

Proceeding of the
Seventh IEEE, IET International Symposium on
**Communication Systems, Networks
and
Digital Signal Processing**

List of Papers

		Page Number
ACSP:	Fast Transform and Algorithms for Communications and Signal Processing	
ACSP-1:	Is it a Specific Vocal Tract Shape in Order to Pronounce Emphatics?	1
ACSP-2:	Coefficients and generating functions of sinc ^N FIR filters	5
ACSP-3:	A Novel Adaptive LMS-based Algorithm Considering Relative Velocity of Source	10
ACSP-4:	Implementation Alternatives of Space Block-Coded OFDM	15
ACSP-5:	Simple DSP-IDFT Techniques for Generating Spectrally Efficient FDM Signals	20
ACSP-6:	Feature Selection using an Ensemble of Optimal Wavelet Packet and Learning Machine: Application to MER Signals	25
ACSP-7:	Parallel Architecture of an All Digital Timing Recovery Scheme for High Speed Receivers	31
ACSP-8:	Fast and Flexible Pipelined Multi-processor Architecture for Multimedia Device	35
ACSP-9:	Fuzzy Future Demand Uncertainty Management in Optical Networks	40
ACSP-10:	Frame Synchronization with Large Carrier Frequency Offsets: Point Estimation versus Hypothesis Testing	45
AHN:	Ad-hoc Networks	
AHN-1:	A Soft Admission Control Methodology for Wireless Ad-Hoc Networks: Evaluating the Impact on Existing Flows before Admission	51
AHN-2:	Graphical System-level Simulations for Wireless Sensor Networks Design Space Exploration at Hardware and Software Levels	56
AHN-3:	Impact of node density and mobility on the performance of AODV and DSR in MANETS	61
AHN-4:	Implementation of Microscopic Parameters for Density Estimation of Heterogeneous Traffic Flow for VANET	66
AHN-5:	Node failures on MANET Routing with Multimedia traffic	71
AHN-6:	Estimating resources in Wireless Ad-Hoc Networks: a Kalman Filter approach	77
ATST:	Antenna Technology for Satellites and Terrestrial Wireless Systems	
ATST-1:	Resource allocation algorithms applying multiuser spatial multiplexing in multiple antenna systems	82
ATST-2:	Measurement on Simple Vehicle Antenna System Using a Geostationary Satellite in Japan	87
ATST-3:	Dual-Band Circularly Polarized Antennas for GNSS Remote Sensing onboard Small Satellites	92
ATST-4:	Expectation Maximization - Matrix Pencil Method for Direction of Arrival Estimation	97
ATST-5:	CRLB and ML Algorithm for Joint Recovery of Carrier Phase and IQ Imbalance	102
ATST-6:	Synthesis of Adaptive Interpolated Beamformer	107

		Page Number
ATST-7:	The Effect of the Turn On Resistance of an Active Device and the Q Factor of the Load harmonic Network on the Efficiency and Efficiency Bandwidth of a Class E Amplifier	112
ATST-8:	The Effects of a Finite Ground Plane on the Characteristics of Printed Patch Antennas with and without a Suspended Patch	117
ATST-9:	Determination of the Transformers Turn Ratios and Design of a Circular Polarised Cross Slot Coupled Antenna	121
ATST-10:	To Design and Model a Class F Amplifier and Investigate the Effect of Losses on the Efficiency of dc to ac Power Conversion	125
ATST-11:	A Low Profile Radiating Element with Nearly Hemispheric Coverage for Satellite Communications On-The-Move Hybrid Array Antenna	129
BWN:	Broadband wired and wireless networks	
BWN-1:	Broadband future access networks with low energy consumption impact	134
BWN-2:	An analytical multiple scattering model to characterize free-space millimeter-wave and optical links in presence of atmospheric impairments	139
BWN-3:	Transparent Wireless Transmission over the ACCORDANCE Optical/Wireless Segment	144
BWN-4:	A Gateway to the European Optical Network Research	149
BWN-5:	Power Consumption of Wired Access Network Technologies	154
BWN-6:	An Efficient FPGA-Based Hardware Implementation of MIMO Wireless Systems	159
BWN-7:	Decode-and-Forward Relaying Technique in UMTS Networks	164
BWN-8:	Orthogonal-SLM Technique for PAPR Reduction and Recovery Without Side Information in OFDM Systems	169
CCEW:	Channel Coding and Equalisation for Wireless Broadband Communications Systems	
CCEW-1:	Joint Carrier Frequency Offset and Fast Time-varying Channel Estimation for MIMO-OFDM Systems	174
CCEW-2:	Efficient Channel Estimation for Chip Multiuser Detection on Underwater Acoustic Channels	180
CCEW-3:	Stochastic Modeling of Pseudolite Clock Errors Using Enhanced AR Methods	185
CCEW-4:	DMT Based Power-Line Communication System Channel Estimation Enhancement via Sparse Bayesian Regression	191
CCEW-5:	Simulated and Semi-Analytical Throughput Evaluation for AMC-OFDMA Systems	197
CCEW-6:	Pilot Aided Equalization with Constrained Noise-Estimation Filter	202
CCEW-7:	Closed-Chain Error Correction Technique for Turbo Product Codes	207
CCEW-8:	The Stability of LDPC Codes with Higher Order Modulation Schemes	213
CCS:	Chaos in Communications Systems	
CCS-1:	Reduced observer and its application to synchronization of chaotic systems	219

	Page Number
CCS-2: Chaotic optical phase generated by electro-optic and optoelectronic nonlinear and nonlocal delayed feedback: successful field experiment at 10 Gb/s	226
CCS-3: Implementation of a secure digital chaotic communication scheme on a DSP board	230
CCS-4: Performances of Chaos Coded Modulation schemes based on High Dimensional LDPC Mod-MAP mapping with Belief Propagation Decoding	235
CCS-5: Synchronization of hybrid systems for secure multimedia streaming	241
CCS-6: Toeplitz-Structured Chaotic Sensing Matrix for Compressive Sensing	247
CCS-7: Embedded Genesio-Tesi Chaotic Generator for Ciphering Communications	252
CCS-8: An FPGA Implementation of a Feed-Back Chaotic Synchronization for Secure Communications	257
CCS-9: Secure Digital Communication based on Hybrid Dynamical Systems	262
CN:	Computer Networks
CN-1: Network Modeling Technique: A Case Study	268
CN-2: Network Performance Evaluation Based on SoC design Methodology	274
CN-3: Optimizing Multicast Operation in Ethernet Carrier Networks	280
CN-4: Dynamic Generation of Flexible User Interface for Remote Control	285
CN-5: Gradient projection based RWA algorithm for OBS network	290
CN-6: On Evaluating the Latency in Handing Over to EAP-enabled WLAN APs from Outdoors	296
CN-7: K Nearest Neighbors Based on Lateration for WLAN Location Estimation	301
CN-8: The Cross-layer Adaptation of TCP-Friendly Rate Control to 3G Wireless Links	306
CN-9: An E-shop Log File Analysis Toolbox	312
CN-10: Improved Antnet Routing Algorithm with link evaporation and multiple ant colonies to overcome Stagnation Problem	318
CN-11: Analysis and Enhancement of SSL Based UMTS Authentication Protocol	327
EMSSN:	Embedded Mixed-Signal System and Sensor Networks
EMSSN-1: A passively powered BAW-based multi-standard in-tire identification and monitoring system	332
EMSSN-2: Energy and Throughput Optimization of a ZigBee-Compatible MAC Protocol for Wireless Sensor Networks	337
EMSSN-3: Designing a Reconfigurable Architecture for Ultra-Low Power Wireless Sensors	343
EMSSN-4: Parallel Text Mining for Large Text Processing	348
EMSSN-5: Comparison of Predicted Path Loss of the 3GPP Spatial Channel Model with the IMT- Advanced Model	354
GS:	General Session
GS-1: Global Motion Estimation Techniques for Compensation of Rolling Shutter Effect	358

	Page Number
GS-2: Single Step Optimal Block Matched Motion Estimation with Motion Vectors Having Arbitrary Pixel Precisions	364
GS-3: CMA-based Adaptive Antenna Array Digital Beamforming with Reduced Complexity	375
GS-4: Analysis and comparison of electrical characteristics for a single molecule wire with different electrode materials	380
GS-5: Design of Novel Sharp Transition Multiband FIR Filter	384
GS-6: Distributed Amplify-and-Forward Cooperation While Maintaining Transmission Freedom	389
GS-7: A Novel Architecture for Quantum-Dot Cellular ROM	395
GS-8: A Simple Mathematical Model for Clocked QCA Cells	399
GS-9: Active Low Pass Notch Filter Using Multielectrode Distributed RC Circuit	403
GS-10: A Measuring Set for Visualization of Ballistic Impact on Soft Armor	407
GS-11: Sequential Symbol Synchronizers based on Pulse Comparison by Positive Transitions at Half Rate	412
GS-12: GS-12: Single Channel Digital Tanlock Based Loop	416
GS-13: GS-13: Time Delay Digital Tanlock Loop with Acquisition-Aided Circuit	421

Nets4Cars: Communication Technologies for Vehicles

Nets4Cars-1: A Timer-based Intelligent Flooding Scheme for VANETs	425
Nets4Cars-2: A Message Propagation Model for Hybrid Vehicular Communication Protocols	430
Nets4Cars-3: Performance Evaluation of Phase Modulation Schemes for TETRA Application	435
Nets4Cars-4: Vehicle wireless communications Cell design	440
Nets4Cars-5: Performance Modeling Methodology of Emergency Dissemination Algorithms for Vehicular Ad-hoc Networks	445
Nets4Cars-6: Intra-Vehicular Verification and Control: A Two-Pronged Approach	449
Nets4Cars-7: Study on Effect of Adding Pupil Diameter as Recognition Features for Driver's Cognitive Distraction Detection	454
Nets4Cars-8: Data Rate Selection in WBSS-Based IEEE 802.11p/WAVE Vehicular Ad Hoc Networks	460
Nets4Cars-9: Coverage Performance of MB-OFDM UWB In-car Wireless Communication	465
Nets4Cars-10: Membership Service Specifications for Safety-Critical Geocast in Vehicular Networks	470
Nets4Cars-11: HMM-based Modelling of Roadside-to-Vehicle WLAN Communications	475
Nets4Cars-12: Cell Breathing and Cell Capacity in CDMA: Algorithm & Evaluation	480
Nets4Cars-13: Improved Time-Domain Channel Estimation Techniques in IEEE 802.11p Environments	485
Nets4Cars-14: Converging Time Synchronization Algorithm for Highly Dynamic Vehicular Ad Hoc Networks (VANETs)	491
Nets4Cars-15: Vehicle-2-Vehicle Communication Channel Evaluation using the CVIS Platform	497
Nets4Cars-16: Prioritizing Travel Time Reports in Peer-to-Peer Traffic Dissemination	502
Nets4Cars-17: Development of a Wireless Communication and Localization System for VRU eSafety	507

		Page Number
OTSP:	Optimisation Techniques for Signal Processing and Communications Systems	
OTSP-1:	A Review and Verification of Detection Algorithms for DVB-T Signals	512
OTSP-2:	Optimization of signal-processing algorithms in the Integrated Navigation System of car	518
OTSP-3:	An Adaptive Statistical Sampling Technique for Computer Network Traffic	523
OTSP-4:	Prototype Implementation and RF Performance Measurements of DSP Based Transmitter I/Q Imbalance Calibration	528
OTSP-5:	Multiuser Detection For Asynchronous ARGOS Signals	534
OTSP-6:	Integer Computation of JPEG2000 Wavelet Transform and Quantization for Lossy Compression	539
OTSP-7:	Optimal Cosine Modulated Nonuniform Linear Phase FIR Filter Bank Design via Stretching and Shifting Frequency Response of Prototype Filter	545
OTSP-8:	A Modified Evolutionary Algorithm Approach For Blind Channel	551
OTSP-9:	Nonlinear Single Channel Source Separation	556
OTSP-10:	On the filtering and smoothing of biomechanical data	561
OWC:	Optical Wireless Communications	
OWC-1:	Optical Multi-Input Multi-Output Systems for short-range free-space data transmission	566
OWC-2:	A unified approach for Channel Modeling of Terrestrial FSO Links	571
OWC-3:	Implementation of a 84 Mbit/s Visible-Light Link based on Discrete-Multitone Modulation and LED Room Lighting	577
OWC-4:	A Gigabit/s Indoor Optical Wireless System for Home Access Networks	581
OWC-5:	A MATLAB-based simulation program for indoor visible light communication system	586
OWC-6:	Indoor Optical Wireless System dedicated to Healthcare application in Hospital	591
OWC-7:	The effects of Gaussian Laser Beam Divergence on the BER of FSO Communication Links	596
OWC-8:	Comparing The Cloud Effects on Hybrid Network Using Optical Wireless and GHz Links	601
OWC-9:	Small Optical Transponder for Small Satellites	606
OWC-10:	Assessing Availability Performances of Free Space Optical Links from Airport Visibility Data	610
OWC-11:	An Overview of Indoor OFDM/DMT Optical Wireless Communication Systems	614
OWC-12:	Integration Scenarios for Free Space Optics in Next Generation (4G) Wireless Networks	619
OWC-13:	Optimal Frequency Band Division for Visible Light Communications	624
OWC-14:	Network Solutions for the LOS Problem of New Indoor Free Space Optical System	630
OWC-15:	On the Use of Free-Space Optical Links for Latency-Tolerant Traffic Applications	636
OWC-16:	Characterization and Modeling of Non-Line-of-Sight Ultraviolet Scattering Communication Channels	641
OWC-17:	Terrestrial Free-Space Optical Links with Temporal Diversity	646
OWC-18:	PPMPWM: a new modulation format for wireless optical communications	652

	Page Number
OWC-19: A Study of Discrete Wavelet Transform Based Denoising to Reduce the Effect of Artificial Light Interferences for Indoor Optical Wireless Communication	658
OWC-20: Free-space Optical Communication Employing Polarization Shift Keying Coherent Modulation in Atmospheric Turbulence Channel	663
OWC-21: Line-of-sight Visible Light Communication System Design and Demonstration	669
OWC-22: Optical Attenuation Modeling of Continental Fog using Exponential Distribution	674
OWC-23: The Use of Linear Projections in the Visual Analysis of Signals in an Indoor Optical Wireless Link	680
PCSN:	Photonics Communications Systems and Networks
PCSN-1: An Error-Free Protocol for Quantum Entanglement Distribution in Long Distance Quantum Communication	686
PCSN-2: Design Rules for Full Pattern-Operated All-Optical XOR Gate With Single Semiconductor Optical Amplifier-Based Ultrafast Nonlinear Interferometer	691
PCSN-3: OCDMA system performance with Prime Codes under beat noise constraints	696
PCSN-4: SOA Gain Uniformity Improvement Employing a Non-Uniform Biasing Technique for Ultra-High Speed Optical Routers	702
PCSN-5: Scalable and data format agnostic optical packet switch sub-systems for optical packet switched networks	708
PCSN-6: A Novel Design to Compensate Dispersion for Square-lattice Photonic Crystal Fiber over E to L Wavelength Bands	714
PCSN-7: Heuristic for Reducing Congestion during Logical Topology Design in De Bruijn WDM Networks using Survivable Routing	719
PCSN-8: Adaptive Algorithms for Nonlinearity Modeling in Laser Heterodyne Interferometer	725
PCSN-9: Repetition Rate and Wavelength Tunable All-Optically Mode-locked Fiber Ring Laser Based on a Reflective Semiconductor Optical Amplifier	730
PCSN-10: A Comprehensive Modeling of Wave Propagation in Photonic Devices	734
PCSN-11: Optical Performance Monitoring Techniques for High Capacity Optical Networks	738
PCSN-12: Chromatic Dispersion Compensation Employing Cascaded Parallel Optical All-pass Filter	742
PCSN-13: Demonstration of the transportation of microwave environment over an optical 10 Gbps Ethernet based IP network	748
PCSN-14: Coupling Gaussian Beam into 30 microns Wide Grating Coupler Using a Novel Expanded Core Fiber	753
PCSN-15: Optical Polymer and Polymer-Clad Silica Fiber Data Buses for Automotive Applications	758
PCSN-16: Multi-Zones of Cognitive to Fibre Networks	762
PCSN-17: Performance of Digital Optical Communication Link: Effect of In-Line EDFA Parameters	766
SC:	Security and Cryptography
SC-1: A New Development of Cryptosystem Using New Mersenne Number Transform	771
SC-2: A New Development of Public Key Cryptosystem	776

	Page Number
SC-3: Performance of Keystroke Biometrics Authentication System Using Multilayer Perceptron Neural Network (MLP NN)	781
SC-4: Evaluation of One-Dimensional NMNT for Security Applications	785
SC-5: The SIESTA project: Near Field Communication based applications for tourism	791
SC-6: Some Security Issues For Web Based Frameworks	796
SC-7: A Novel Modality Independent Score-Level Quality Measure	802
SC-8: Comparison and Development of a Competition Model for a Cryptosystem Based on the Analytic Network Process and Bayesian Networks	806
SC-9: Number Transform Signal Encryption Using New Mersenne Number Transform	812
 SIP: Signal and Image Processing	
SIP-1: Design and Implementation of a Wireless Multi-Channel EEG Recording	817
SIP-2: Improved Lowpass Differentiator for Physiological Signal Processing	823
SIP-3: Facial Tracking Method for Noncontact Respiration Rate Monitoring	827
SIP-4: Human Identification using time normalized QT signal and the QRS complex of the ECG	831
SIP-5: Breast Cancer Prediction and Cross Validation Using Multilayer Perceptron Neural Networks	836
SIP-6: Performance Efficient FPGA Implementation of Parallel 2-D MRI Image Filtering Algorithms using Xilinx System Generator	841
SIP-7: Real-Time Vision Based Respiration Monitoring System	846
SIP-8: A Review of Content-Based Image Retrieval	851
SIP-9: Neural Networks-Based Tool for Diagnosis of Paranasal Sinuses Conditions	856
SIP-10: Breast Cancer Detection using Microwave Holography	861
SIP-11: A New Algorithm for Completing Fragmented Boundaries in Images	865
SIP-12: Fundamental Issues in Antenna design for Microwave Medical Imaging Applications	869
SIP-13: Multimodal Biometric Fusion at Feature Level: Face and Palmprint	875
SIP-14: Hyperspectral Image Compression with Modified 3D SPECK	880
SIP-15: ECG Signal Analysis for Arrhythmia Detection	885
SIP-16: Performance Improvement Algorithms for Colour Image Compression Using DWT and Multilevel Block Truncation Coding	891
 TMTE: Teletraffic Models and Traffic Engineering	
TMTE-1: Compression Mechanism for Multi-service Switching Networks with BPP traffic	896
TMTE-2: QoS Guarantee in the Erlang Multirate Loss Model based on Derivatives of Blocking Probabilities	902
TMTE-3: Chip Impedance Characterization for Contactless Proximity Personal Cards	906
TMTE-4: Model of the k-cast connections in mobile networks	911
TMTE-5: ON-OFF Traffic Models for a Hybrid TDM-WDM PON with Dynamic Wavelength Allocation	916
TMTE-6: The Evaluation of the Quality of Multicast Routing Algorithms in Packet Networks	921

		Page Number
WN:	Wireless Networks	
WN-1:	Chirp Sounder Measurements for Broadband Wireless Networks and Cognitive Radio	926
WN-2:	Cooperative control of connected micro-cells for a virtual single cell for fast handover	932
WN-3:	VoIP Design and Implementation with Network Coding Schemes for Wireless Networks	937
WN-4:	Reinforcement Learning Algorithm for Optimised Cross Layer Medical Video Streaming over WiMAX Networks	942
WN-5:	A Contribution to Laboratory Performance Measurements of IEEE 802.11 a/g WEP Point-to-Point Links using TCP, UDP and FTP	947
WN-6:	Methodology to Evaluate and Improve the QoS ICT Networks in the Healthcare Service	951
WN-7:	Cluster topology formation in Manet: a mathematical model approach	956
WN-8:	Decentralized Spectral Resource Allocation for OFDMA Downlink of Coexisting Macro/Femto Networks using Filled Function Method	961
WSN:	Wireless Sensor Networks	
WSN-1:	End-to-End QoS in Integrated Wireless and Sensor Network: System Implementation	966
WSN-2:	Deployment and Performance Issues of an Integrated Wireless Sensor Network and Wireless Mesh Campus Network	970
WSN-3:	Applying Formal modelling to detect DoS attacks in wireless medium	976
WSN-4:	Decentralized BDI-based Intelligent Multi-agent for Optimizing Wireless Sensor Network	981
WSN-5:	An approach to the automated selection of protocols for a wireless network design	987
WSN-6:	Survey and Analysis of Power Control for Collaborative Networks	993
WSN-7:	Analytic Hierarchy Process Applied to a Wireless Sensor Network for the Cluster Head Selection	999
WSN-8:	Hydrophone Calibration Based on Microcontrollers for Acoustic Detection of UHE Neutrinos	1004

Designing a Reconfigurable Architecture for Ultra-Low Power Wireless Sensors

Johann Glaser, Jan Haase, Christoph Grimm
Vienna University of Technology
Institute of Computer Technology
EMail: {glaser,haase,grimm}@ict.tuwien.ac.at

Abstract—In wireless sensor network nodes all tasks are controlled and scheduled by the CPU of the node. It is activated repeatedly from its low-power sleep mode for e.g., measurements and communications tasks. The periodic wake-ups cause a high overhead in power consumption. This problem can be solved by supplementing the CPU with additional modules which autonomously execute selected tasks to enable the CPU to stay in its inactive low-power mode. It is only activated if more complex processing is required, e.g. for generating a network packet to transmit a changed sensor value. We introduce logic modules with a reconfigurable architecture which allow to flexibly adopt the functionality but still perform relatively complex tasks. This work presents the methodology for the design of these reconfigurable logic modules.¹

I. INTRODUCTION

Wireless sensor network (WSN) nodes comprise a central processing unit (CPU), memory (both volatile and non-volatile), a timing unit like a real time clock (RTC), sensors for environmental measurements (e.g., temperature), communication modules like RF transceivers and a power supply [1]. The applications of WSNs are manifold, including building automation, automotive systems, container tracking and environmental monitoring.

Every application poses different requirements on the WSN and its nodes. Hence, a large amount of different architectures, network protocols, sensor and power supply techniques are available. The optimization goal for a specific WSN node is determined by its field of application. So, there is no WSN node architecture which is suitable for all applications, let alone the optimum for all applications [1].

Besides this there is one requirement which is common to most applications: low energy consumption of the WSN node due to limited power scavenging potential and a limited battery capacity. The main approach for reducing the power consumption is to reduce the activity of the WSN node by periodically entering an inactive low-power mode. On the other hand, every task of the node has to be performed by the CPU, which controls the whole node.

We propose to insert specific reconfigurable peripheral control modules into the WSN node as depicted in Fig. 1 (shaded boxes). These independently perform certain sub-tasks and only activate the CPU if further, more complex processing is required. With this approach the CPU can

¹This work was funded by the Austrian government via FIT-IT (grant number 815069/13511) within the European ITEA2 project GEODES (grant number 07013).

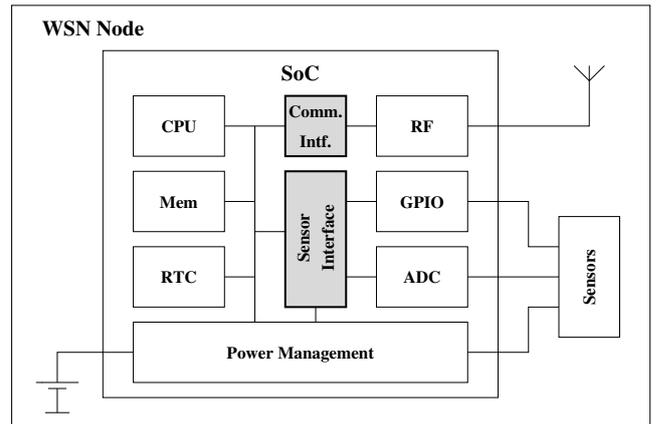


Fig. 1. WSN node with reconfigurable hardware blocks (CPU: central processing unit, Mem: memory, RTC: real time clock, Comm.Intf.: communication interface, RF: radio frequency transceiver, GPIO: general purpose IO, ADC: analog-to-digital converter).

remain in an inactive low-power mode for extended periods and thus reduce the total power consumption [2]. While the power consumption of a peripheral module is independent whether it is controlled by the fully-featured CPU or by a dedicated peripheral control module, the peripheral control module must have considerably less power consumption than the CPU. This is ensured due to its specialized design and therefore small count of logic gates.

For example, for the periodic measurements of the sensor values the CPU is activated from its inactive low-power mode. Then the power supply of an external sensor is turned on. After some settling time the sensor value is ascertained by an analog/digital conversion (ADC). The resulting value is finally compared to the previous value and only if the difference is greater than a certain threshold, further processing is conducted, like sending the value via the wireless network. Otherwise the CPU immediately switches back to an inactive low-power mode.

Another example is the MAC (media access control) layer of the network stack. For a network protocol with periodic listening intervals the receiver is turned on for a certain time waiting to receive a packet. If a packet is received, its destination address is inspected to decide upon further processing. After these relatively simple tasks the more complex tasks of the higher protocol layers follow.

These cycles consist of several simple tasks (set and read digital IOs, compare two integers) where delays occur (sensor settling time, network listening). These are usually performed by the CPU, which is overqualified and oversized for these tasks. It consumes power during the whole active state as well as for the wakeup procedure. The proposed reconfigurable peripheral control modules are intended to avoid this waste of energy. Here we will discuss the methodology for the design of these modules.

Shifting the border in a software/hardware partitioning process towards hardware reduces the flexibility of the final application. While software can be modified by reprogramming the code memory, synthesized logic cores require a redesign of the chip. Still, there are three important reasons for flexibility:

- 1) Covering multiple different applications for a higher market potential.
- 2) Adopting to different external components across PCB design cycles.
- 3) Fixing bugs without a chip redesign.

Therefore we propose to make the introduced dedicated hardware blocks *reconfigurable*.

The next section describes alternative approaches as well as the basis of our methodology. This is followed by a description of the design flow for ultra-low power reconfigurable logic modules and a section on the design tools. In the following section an example of this design flow is given. This work is finished with conclusions and an outlook to future work.

II. RELATED WORK

The above mentioned sensor interface task was investigated in [3] for a conventional implementation where the CPU programmed with firmware performs all control tasks as well as for two different hardware implementations, one using the proposed reconfigurable architecture and the other using a hard-coded finite state machine (FSM). The latter reduced the power consumption by a factor of more than 90 while the reconfigurable hardware implementation still achieved a reduction by a factor of nearly 34, both compared to the firmware implementation. This number is still largely underestimated, because the authors used an FPGA as underlying architecture instead of a custom chip. From this result we conclude, that our goal to reduce the power consumption of a WSN node is achievable by relieving the CPU from such simple tasks.

An embedded FPGA was presented in [4] for the control of IO ports and protocols. The RISC CPU core of the high-performance SoC can be extended by the “Pipelined Configurable Gate Array (PiCoGA)” where application-specific functions can be mapped ([4, p. 87]). While this approach is very flexible, it is clearly designed for high-performance embedded systems and does not concentrate on low-power design.

The high overhead of an FPGA was tackled by the soft embedded programmable logic cores presented in [5]. They used a non-regular architecture, but synthesis tools get into difficulties then. In later work the overhead of standard cells was reduced by the introduction of

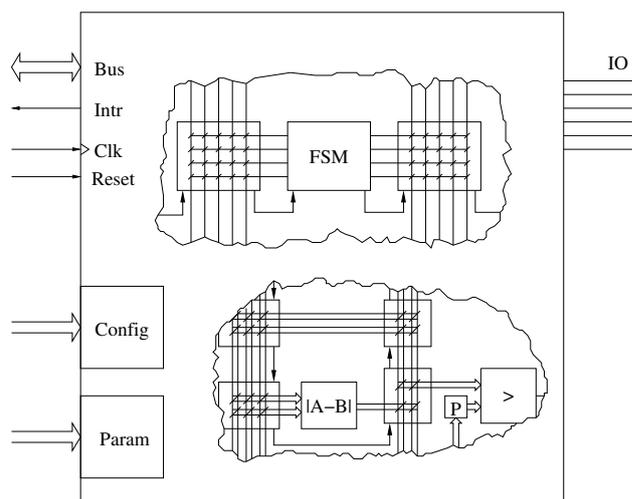


Fig. 2. Internals of a reconfigurable module. Intr: Interrupt, FSM: Finite State Machine, P: Parameterization Cell, Config: Configuration Interface, Param: Parameterization Interface.

architecture specific tactical cells [6]. This work is a good starting point for a logic core inserted between a sensor interface and the CPU, but still lacks the optimization potential spanned by the anticipated knowledge of the application.

An FPGA holds a large array of universal logic elements with versatile routing resources to combine the elements. This ensures that the underlying reconfigurable FPGA architecture is appropriate for virtually any application. In [2] we presented a different approach on a reconfigurable architecture for peripheral control modules in WSN nodes. Based on the finding that the configuration bit stream of FPGAs consists of only 10% for the logic configuration and 90% for the routing configuration, we proposed to reduce this high overhead by designing the underlying hardware architecture tailored to the specific *application class* of each reconfigurable module, (e.g. communication interface and sensor interface in Fig. 1). This defines the range of applications of the reconfigurable module. Referring to the previous sensor interface example, the application class defines how these sensors can be controlled as well as the sensor value post processing algorithms. For the final application the actual sensor and actual post processing are specified. From this the configuration of the reconfigurable module is derived and applied during system startup.

In our approach we limit the range of applications of a single reconfigurable module and therefore are able to place specialized cells and reduce the routing resources. The specialized cells include arithmetic operations like adders or the absolute difference of two integer values (e.g. $|A - B|$ in Fig. 2). [6] report a reduction of area and delay overhead by 58% and 40%, respectively with the introduction of these so called “tactical cells”.

Reconfigurable logic modules typically also contain control logic like a finite state machine (FSM). The logic which calculates the output signals as well as the next state is usually built from many combinational gates. [7] introduced an architecture which directly treats the

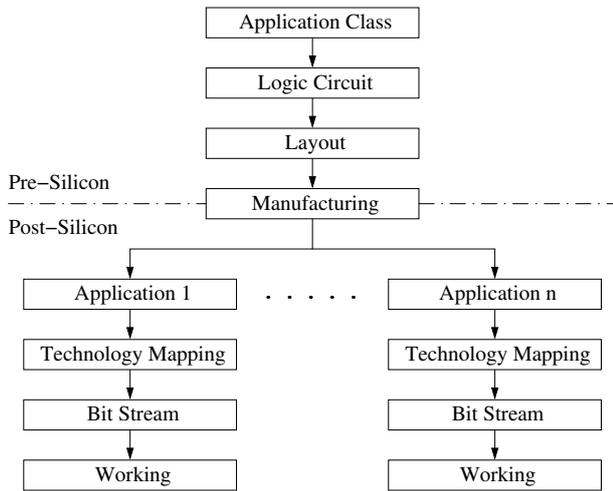


Fig. 3. Design flow diagram for reconfigurable logic modules

state transitions instead of the transition function. With this approach the total count of configuration bits was reduced to only between 7.3 and 12.7 % compared to an implementation with a commercial FPGA for certain sets of predefined automata.

III. DESIGN

The design of a reconfigurable logic module is a two-part process. In the *pre-silicon phase*, the application class is defined and a reconfigurable circuit is developed. This results in chip layout data and finally in a produced chip. In the *post-silicon phase*, the reconfigurable logic block is configured, analogous to an FPGA, to implement the actual application (see Fig. 3).

A. Pre-silicon Phase

In the pre-silicon design phase the desired class of application has to be specified. Several different but similar applications are considered (also referred to as a *set of target applications* by [8]). The requirements for logic blocks and connectivity are specified and used to develop the deployed gates and routing resources which finally result in the chip layout.

Note that for other reconfigurable architectures like FPGAs the pre-silicon phase is usually performed by the chip vendor. The internal structures, gates, connections and switches are designed to meet the requirements of as many applications as possible. The designer will then utilize these structures to implement his application.

In the above sensor interface example, the application class is defined by the desired range of covered sensors and post-processing algorithms. Some external sensors only require a power supply for their activation while others need an additional selection signal. Some sensors provide an output signal to show that the value is stable and others just specify a time interval. Some sensors have an analog output and others have a digital interface (e.g., I²C, SPI). The post-processing includes a comparison of the current value with the previous one. Only if the difference exceeds a certain threshold, the CPU is notified to take over control and perform more complex tasks.

All combinations of these cases have to be covered by the reconfigurable logic block. It needs interfaces to peripherals (ADC, I²C and SPI) and the CPU (interrupt), external signals (power supply of the sensor, activation signal, ready signal), a timer (settling time), a comparator for the sensor value, and registers to store the old value and the (configurable) threshold. This threshold and the timing values are setup via special parameterization interfaces to allow adoption of numeric value during operation without applying a different configuration. This is complemented by a flexible state machine to control the order of processing (compare Fig. 3).

From this specification a logic circuit is designed which includes the required blocks. All these parts are connected via a routing infrastructure to provide the connections necessary to implement the desired functionality. This is enriched by the configuration and parameterization infrastructure logic. With synthesis, technology mapping and place and route tools the final layout of the reconfigurable logic block is developed. The pre-silicon phase is finalized by the production of the SoC.

After production, the hardware circuits can not be modified (with reasonable costs) so the user has to get by with the available structures to implement the desired application. This means that the pre-silicon design sets restrictions to the post-silicon design space by the available set of blocks, routing resources and configuration and parameterization options.

B. Post-silicon Phase

The post-silicon phase is analogous to the design flow for an FPGA. The actual application is specified and designed. We plan to utilize commercially available tools to synthesize the design to a netlist. For special tactical cells like an FSM a stand-alone description is fed to custom tools which derive the configuration of these cells.

The synthesized netlist is then mapped to the underlying logic module inventory as designed in the pre-silicon phase. Due to the complex non-regular structure we expect difficulties for the tools as already mentioned by [5]. These also have to check timing constraints. The output of this step is a technology specific netlist. The cell instances are then mapped to the locations of the available hardware cells and the connections between these cells are assigned.

From the placement and routing information and the specified functionality of configurable cells (e.g. the FSM) the content of the configuration store is constructed. This is mapped to the configuration chain and finally to the bit stream.

At system start up of the SoC the integrated CPU runs its boot sequence. This includes the setup of peripheral units like timers and ADCs by programming the according registers. Additionally the configuration bit stream is downloaded to the reconfigurable logic modules. We propose to implement the configuration interface as memory mapped registers. After configuring the reconfigurable logic module with the loaded bit stream (termed “configware” by [9], analogous to “software”) the

parameter registers are set to the required values. Finally the reconfigurable logic module is activated and starts to perform its task as specified by the application definition.

IV. TOOLS

This section gives an overview of the tool chain utilized for the above design flow. This is a topic of ongoing research and therefore preliminary and work in progress.

The design starts with the definition of the application class. Current research is performed on the format of this specification and on how an automated flow can process it. The output of this step are HDL (hardware description language) files with RTL (register-transfer-level) and structural code. These instantiate tactical cells as well as reconfigurable cells (e.g. routing box and configuration registers).

These files are fed to a commercial synthesis tool which outputs a netlist. The synthesis tools uses standard cell libraries together with custom libraries which provide optimized cells for the logic module, e.g. multi-bit cells. These are automatically inferred during the synthesis. For interfacing to the commercial tool we prefer open standard formats, especially the EDIF file format (electronics design interchange format) for the netlist and the Open Liberty Library file format for the custom library.

The placement and routing of the netlist is performed by automated commercial tools. The proposed design flow also includes manual placement and routing on a hierarchically higher level. Therefore the automated tools are used for small blocks. Several of these are then placed and routed by hand. This can provide improved delay and area characteristics by utilizing structural knowledge.

The post-silicon phase depends on the resources provided by the pre-silicon phase. To import the description of the reconfigurable architecture and its inventory and routing resources, the netlist from the pre-silicon phase is loaded. Alternatively the pre-silicon tool could generate a machine readable architecture description besides the previously mentioned HDL files, which is used instead of the netlist.

Two differently complex variants of the post-silicon synthesis are planned. The simple variant provides a graphical representation of the hierarchical reconfigurable architecture and permits to configure every cell including the routing connections to implement the actual application. After the module configuration is completed, the bit stream is generated utilizing the knowledge of the configuration chain. This variant does not require any higher intelligence in the tool and provides the user with full control over the configuration.

A lot more intelligence is required for the complex variant tool. This supports the synthesis of an HDL description of the actual application. The resulting netlist is mapped to the underlying technology provided by the pre-silicon phase. The final automatic place and route results in the same data as the user has to setup by hand with the simple variant tool. After that the configuration bit stream is generated by the same routine as used by the simple variant.

V. EXEMPLARY APPLICATION

From the many components used in a reconfigurable module like combinational logic, registers, a finite state machine (FSM), the data path, routing and parameterization values, we pick the FSM and give an example of the above described design flow. For simplicity we only look at a single component instead of a whole reconfigurable module.

A. Transition-based Reconfigurable FSM

The Transition-based Reconfigurable FSM (TR-FSM) was presented in [7]. It is a reconfigurable cell which is then integrated into the whole reconfigurable logic module. Based on the fact, that FSMs in actual applications have considerably less transitions than the theoretical maximum for a given number of states and input signals, the TR-FSM focuses on the *transitions* rather than on the state transition function. It provides several smaller reconfigurable blocks for implementing each transition (see Fig. 4).

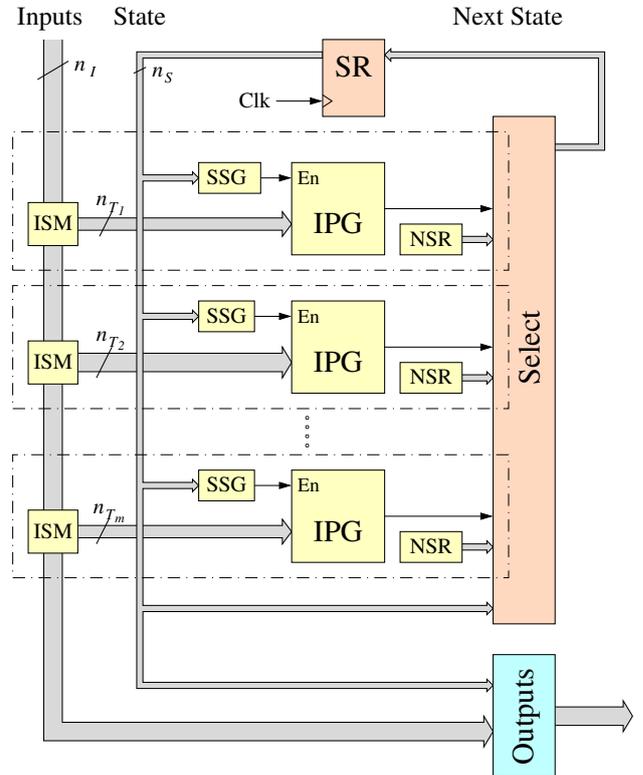


Fig. 4. The overall TR-FSM architecture (SR: state register, ISM: input switching matrix, SSG: state selection gate, IPG: input pattern gate, NSR: next state ID register).

The basis of the proposed architecture is the so called *transition row*. A state selection gate (SSG) compares the current state to its configured value and enables the transition row. A reconfigurable input switching matrix (ISM) selects a subset of the input signals which then serve as the inputs for the input pattern gate (IPG), which is a reconfigurable combinational logic block, e.g. a look-up table (LUT). If activated, the IPG outputs a “1” iff the input pattern matches a certain transition from the

recognized state. The reconfigurable next state ID register (NSR) of the corresponding transition row is then selected by a multiplexer (“Select”) and fed back to the state register (SR) as the new state of the FSM.

The overall architecture contains many such transition rows with varying input width. The output signals are computed in a separate configurable logic block (“Outputs” in Figure 4), which for example can be a LUT or an embedded FPGA IP. The output pattern can also be included in every transition row, but this will increase the number of required transition rows if the output pattern depends on input patterns for common next states.

B. Logic Design

In the *pre-silicon phase* all parameters for the chip circuitry have to be specified. Regarding the TR-FSM these are the number of input and output signals, the width of the state vector, the number of transition rows and the number of inputs for each transition row. These parameters are derived from the specification of the application class. This is either given by a set of predefined automata which have to be implemented by the TR-FSM or by a list of common properties of the implemented automata.

The first case is investigated in more detail here. Besides the input and output signals of these automata, the state transitions are investigated to find the total number of transitions and the number of inputs from which each of these depends. The TR-FSM hardware is then specified by the maximum of transitions and the maximum of the numbers of the transition rows. Note that a transition row with n_{T_i} inputs can also implement a transition with $n_{T_j} < n_{T_i}$ because the IPG is able to ignore the state of inputs. For example, one FSM requires 7 states with 3 inputs each plus 3 states with 2 inputs each. Another FSM requires 3 states with 3 inputs each but 7 states with 2 inputs each. In this case, we need 7 transition rows with 3 inputs plus 3 transition rows with 2 inputs. Four of the 7 states with 2 inputs each of the second FSM can be mapped to transition rows with 3 inputs. Additionally the TR-FSM offers the flexibility to implement either an FSM with a high number of states with very few transitions each, as well as an FSM with a low number of states with many transitions each or a combination of both.

From this specification the semiconductor is produced. In the *post-silicon phase* the actual FSM is specified. This includes the number of states and the input patterns for each transition with their next states. From this FSM description the configuration of the ISMs is generated to select exactly these input signals, which are inspected by the respective states. The configuration of the IPGs is generated so that they output a logic “1” in every case (depending on the selected inputs), where the transition is active. Finally the state mapping is performed by assigning a bit pattern to every state which is then used to generate the configuration of the SSGs and NSRs. All this configuration information is then assembled to a single configuration bit stream to be downloaded into the TR-FSM configurable block.

VI. CONCLUSION

This paper discusses the methodology for designing reconfigurable logic modules for WSN nodes. These are placed additionally to the CPU into a WSN node SoC and autonomously perform tasks of moderate complexity. Due to the reconfigurability of every module, its functionality can be adapted flexibly. By relieving the CPU from these tasks, it will remain in an inactive low-power mode for extended periods which leads to fundamental power reductions.

Every reconfigurable logic module is specifically tailored to the application class of its operational area. In the pre-silicon phase the reconfigurable resources including the routing are specified. After chip production, i.e. in the post-silicon phase, the actual application is specified. The configuration is derived from the application description and mapped to a configuration bit stream. This is applied to the reconfigurable circuitry and so configures it to implement the application logic. The objective is to reduce power consumption while maintaining a high degree of flexibility for the implemented applications.

Precise measurements of the power reduction will be gained and published after the production of a test chip in a 130 nm CMOS process, which is scheduled for end of May 2010. Future work includes research on reconfigurable routing, development of multi-bit cells and a tool chain to ease the implementation of reconfigurable hardware modules.

REFERENCES

- [1] H. Karl and A. Willing, *Protocols and Architectures for Wireless Sensor Networks*. Wiley, 2005.
- [2] J. Glaser, J. Haase, M. Damm, and C. Grimm, “A Novel Reconfigurable Architecture for Wireless Sensor Networks,” in *Tagungsband zur Informationstagung Mikroelektronik 10*. Vienna, Austria: OVE, 7.-8. Apr. 2010, pp. 284–288.
- [3] —, “Investigating Power-Reduction for a Reconfigurable Sensor Interface,” in *Proceedings of Austrochip 2009*, Graz, Austria, 7. Oct. 2009.
- [4] A. Lodi, A. Cappelli, M. Bocchi, C. Mucci, M. Innocenti, C. D. Bartolomeis, L. Ciccarelli, R. Giansante, A. Deledda, F. Campi, M. Toma, and R. Guerrieri, “XiSystem: A XiRisc-Based SoC With Reconfigurable IO Module,” *IEEE Journal of Solid-State Circuits*, vol. 41, no. 1, Jan. 2006.
- [5] S. J. E. Wilton, N. Kafa, J. C. H. Wu, K. A. Bozman, V. O. Aken’Ova, and R. Saleh, “Design Considerations for Soft Embedded Programmable Logic Cores,” *IEEE Journal of Solid-State Circuits*, vol. 40, no. 2, pp. 485–497, Feb. 2005.
- [6] V. Aken’Ova, G. Lemieux, and R. Saleh, “An Improved “Soft” eFPGA Design and Implementation Strategy,” in *Proceedings of the IEEE Custom Integrated Circuits Conference (CICC)*, 18.-21. Sep. 2005, pp. 179–182.
- [7] J. Glaser, M. Damm, J. Haase, and C. Grimm, “A Dedicated Reconfigurable Architecture for Finite State Machines,” in *Reconfigurable Computing: Architectures, Tools and Applications, 6th International Symposium, ARC 2010*, ser. Lecture Notes on Computer Science, vol. LNCS 5992. Bangkok, Thailand: Springer, Mar. 2010, pp. 122–133.
- [8] J. O. Filho, T. Schweizer, T. Oppold, T. Kuhn, and W. Rosenstiel, “Tuning Coarse-Grained Reconfigurable Architectures towards an Application Domain,” in *IEEE International Conference on Reconfigurable Computing and FPGA’s, ReConFig 2006*, Sep. 2006, p. 7.
- [9] R. Hartenstein and V. Milutinovic, “Introduction to the Configurable Minitrack – From Glue Logic Synthesis to Reconfigurable Computing Systems,” in *Proceedings of the 32nd Annual Hawaii International Conference on System Sciences, 1999. HICSS-32*, vol. Track 3, 1999, pp. 326–326.