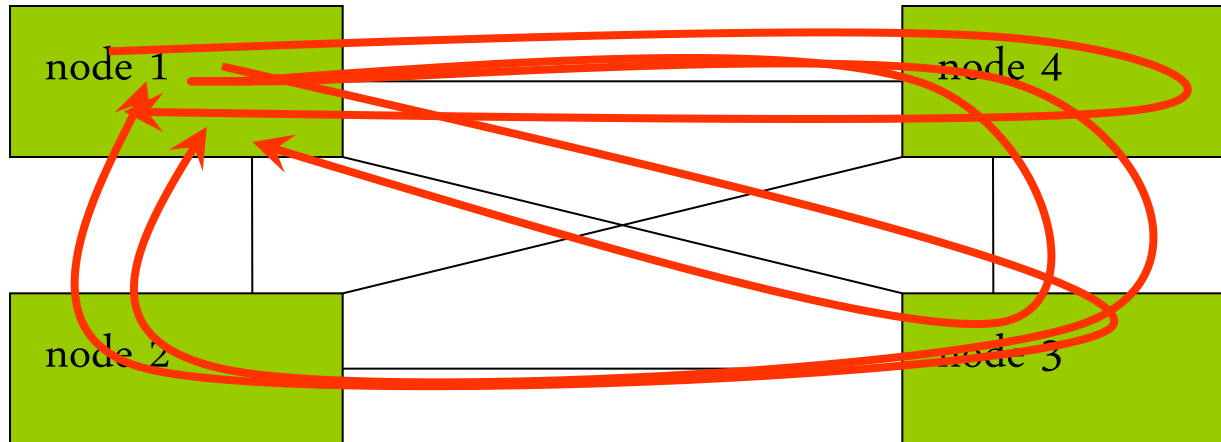
<http://ti.tuwien.ac.at/ecs/research/projects/darts>

Advantages

- Reasonable synchrony
- Uncritical clock distribution
- Clock is **no single point of failure!**



Fault-tolerant distributed ring oscillator:



➡ **Self-oscillating**

➡ **Fault-tolerant**

The DARTS FT Distributed Algorithm

For $n \geq 3f + 1$ and up to f node failures,
with (small) e-t-e delays $\in [d, d+\varepsilon]$:

- Suppose node p sends $tick(C+1)$ at time t
- Then, node q also sends $tick(C+1)$ by time $t+d+2\varepsilon$

⇒ **Clock ticks occur approximately at the same time**

On init

→ send $tick(0)$ to all; $C := 0$;

If got $tick(l)$ from $f+1$ nodes **and** $l > C$

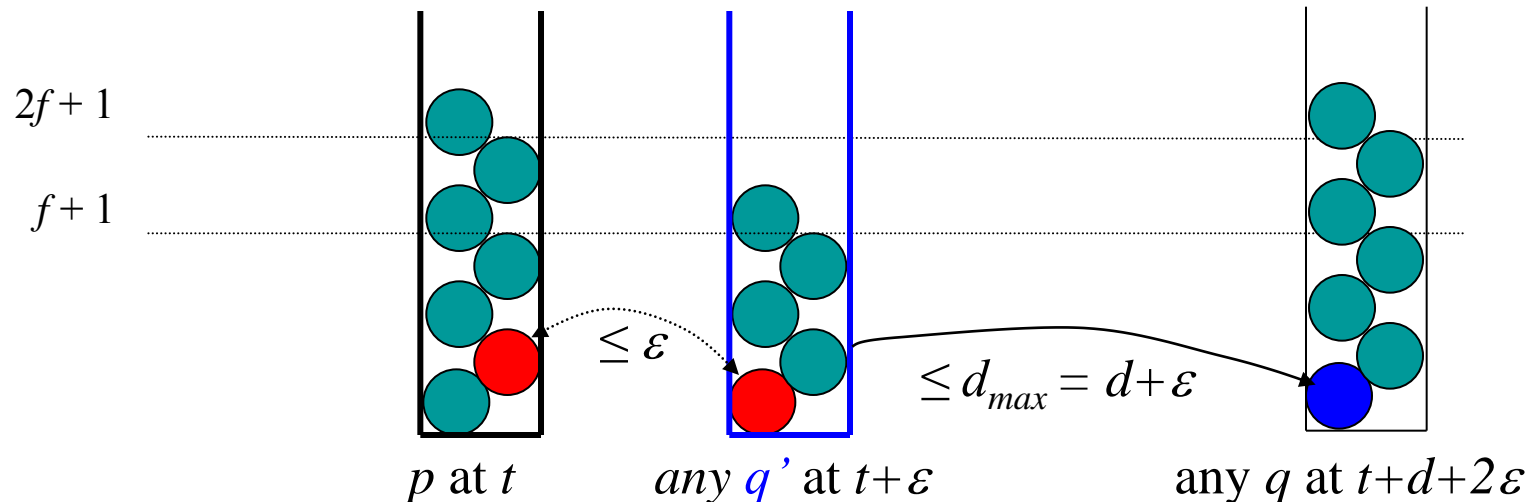
→ send $tick(C+1), \dots, tick(l)$ to all;

$C := l$;

If got $tick(C)$ from $2f+1$ nodes

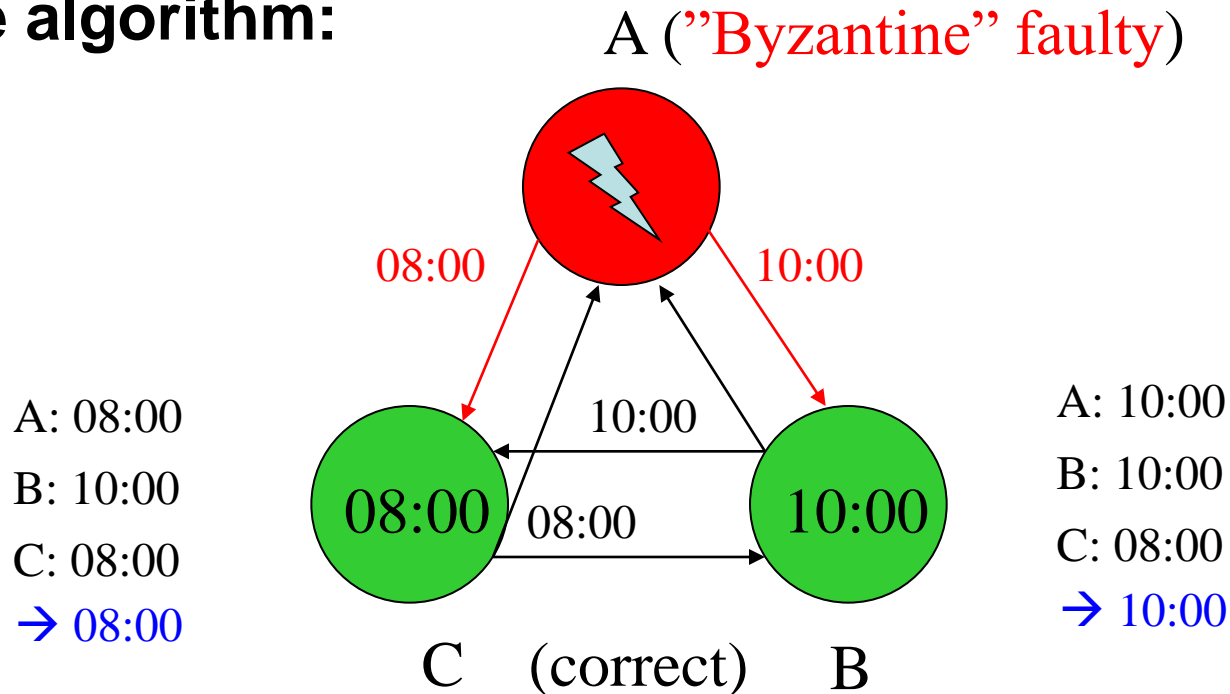
→ send $tick(C+1)$ to all;

$C := C+1$;



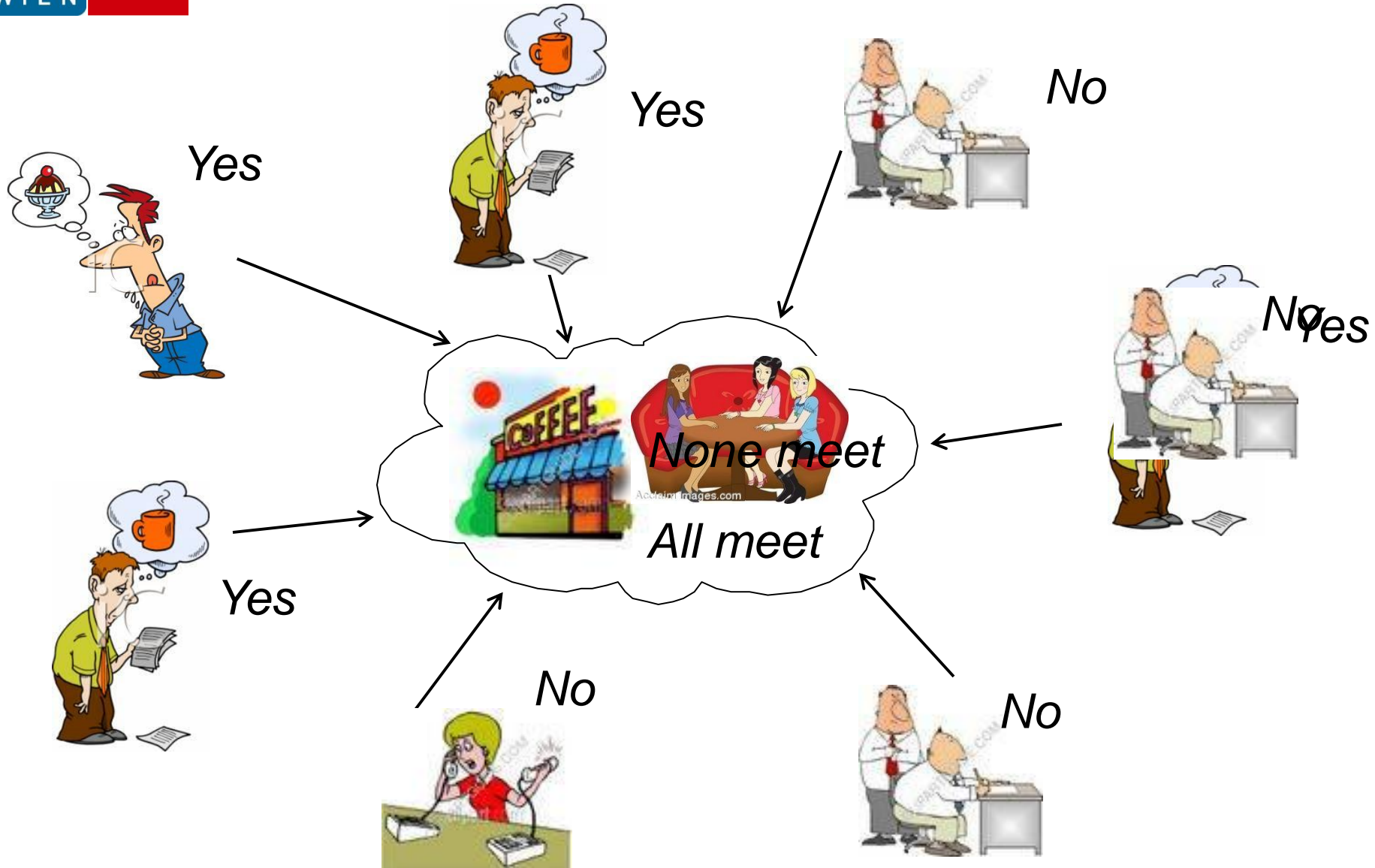
Why do Failures hurt (so much) ?

A simple algorithm:



- B and C never get closer together
- Majority ($n = 2f + 1$) not enough for f Byz. failures!
- Could we prove this for any algorithm ?

An Abstract Problem: Consensus



Consensus Properties

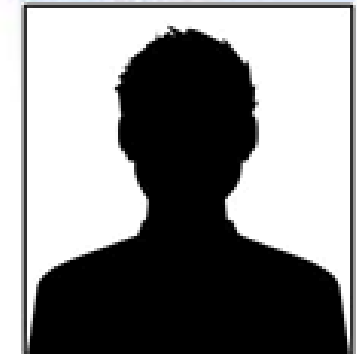
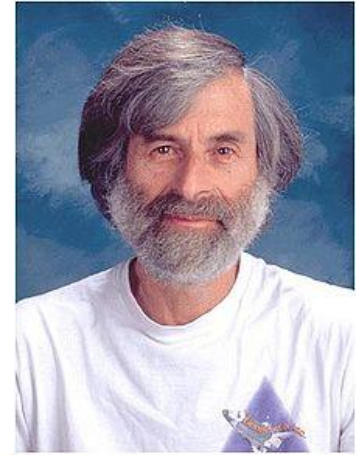
- Every process p_i
 - has initial value x_i chosen from some finite set V (often $V=\{0,1\}$)
 - shall irrevocably decide on output value y_i
- **Termination:** Every correct processor eventually decides
- **Agreement:** Every two correct processors p_i , p_j decide on the same value $y_i = y_j$
- **Validity:** If all correct processors have the same input value x , then x is the only possible decision value

Lamport, Shostak and Pease [LSP82]:

“There is a deterministic algorithm for solving consensus in a synchronous distributed system of $n \geq 3f+1$ processors in the presence of at most f Byzantine failures.”

But:

It is impossible to solve consensus if $n = 3f$!



Consensus Impossibility for $f=1, n=3$

- Suppose correct algorithm $A = (A, B, C)$ for (p_0, p_1, p_2) existed

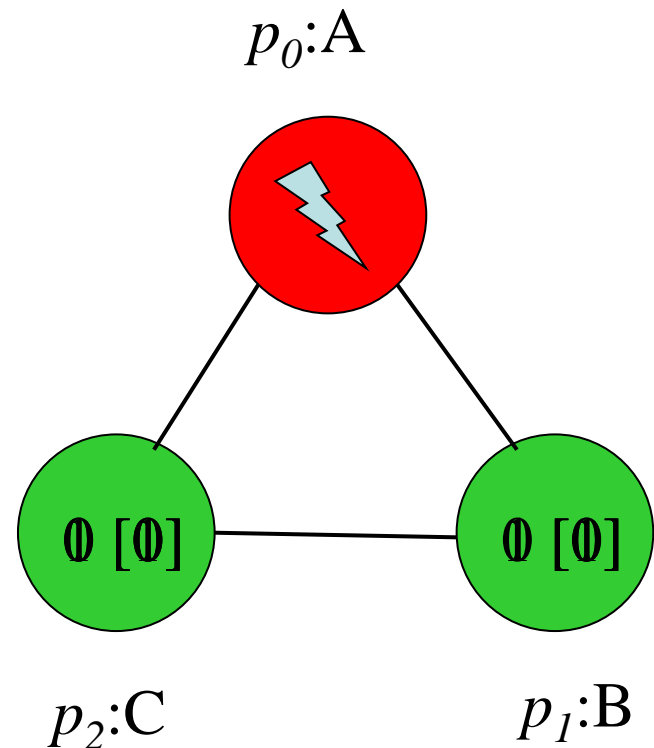
- Assume p_0 faulty

- By Validity:

- $x_1 = x_2 = 0 \rightarrow y_1 = y_2 = 0$
- $x_1 = x_2 = 1 \rightarrow y_1 = y_2 = 1$

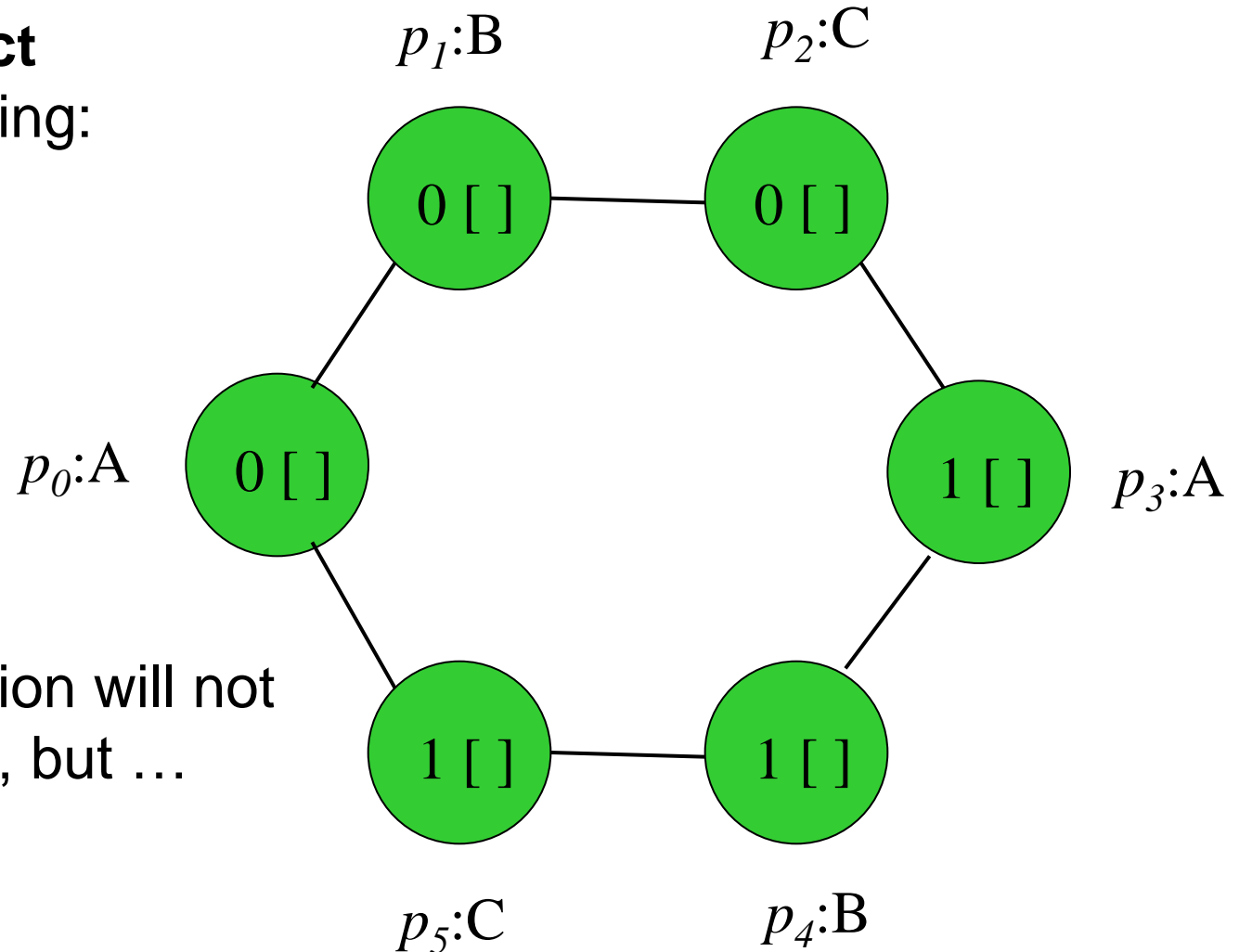
- By Agreement:

- $x_1 \neq x_2 \rightarrow y_1 = y_2$



„Easy Impossibility Proofs“ [FLM86] (I)

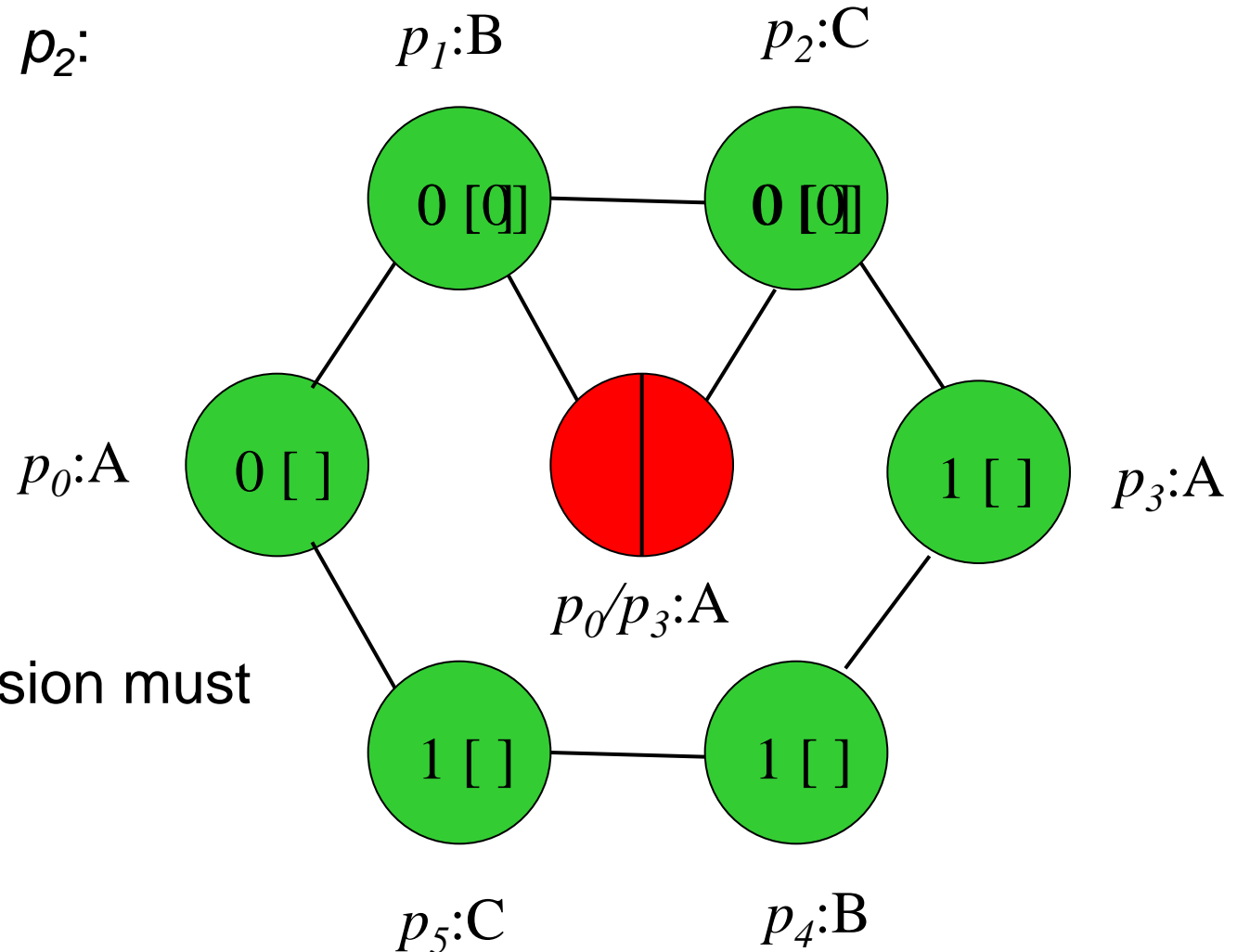
Arrange 6 **correct** processors in a ring:



Resulting execution will not solve consensus, but ...

„Easy Impossibility Proofs“ [FLM86] (II)

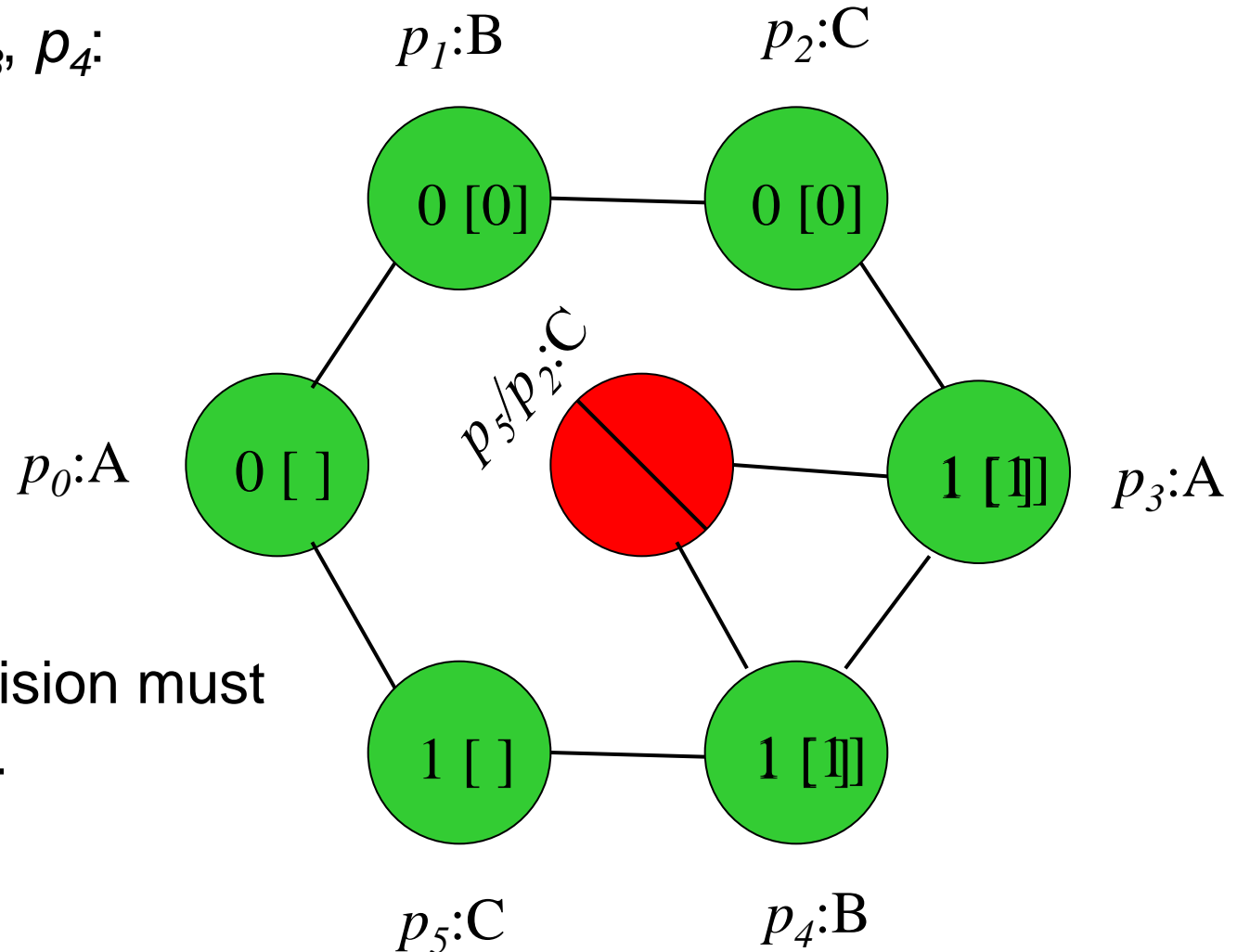
Local view of p_1, p_2 :



By Validity: Decision must be $y_1 = y_2 = 0 \dots$

„Easy Impossibility Proofs“ [FLM86] (III)

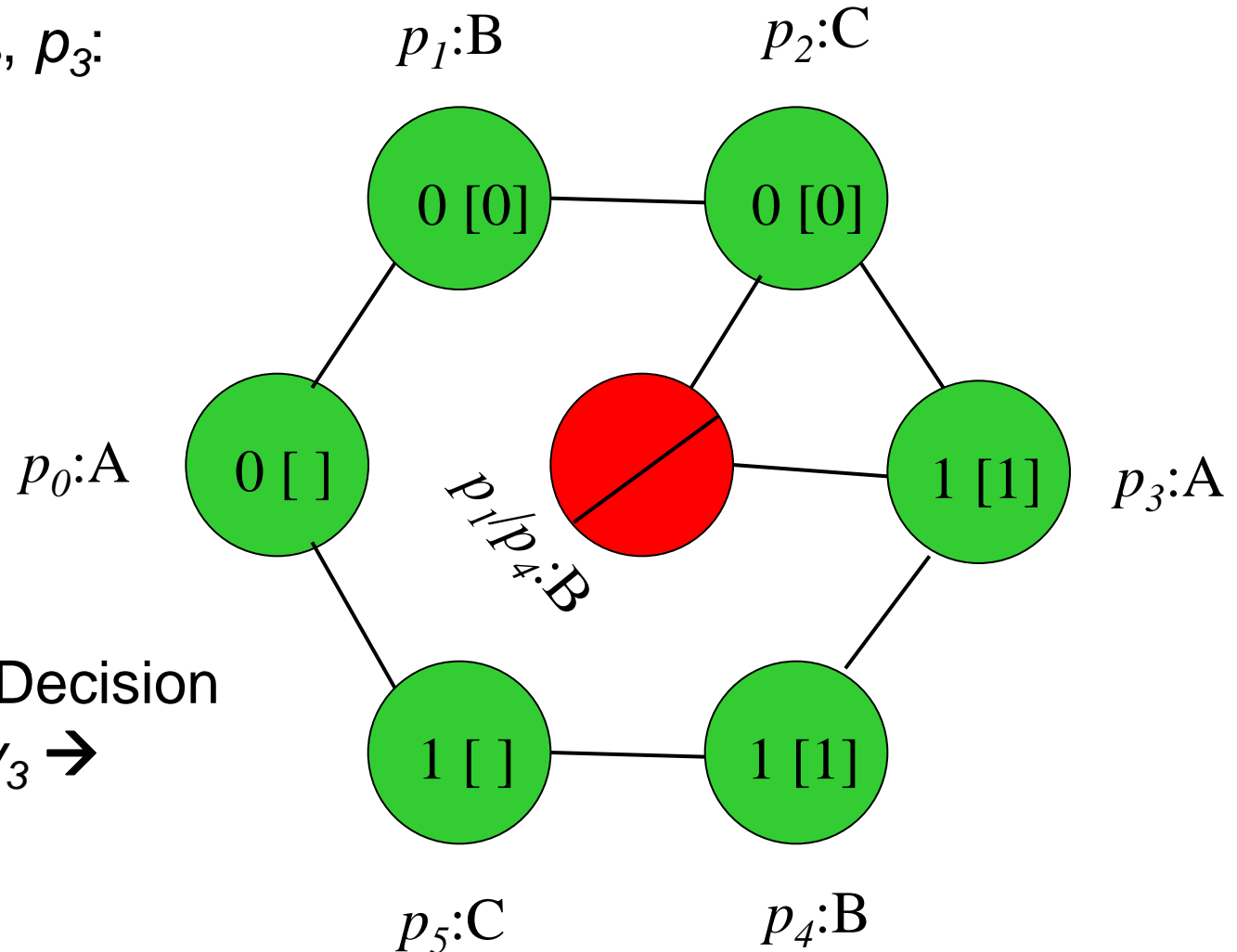
Local view of p_3, p_4 :



By Validity: Decision must be $y_3 = y_4 = 1 \dots$

„Easy Impossibility Proofs“ [FLM86] (IV)

Local view of p_2, p_3 :



By Agreement: Decision should be $y_2 = y_3 \rightarrow$
Contradiction

DARTS Correctness Proofs

Pipe Compare Signal Generators (PCSGs): There exists a dedicated detection circuit for each pair of pipes which generates the status signals $GEQ_{p,q}^{o/e}(t)$ and $GR_{p,q}^{o/e}(t)$. In particular, $GEQ_{p,q}^o(t')$ becomes active (i.e.,

$GEQ_{p,q}^o$ previous **Definition 4.1.** (Direct Causality). *Let $I(t')$ and $O(t)$ be two events of some specific signal input and output, respectively, of a correct component C . Then $I(t')$ and $O(t)$ are directly causally related, denoted by $I(t') \rightarrow O(t)$, if*

- (i) $r_{p,q}^{se}$
- (ii) $[r_{p,q}^r$

Similarl (i) $r_{p,q}^{self}(t) \in \mathbb{N}_{odd}$ and

- (i) $r_{p,q}^{self}(t) \in \mathbb{N}_{odd}$ and

Theorem 4.13. (Precision). *The precision $\pi \geq |b_q(t) - b_p(t)|$ of our algorithm is bounded by $\pi \leq \left\lfloor \frac{T_{sim}}{T_{first}^-} \right\rfloor + 1$.*

Proof. First of established for a $k + 1$, i.e., $t_k^p \leq$

$b^{max}(t')$

Assume that pro

Theorem 4.14. (Accuracy). *Given $\Delta = t_2 - t_1$, the accuracy $|b_p(t_2) - b_p(t_1)|$ of any correct process p is bounded by $\max \left\{ 0, \frac{\Delta - T_{sim} - T^+}{T^+} \right\} \leq |b_p(t_2) - b_p(t_1)| \leq \left\lceil \frac{\Delta}{T_{first}^-} \right\rceil + \min \left\{ \pi + 1, \left\lceil \frac{\Delta}{D^-} \right\rceil - \left\lceil \frac{\Delta}{T_{first}^-} \right\rceil \right\}$.*

Proof. The upper bound for accuracy will be shown first: It is known that $\forall t : b_p(t) \geq b^{max}(t) - \pi + (1 - I_{usync}(t))$ and $\forall t : b_p(t) \leq b^{max}(t)$ from Lemma 4.13 and Lemma 4.11. Thus $b_p(t_2) - b_p(t_1) \leq b^{max}(t_2) - b^{max}(t_1) + \pi - (1 - I_{usync}(t_1))$. By applying Lemma 4.11, $b_p(t_2) - b_p(t_1) \leq \left\lceil \frac{t_2 - t_1}{T_{first}^-} \right\rceil + 2I_{usync}(t_1) - 1 + \pi \leq \left\lceil \frac{t_2 - t_1}{T_{first}^-} \right\rceil + \pi + 1 \leq \left\lceil \frac{t_2 - t_1}{T_{first}^-} \right\rceil + \pi + 1$. Moreover, from Lemma 4.7 it follows that $b_p(t_2) - b_p(t_1) \leq \left\lceil \frac{t_2 - t_1}{D^-} \right\rceil$. Hence, $b_p(t_2) - b_p(t_1) \leq \min \left\{ \left\lceil \frac{\Delta}{T_{first}^-} \right\rceil + \pi + 1, \left\lceil \frac{\Delta}{D^-} \right\rceil \right\} \leq \left\lceil \frac{\Delta}{T_{first}^-} \right\rceil + \min \left\{ \pi + 1, \left\lceil \frac{\Delta}{D^-} \right\rceil - \left\lceil \frac{\Delta}{T_{first}^-} \right\rceil \right\} \leq \left\lceil \frac{\Delta}{T_{first}^-} \right\rceil + \min \left\{ \pi + 1, \left\lceil \frac{\Delta}{D^-} \right\rceil - \left\lceil \frac{\Delta}{T_{first}^-} \right\rceil \right\}$ since $[x + y] \leq [x] + [y]$.

To prove the lower bound, first define $b_1 = b_p(t_1)$, $b_2 = b_p(t_2)$ and $t_{b_1}^p \leq t_2$, $t_{b_2}^p \leq t_2$ as the points in time when p sends tick b_1 and b_2 . Clearly $t_{b_2+1}^p > t_2$,



Refined Analysis of Clock Frequency

- Consider simplified version of DARTS algorithm (for $f = 0$)
 - A node generates $tick(k+1)$ when all $tick(k)$ arrived
 - Clock frequency determined by **time series** of tick times $\forall k, p: x_p(k)$

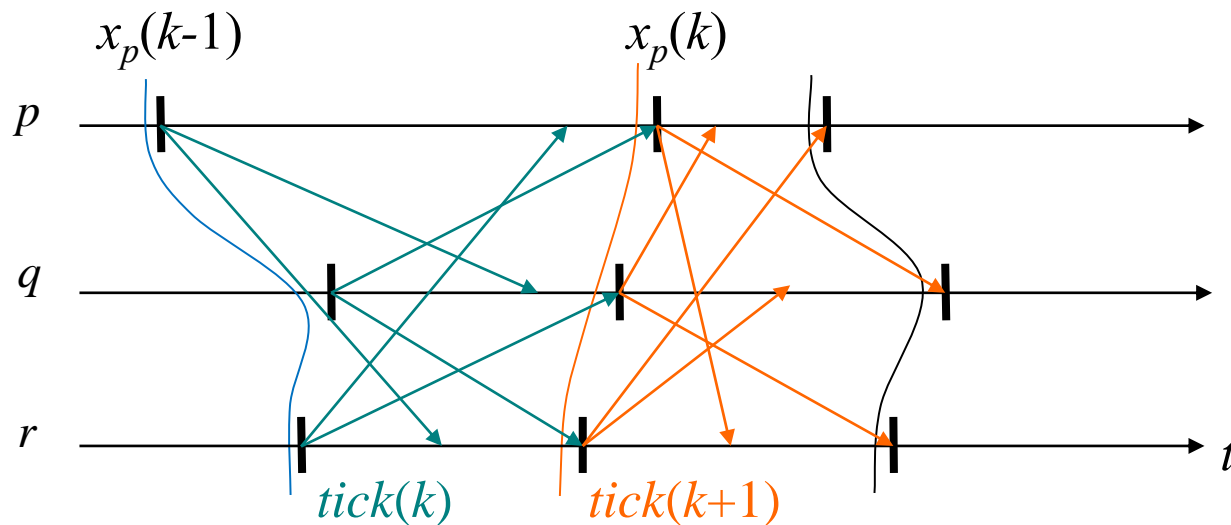
On init

→ send $tick(0)$ to all; $C := 0$;

If got $tick(C)$ from all nodes

→ send $tick(C+1)$ to all;

$C := C+1$;



On init

→ send $tick(0)$ to all; $C := 0$;

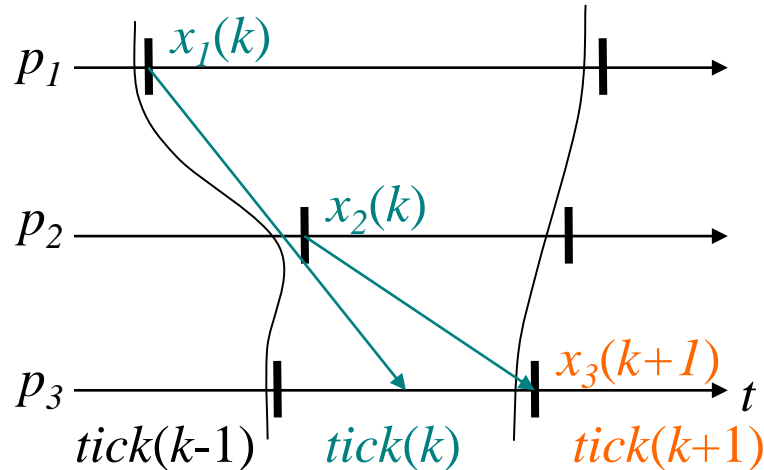
If got $tick(C)$ from all nodes

→ send $tick(C+1)$ to all;

$C := C+1$;

Setting

- $x_p(k)$ = time of sending $tick(k)$ by node p
- δ_{ij} = delay of message from node i to j



we observe:

- $x_3(k+1) = \max\{x_1(k) + \delta_{13}, x_2(k) + \delta_{23}\}$

Max,+ Algebra (II)

$$\mathcal{R}_{max} = (\mathbb{R}_{max}, \oplus, \otimes, \epsilon, e)$$

with

$$\epsilon := -\infty$$

$$e := 0$$

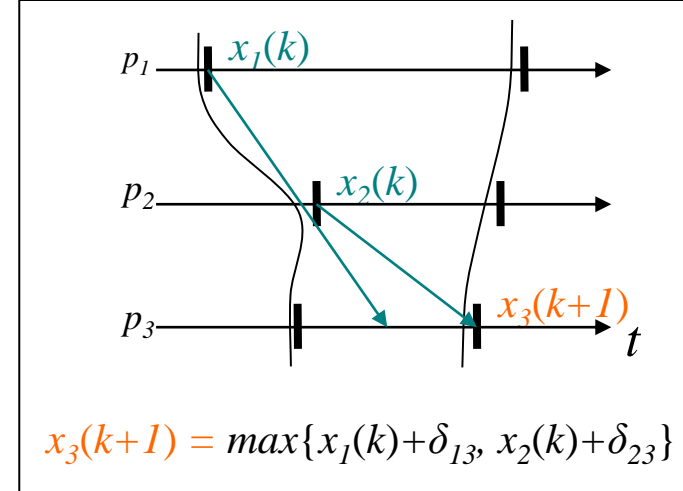
$$\mathbb{R}_{max} := \mathbb{R} \cup \{\epsilon\}$$

$$a \oplus b := \max(a, b); \quad a, b \in \mathbb{R}_{max}$$

$$a \otimes b := a + b; \quad a, b \in \mathbb{R}_{max}$$

yields

$$x_3(k+1) = (x_1(k) \otimes \delta_{13}) \oplus (x_2(k) \otimes \delta_{23})$$



Max,+ Algebra (III)

- Transmission delay matrix

Tick times vector

$$\Delta = \begin{pmatrix} \epsilon & \delta_{12} & \dots & \delta_{1n} \\ \delta_{21} & \epsilon & & \delta_{2n} \\ \vdots & & \ddots & \vdots \\ \delta_{n1} & \delta_{n2} & \dots & \epsilon \end{pmatrix}$$

$$\vec{X}(k) = \begin{pmatrix} x_1(k) \\ x_2(k) \\ \vdots \\ x_n(k) \end{pmatrix}$$

- Max,+ matrix multiplication yields

$$\vec{X}(k) = (\Delta^T)^{\otimes k} \otimes \vec{X}(0)$$

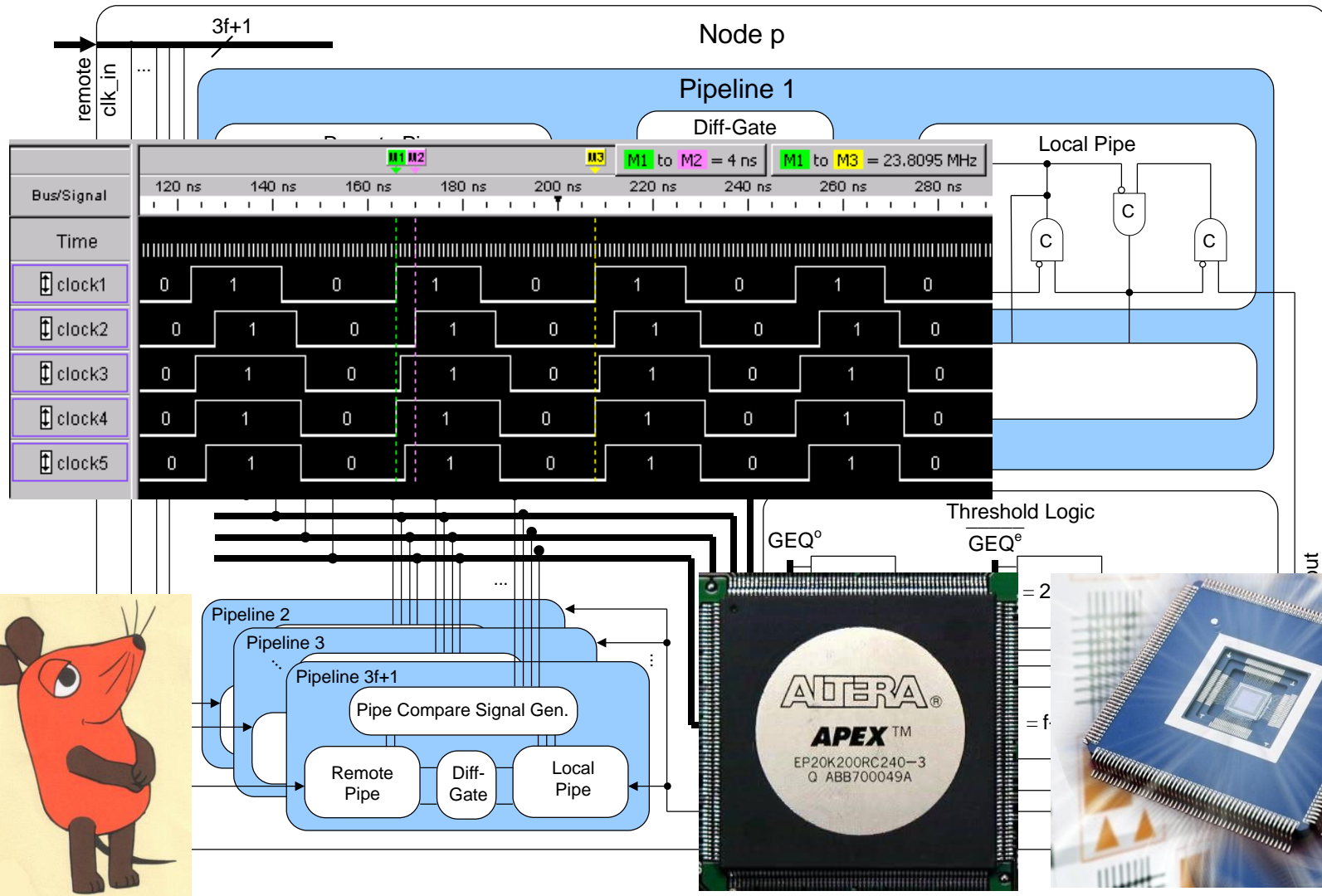
- Can utilize full power of linear algebra (eigenvalue analysis etc.)

„Exotische“ Mathematik-Kenntnisse
in TI-Forschung offenbar nützlich ...

TI Research = Theorie?

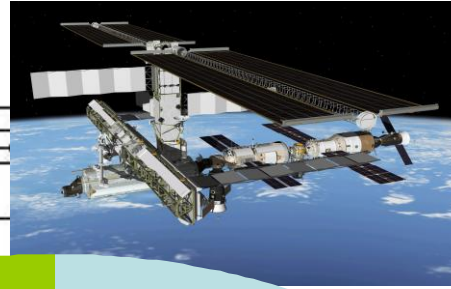
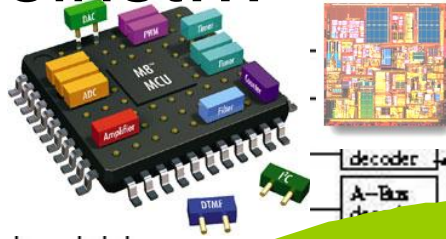
- ✓ Formal-mathematische Komponente wichtig, aber keineswegs hinreichend:
- ✓ Nicht alle Probleme analytisch lösbar
- ✓ Modell-Generierung & Validation erfordert jedenfalls Experimente
- ✓ Demonstration der technologischen Realisierbarkeit einer Lösung erfordert jedenfalls Prototyp-Entwicklung (HW, SW)
- **Engineering ebenfalls wichtig!**

DARTS Implementation



- ✓ Warum bin ich hier?)
- ✓ Versuch eines Einblicks in die TI-Forschung:
DARTS
 - Fault-tolerant distributed algorithms
 - Performance analysis using max,+ algebra
- Technische Informatik an der TU Wien
- Probleme
- Lösungsideen

Anwendungen Technische Informatik einst...



Systems-on Chip

Networked Embedded Systems

HW/SW Codesign

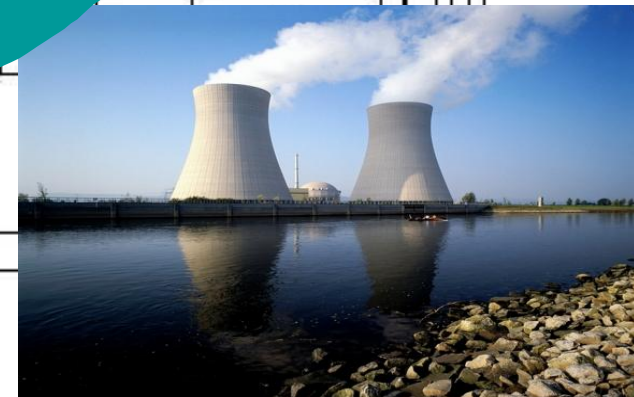
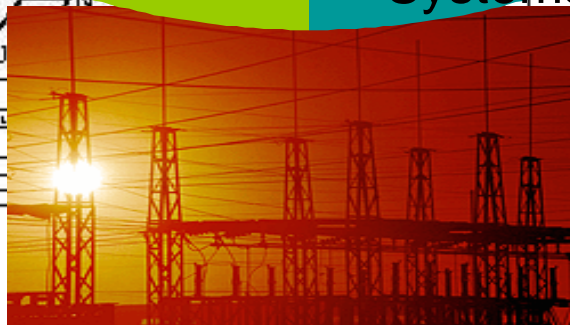
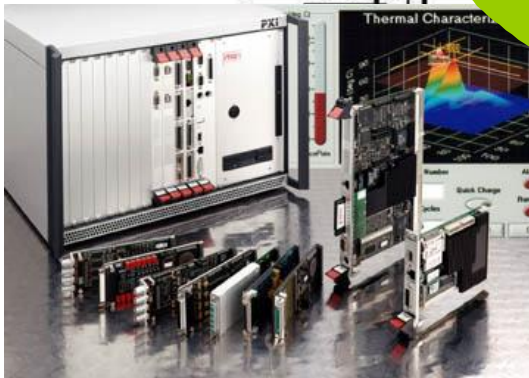
und jetzt

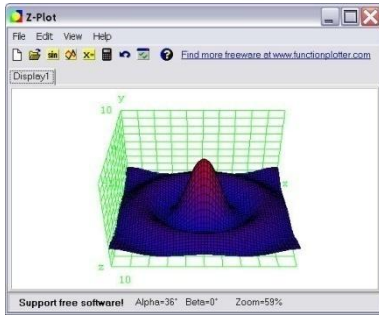
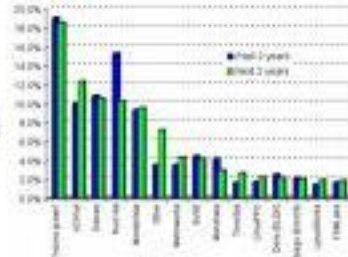
Real-Time Systems



System Architectures

Fault-Tolerant Systems



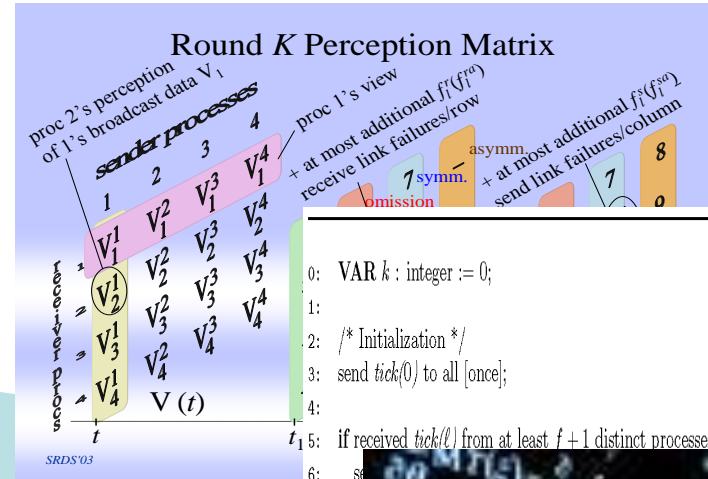


Experimentell

Theoretisch

Technologisch

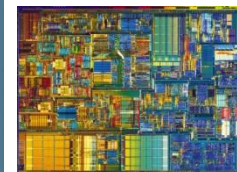
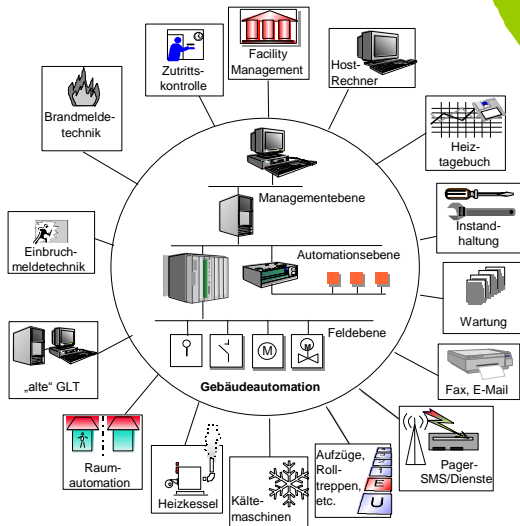
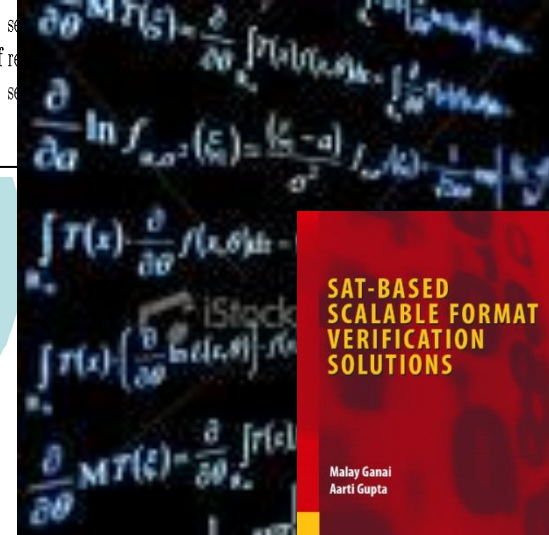
TI



```

0: VAR k : integer := 0;
1:
2: /* Initialization */
3: send tick(0) to all [once];
4:
5: if received tick(l) from at least f+1 distinct processes with l > k
6:   send tick(k+1) to all;
7: if received tick(k+1) from at least f+1 distinct processes with l > k+1
8:   send tick(k+2) to all;

```



Interdisziplinarität:

- Informatik
- Mathematik (auch kontinuierlich!)
- Elektrotechnik
- Physikalisch/technisches Anwendungswissen

Integrative Sichtweise:

- Hardware/Software
- Signalverarbeitung
- Kommunikation
- Fehlertoleranz
- Echtzeitverarbeitung
- Security
- Energie/Ressourcen-Effizienz

Wissenschaft:

- Grundlagenwissen („Wissen warum“)
- Formal-mathematische Fertigkeiten (Beweise!)
- Abstraktionsvermögen (Modellbildung)
- Schreib- und Präsentationsfertigkeiten
- Neugierde

Engineering:

- Anwendungswissen („Wissen wie“)
- Entwicklungsfertigkeiten (Programmierung)
- Umgang mit Tools & Meßgeräten
- Beherrschung HW/SW-Schnittstelle
- „Schöpferdrang“



TI Studien

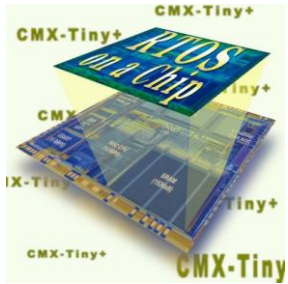


✓ **Bachelor Technische Informatik (BTI, 535)**

- Studierendenzahlen (typisch-max)
 - 60-90 Anfänger
 - 40-60 im 3. Semester
 - 40-50 Absolventen
- Importiert LVAs aus der Fakultät Elektrotechnik
- LVAs 1. Jahr meist gemeinsam mit anderen Informatik-Bachelors

✓ **Master Technische Informatik (MTI, 938)**

- Typische Studierendenzahlen
 - 30 im 2. Semester
 - 20 Absolventen
- Individuelle Wahlmöglichkeiten



- Gehobene Entwicklungsaufgaben im Bereich Embedded Systems
 - VLSI Design
 - Microcontroller/DSP Software-Entwicklung
 - Netzwerkprogrammierung
- Application Engineering
- Unterstützende Aufgaben im Forschungsumfeld

Übersicht Bachelor TI

1.	Programm- konstruktion (8.8 ECTS)	Grundlagen Digitaler Systeme (6 ECTS)	Analysis (6 ECTS)	Algebra und Diskrete Math. (9 ECTS)	STEG (0.2 ECTS)
2.	Algorithmen und Datenstrukuren (9)	Elektrotechnische Grundlagen (7.5)	Analysis II (7.5)	Theoretische Inf. und Logik (6)	
3.	Rechnerstrukt. u. Betriebssysteme (9)	Modellbildung in der Physik (6)	Signale und Systeme (4.5 + 4.0)		Wahrscheinlich- keitstheorie u. Stoch. Proz. (7.5)
4.	Microcontroller u. Betriebssysteme (10)	Digital Design (3 + 9)		Programm- und Systemverifikation (6)	Zuverlässige Echtzeitsysteme (6)
5.	Dezentrale Automation (6)		Regelungstechnik (6)	Wahlmodul (6)	Einf. Wiss.Arb. (3) +
6.				Wahlmodul (6)	Bachelor-Arbeit (10)

Programming

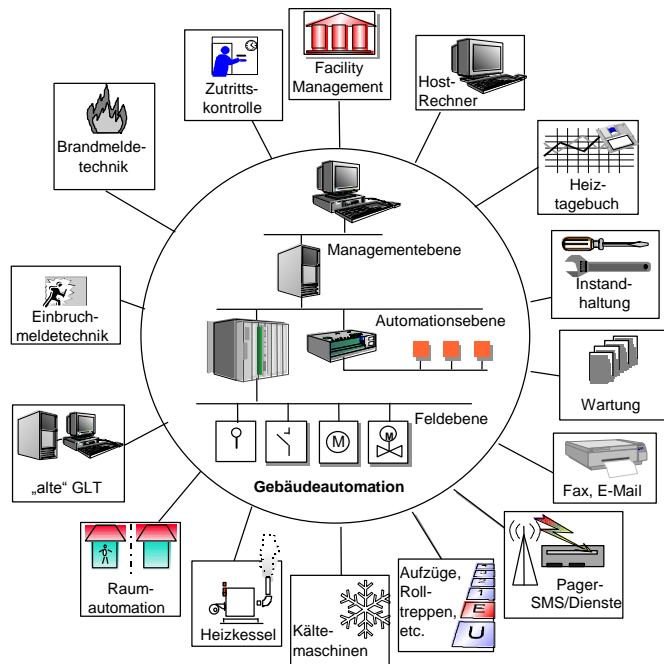
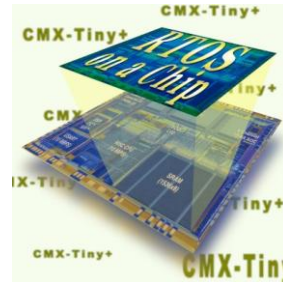
Hardware

*Signale &
Systeme*

Dependable Systems

Checklist „Bachelor TI 4 Me“ ?

- ✓ Interesse an Elektrotechnik und technischen Anwendungen ?
- ✓ Interesse an Mathematik (inklusive kontinuierlicher) ?
- ✓ Ganzheitliche Sichtweise ?
- ✓ Abstraktionsvermögen ?
- ✓ Neigung zur Tüftelei ?
- ✓ Schriftliche und mündliche Kommunikationsfähigkeit, v.a. mit Vertretern anderer Disziplinen ?
- ✓ Teamfähigkeit ?



- Wissenschaftliche Forschung
- Industrielle Forschung
- Design und Analyse komplexer eingebetteter Systeme

Schlüsselbereiche: Dig. Design Sig.& Systeme Dep.Systems Verifikation

1.

Discrete
Mathematics
(9 ECTS)

HW/SW Codesign
(6 ECTS)

Stochastische
Signalverarbeitung
(4.5 ECTS)

Formal Methods
in Computer
Science (6 ECTS)

2.

Advanced
Programming
(10.5)

Designing
Technosocial
Systems (6)

Verteilte
Algorithmen (6)

Computer-Aided
Verification (6)

3.

Wahlmodule (9)

Embedded Systems Engineering (6 + 6)

4.

Wahlmodule (6)

Diplomarbeit (30)

Studienzulassung für Master TI [Auflagen]

Für jedes angeführte Modul müssen die angegebenen ECTS an thematisch eng verwandten und vom Level her entsprechenden LVAs bereits absolviert worden sein. Ist das nicht der Fall, muß das angeführte Modul (zur Gänze) als Auflage nachgeholt werden.

■ Einstieg aus Inf/ET/Math/Physik Bachelor („generisch“)

- 9 Algebra und Diskrete Mathematik
- 6 Analysis
- 6 Wahrscheinlichkeitstheorie und Stochastische Prozesse
- 9 Algorithmen und Datenstrukturen
- 6 Theoretische Informatik und Logik
- 7,5 Elektrotechnische Grundlagen
- 9 Rechnerstrukturen u. Betriebssysteme
oder 10 Microcontroller u. Betriebssysteme
- 12 Digital Design
- 8,5 Signale und Systeme
- 6 Zuverlässige Echtzeitsysteme

■ Einstieg aus BSI

- 7,5 Elektrotechnische Grundlagen
- 12 Digital Design
- 8,5 Signale und Systeme
- 6 Zuverlässige Echtzeitsysteme

■ Einstieg aus BMI, BZI

- 7,5 Elektrotechnische Grundlagen
- 6 Betriebssysteme
- 12 Digital Design
- 8,5 Signale und Systeme
- 6 Zuverlässige Echtzeitsysteme

■ Einstieg aus Bachelor ETI

- 9 Algorithmen und Datenstrukturen
- 6 Theoretische Informatik und Logik
- 9 Rechnerstrukturen und Betriebssysteme
- 6 Zuverlässige Echtzeitsysteme

Checklist „Masterstudium TI 4 Me ?“

- ✓ Bachelor-level Grundlagenkenntnisse in Informatik und Elektrotechnik (Ideal: Bachelor TI) ?
- ✓ Interesse an wissenschaftlicher Forschung ?
- ✓ Formal-mathematische Neigung ?
- ✓ Interdisziplinäres Interesse ?
- ✓ Holistische Sicht der Dinge ?
- ✓ Writing Skills ?
- ✓ Selbstorganisation ?

- TI-Portal: *<http://ti.tuwien.ac.at>*
- Fakultät: *<http://www.informatik.tuwien.ac.at>*
- Studienpläne: *<http://www.informatik.tuwien.ac.at/lehre/studien>*

- ✓ Warum bin ich hier?)
- ✓ Versuch eines Einblicks in die TI-Forschung:
DARTS
 - Fault-tolerant distributed algorithms
 - Performance analysis using max,+ algebra
- Technische Informatik an der TU Wien
- Probleme
- Lösungsideen

Warum bin ich hier ?

Ein Ingenieur, ein Physiker und ein Mathematiker übernachten in einem Hotel ...

■ Probleme der TU Wien - Informatik:

- 💣 Generell: Wissenschaft vs. Engineering
- 💣 Falsche Vorstellungen von Zielen und Anforderungen eines akademischen Informatik-Studiums
- 💣 Fehlende Vorkenntnisse und Fertigkeiten der Studienanfänger

In meiner Funktion
als Koordinator der TI
- Studien an der TU
Wien

Wissenschaft vs. Engineering?

- Umgang mit unterschiedlichen Anforderungen an unsere Absolventen:
- Strategie:
 - Bachelor-Studien:
 - Wesentlich Grundlagen-/Wissenschaftslastiger als FHs
 - Engineering-Grundausbildung für die industrielle Praxis **von heute**
 - Orientiert sich daher auch am derzeitigen Bedarf der Industrie
 - Master-Studien:
 - Schaffung der Grundlagen für die industrielle Praxis **von morgen**
 - Vorbereitung für wissenschaftliche Karriere
 - Orientiert sich wenig am derzeitigen Bedarf der Industrie
- Hauptproblem: Umgang mit unterschiedlicher Vorbildung HTL/AHS-Schüler

Fehlende Vorkenntnisse und Fertigkeiten

Typische HTL-Absolventen:

- + **Gute Engineering-Grundausbildung (Elektrotechnik, Programmieren)**
- + Motivation und Arbeitsdisziplin
- Mangelnde Wertschätzung formal-mathematischer Inhalte
- Mangelndes Abstraktionsvermögen
- Falsche Primärerwartung: Vertiefte Engineering-Ausbildung („more of the same“)
- Mangelnde Deutsch und Englischkenntnisse

Typische AHS-Absolventen:

- + Offenheit für Theorie
- + Gute Deutsch und Englischkenntnisse
- Unkenntnis der Studieninhalte
- **Keinerlei Engineering-Grundausbildung**
- Mangelnde Motivation und Zielstrebigkeit

- TU-Informatik wird von Studienanfängern gestürmt (> 1000 pro Jahr in unseren 5 Bachelor-Studien)
- Seit 2011 verpflichtende „Studieneingangsgespräche“ vor Beginn des 1. Semesters
- Seit 2012 Beschränkung durch frühe Leistungsüberprüfung (max. 412 Studierende pro Jahr in jeder Lehrveranstaltung des 1. Semesters)
- Erkenntnisse daraus:
 - Vorbereitung geeigneter Schüler auf TU-Studium wäre sinnvoll
 - Angemessene Anrechnung der Engineering-Grundausbildung aus einer HTL würde
 - + Langeweile von HTL-Absolventen in Grundlagen-LVAs vermeiden
 - + es uns erlauben, uns auf unsere eigentliche (= Wissenschafts-) Kompetenz zu beschränken
 - + unsere Ressourcen entlasten

Warum bin ich hier ?

- Naheliegende Idee: **Abhilfe durch Verbesserung des „Interfaces“ zu den Schulen!**
- Leichter gesagt als getan: Erfordert
 - Herstellung einer „Win-Win“ Situation!
 - Ressourcensparende Realisierung!
 - Zeit ...
 - Ideen ...
 - Initiative ...

In meiner Funktion
als Koordinator der TI
- Studien an der TU
Wien

- ✓ Warum bin ich hier?)
- ✓ Versuch eines Einblicks in die TI-Forschung:
DARTS
 - Fault-tolerant distributed algorithms
 - Performance analysis using max,+ algebra
- Technische Informatik an der TU Wien
- Probleme
- Lösungsideen

Warum bin ich hier ?

Aus
persönlichen
Gründen

Für mich:

HTL St. Pölten

≡

Extrem vielfältiges Angebot, Wissen
und Fertigkeiten zu erwerben

Einige unausgegeborene Ideen ...

Möglichkeiten:

- + Angemessene Anerkennung der HTL Engineering-Grundausbildung
 - + Programmierung
 - + Microcontroller
 - + Elektrotechnik (Labor)
- + Förderung/Vorbereitung besonders begabter Schüler
 - + Mathematik
 - + Deutsch/Englisch
- + Kooperationen via Schülerprojekte
 - + „Auslagerung“ von Prototyp-Entwicklungen an HTL
 - + „Auslagerung“ wissenschaftlicher Betreuung an TU

Schwierigkeiten:

- Konkrete Realisierung ?
 - 1:1 Anrechnung: Qualitätssicherung ?
 - „Easy path“ ?
 - Abstimmung der Lehrinhalte !
- Konkrete Realisierung ?
 - Aufwertung dieser Fächer im Regelunterricht ?
 - Vertiefungsfächer ?
- Ressourcenfrage ?



© 2007, WDR

Vielen Dank für Ihre Aufmerksamkeit !