

# Modeling a GPS Receiver Using SystemC

**Bernhard Niemann  
Reiner Büttner  
Martin Speitel**

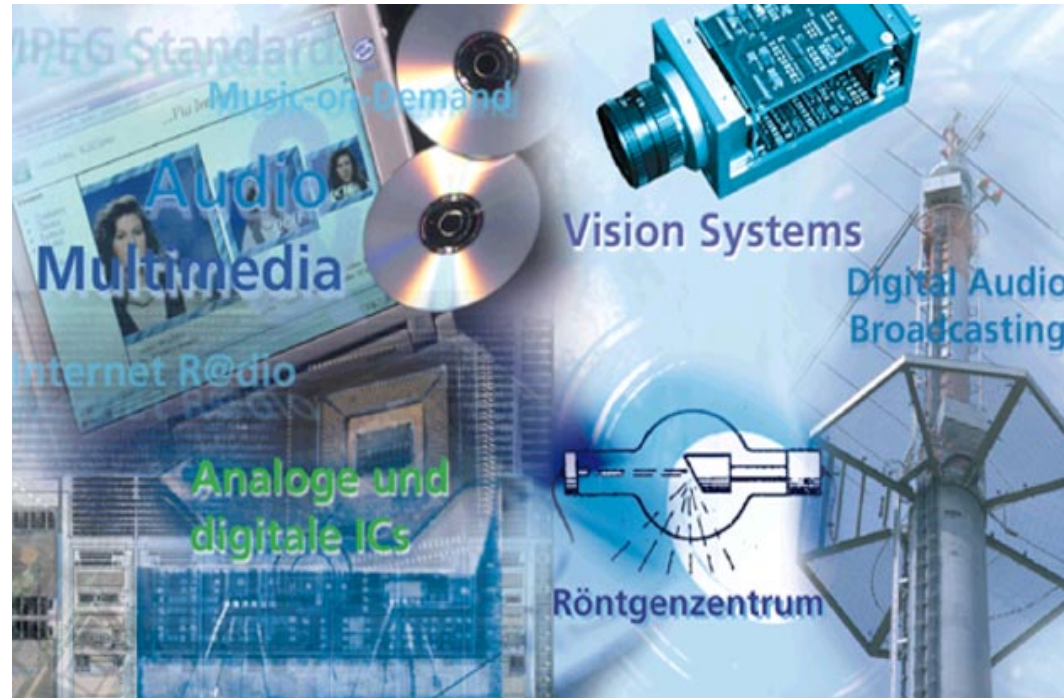
**<http://www.iis.fhg.de>  
<http://www.iis.fhg.de/kursbuch/kurse/systemc.html>**



# The Fraunhofer Institute for Integrated Circuits

**Audio & Multimedia**  
**Image Processing**  
**Digital Broadcasting**  
**Wireless Networks**  
**Net Access Technology**

**ASICs**  
**Vision Systems**  
**X-Ray Technology**  
**Security Systems**  
**Training and Support**  
**Transport Logistics**



# Introduction

---

- **Introduction**
- **GPS Basics**
- **System Overview**
- **The Model**
- **Simulation Results**
- **Conclusions**

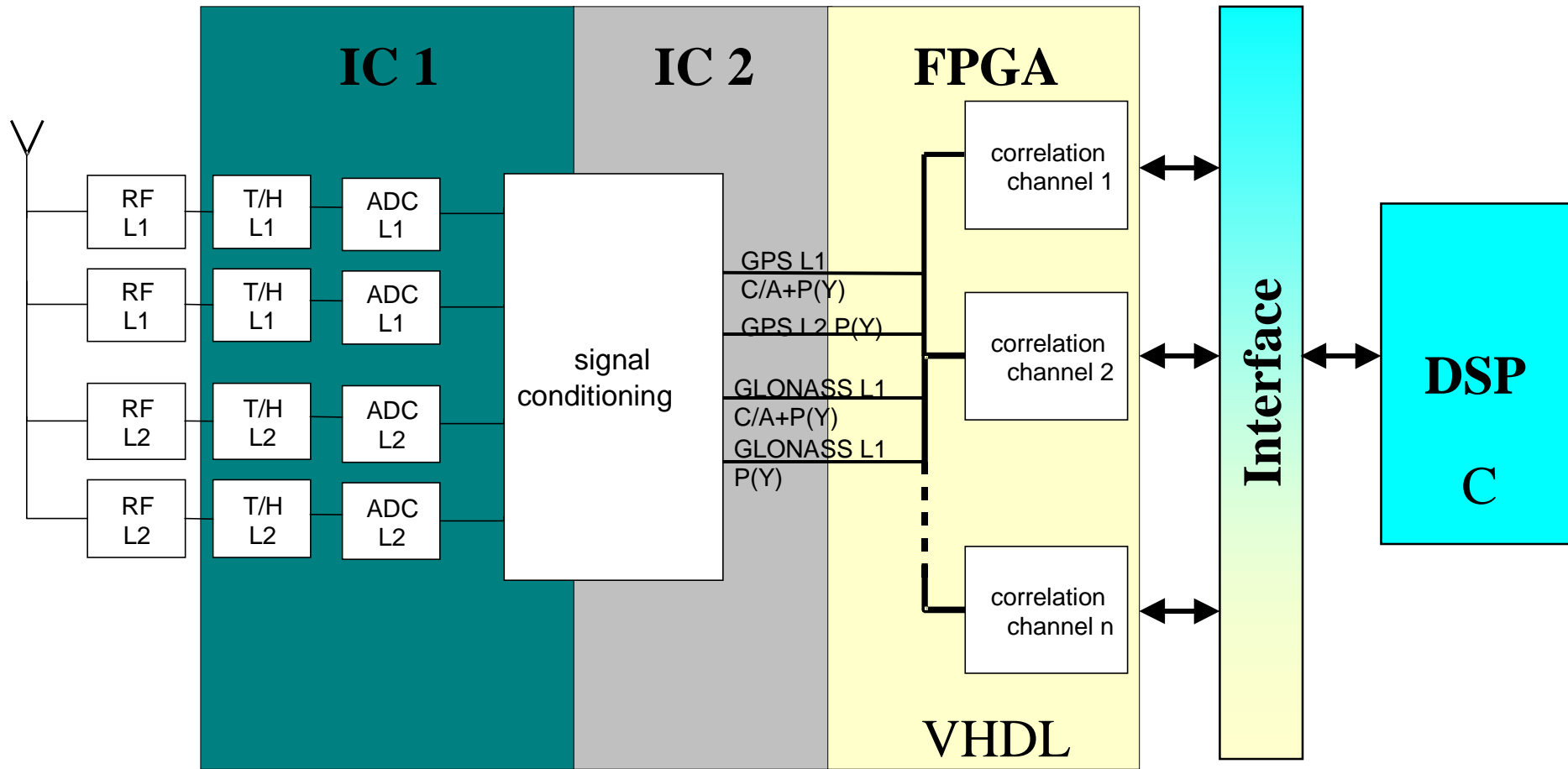


# GPS - A Brief Introduction

---

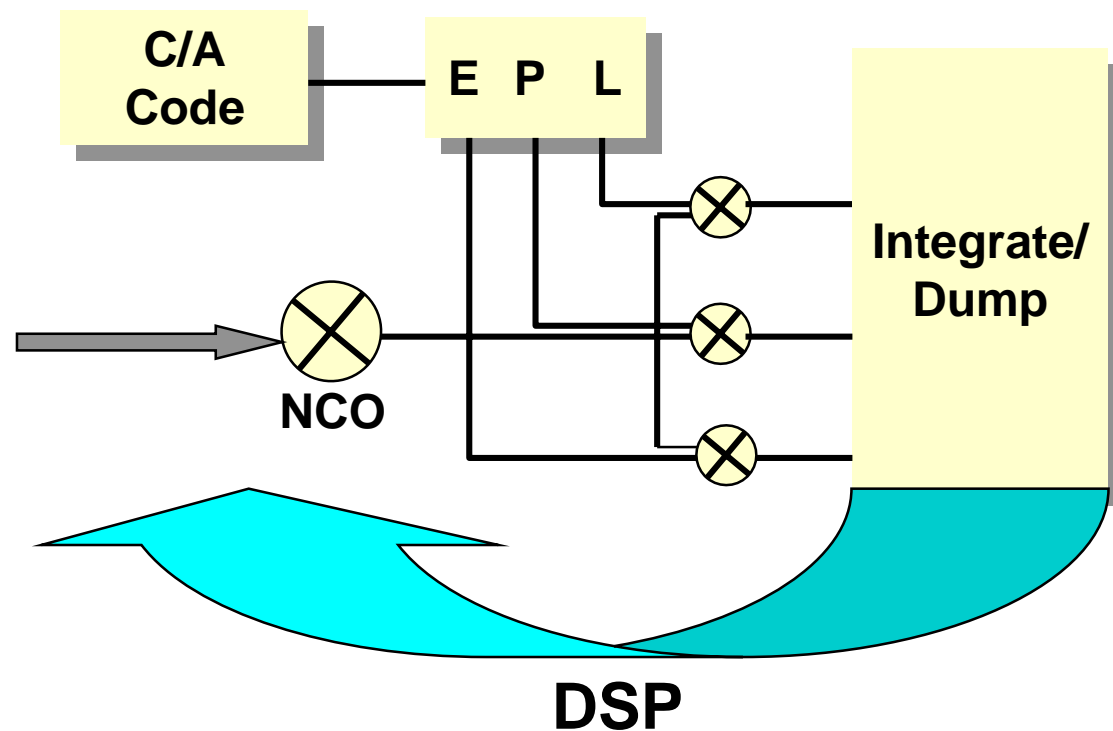
- **GPS receivers measure the transmission time of signals sent by at least three satellites to determine the position**
- **For code phase detection a pseudo random noise (PRN) code is used**
  - **Navigation data is modulated on this code**
  - **There are two different PRN codes in use**
    - **C/A (Coarse Acquisition) - 1023 chips in length - 1.023 MCps**
    - **P(Y) (Precise) for military usage**
- **The GPS receiver has to strip off the C/A code in order to regenerate the navigation data**
  - **Use the properties of PRN sequences**
  - **Search for a peak in the auto correlation function**

# System - Architecture



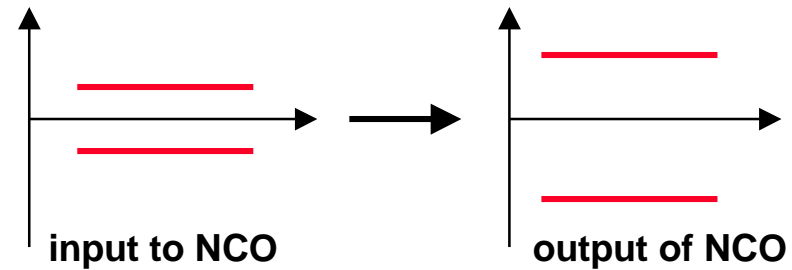
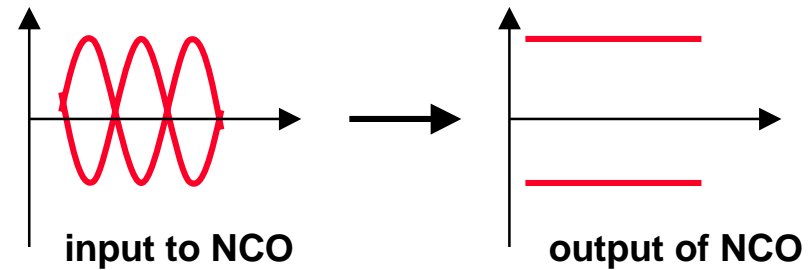
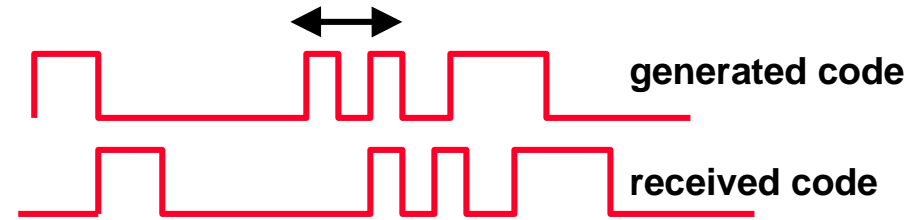
# The Correlator - Architecture

- generate C/A code
- delay the generated code within the E(arly), P(resent), L(ate) shift register
- compare the incoming code with the delayed versions and integrate for one C/A code cycle (1023 chips)
- send the integrated values to the DSP
- the DSP calculates new input values for NCO and code generator



# The DSP code

- **Code tracking**
  - **synchronize code**
  - **shift generated code**
- **Frequency correction**
  - **strip off remaining carrier**
  - **set increment for NCO**
- **Phase correction**
  - **compensate phase offset**
  - **set offset for NCO**



© FhG IIS-A, 2000

Page 7

# Why a SystemC Model of an Existing System?

---

- **Exploration**
  - **different loop algorithms**
  - **different HW/SW partitioning possibilities**
  - **new architectural ideas**
  
- **Experience**
  - **comparison VHDL/SystemC**
  - **modeling with SystemC**
  
- **Speed**
  - **fast turn around times for new DSP code**
  - **more flexibility for input generation**
  
- **Testing**
  - **strobe what ever you want**
  - **generate different output formats without effort**
  - **have an executable spec for implementing features**

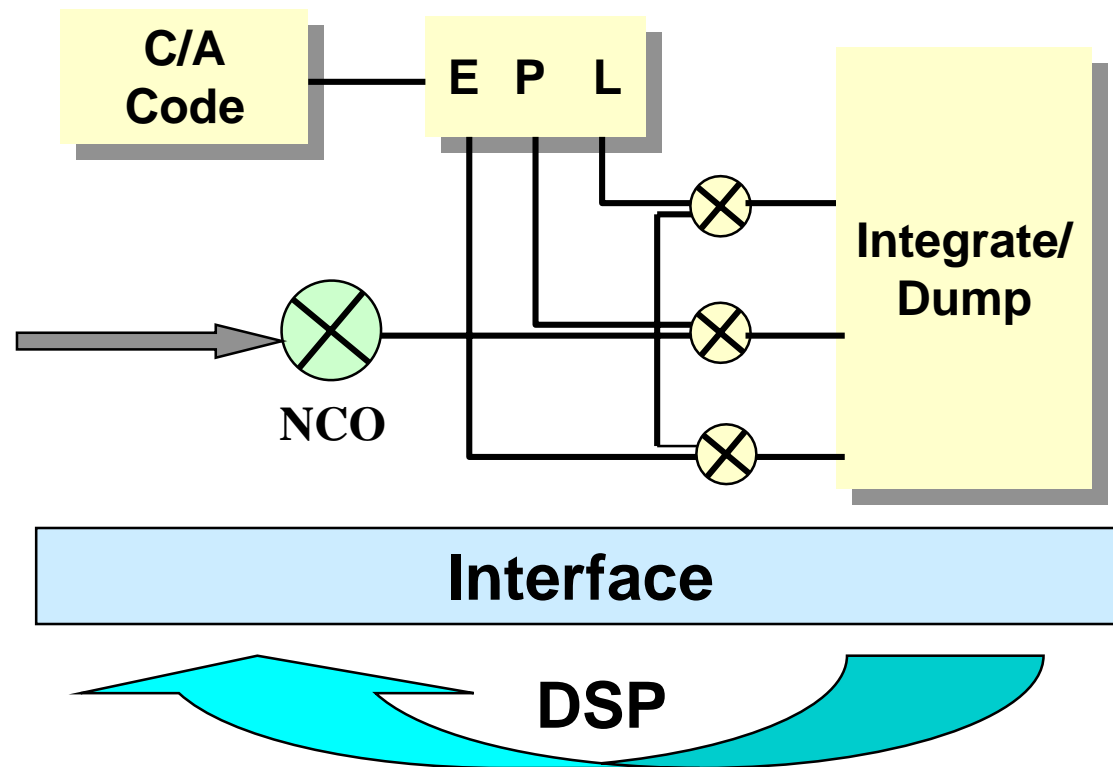
# The Approach - Levels of Hierarchy

- VHDL RTL -> SystemC RTL
  - std\_logic -> bool
  - std\_logic\_vector -> int

- VHDL RTL -> SystemC Beh.

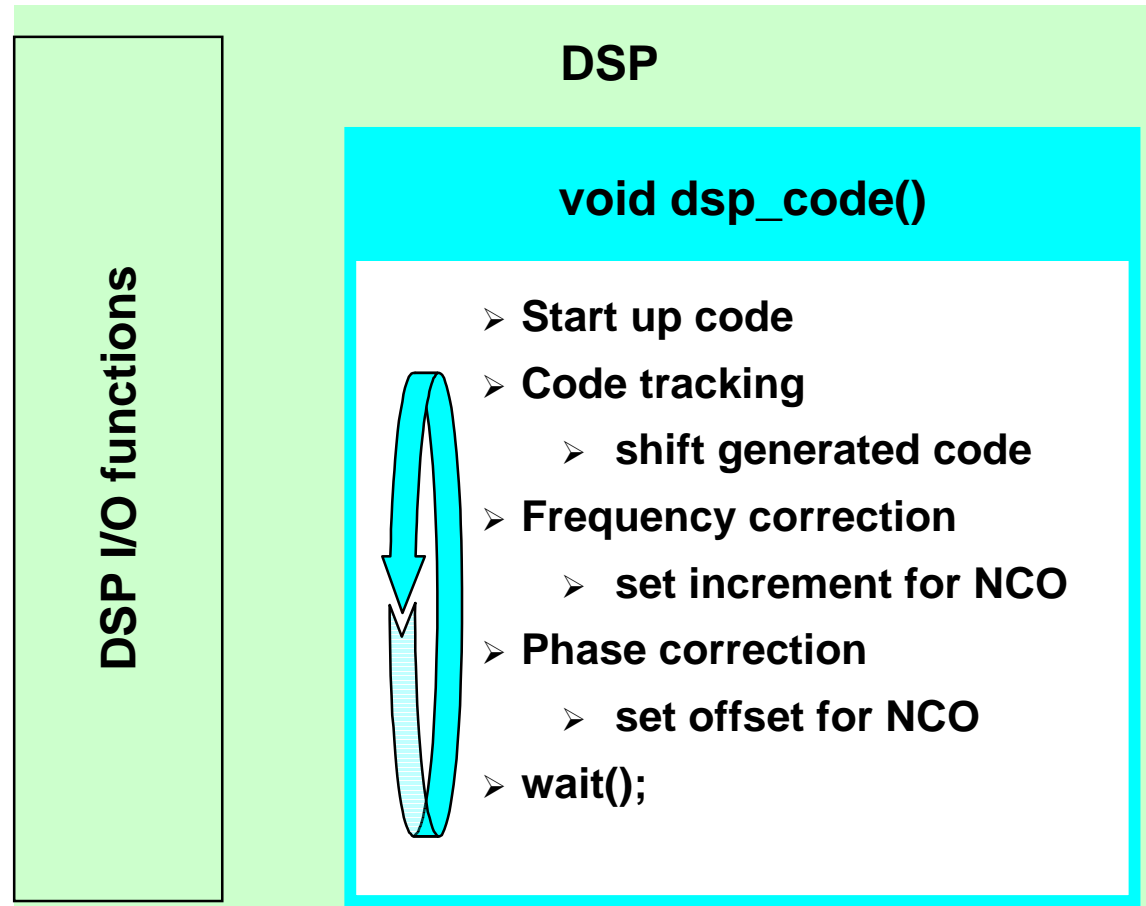
- Event driven modeling

- DSP code integration



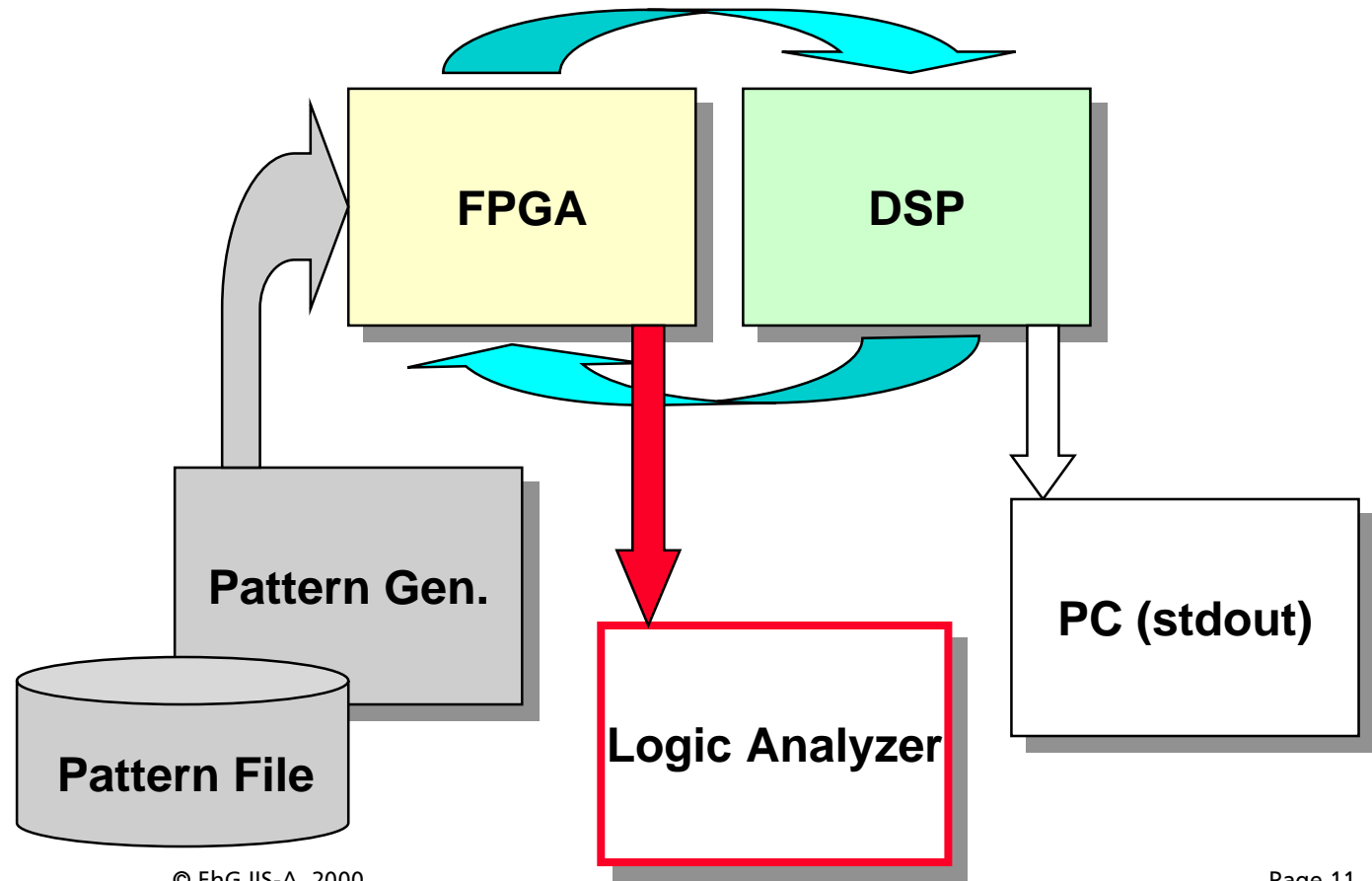
# The Approach - Loop Integration

- minimal modeling effort
- minimal integration effort for new code
- maximum simulation speed



# The Complete System

- FPGA
- DSP
- Pattern Generator
- Logic Analyzer
- PC

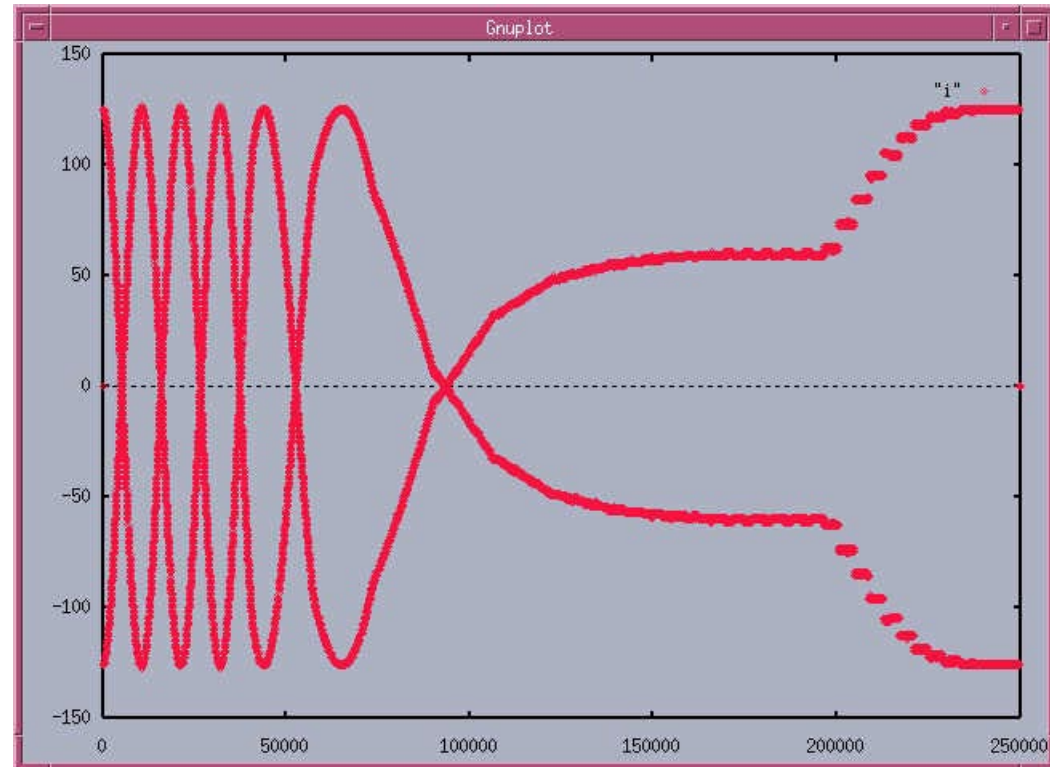


© FhG IIS-A, 2000

Page 11

# Simulation Results

- Input is C/A code data with code offset, frequency modulation (200Hz) and phase offset
- After correlation has been reached, the frequency correction starts
- The last step is to correct the phase
- 8bit representation is used for C/A code data, so maximum values are  $\pm 127$



# Conclusions

---

## ➤ Pros

- Easy to bring existing HW and SW modules into SystemC
- HW/SW Codesign without any commercial tools
- Easy debugging
- Easy design exploration

## ➤ Cons

- Until now no converters (V)HDL <-> SystemC
- Some problems with SystemC data types

## ➤ Project will go on

