802.15.4 Powered Distributed Wireless Smart Cameras Network

Erik Ljung,
Erik Simmons
Philips IC-Lab,
Building SFJ-6, Glaslaan 2,
5616 LW Eindhoven, The Netherlands
+31-(0)40-2732330
{ljunge, simmonse}@iclab.ce.philips.com

Alexander Danilin,
Richard Kleihorst, Ben Schueler
Philips Research Europe
High Tech Campus 5,
5656 AE, Eindhoven, The Netherlands
+31-(0)40-2742366
{alexander.danilin, richard.kleihorst, ben.schueler}@philips.com

ABSTRACT
This article introduces a novel communication approach to share resources in wireless smart camera networks. The communication is based on the ZigBee™ MAC layer (IEEE 802.15.4). The main idea is to use a packet-based approach to communicate directly between different sub-systems of different wireless smart cameras. Having one packet definition through all software layers helps to reduce the overhead for the inter-layer communication and thus increase the performance. Using this novel communication approach we effectively implemented different wireless smart camera applications.

General Terms
Algorithms, Design, Experimentation, Human Factors.

Keywords
Wireless, Smart Camera, Networks, ZigBee MAC Layer, 802.15.4.

1. INTRODUCTION
A smart camera is a device that performs real-time analysis of complex (natural) video scenes. Smart cameras are gaining more attention as they become less expensive, less power voracious, and more computationally efficient. The recent advances in technology help to cope with an insatiable demand for real-time performance of video processing algorithms. Modern smart cameras not only perform pixel processing, but also run rather complicated algorithms to extract valuable information from streaming video, i.e. human detection, surveillance, motion analysis, facial detection, and traffic monitoring [2][3][4][12][13].

As smart cameras become cheaper, more possibilities arise for building distributed smart camera environments to get higher resolution of a scene, detect persons in 3D space, or to improve occlusion handling [6]. The communication between such distributed cameras is desired to be wireless to enable the mobility and adaptability. Adding a wireless aspect to smart cameras leads to the challenge optimizing the power consumption. Because of increasing computational efficiency of digital circuitry and the limited power efficiency of the transmitter circuit [10] image processing and compressing should be part of the camera.

To challenge this problem we agree with [5] transmitting only selected pieces of information between the smart cameras to minimize the communication bandwidth and power consumption. Instead of using Internet protocols for communication we use the IEEE 802.15.4 MAC layer. To bring down the power consumption even further we transfer only a high-level description of an actual event occurring in a video scene. This high level description is understandable by the other distributed smart cameras, and requires only a few bytes to be transmitted.

This paper presents a software stack developed for the Philips wireless smart camera WiCa (Figure 1) [7]. This software stack allows the building of a powerful distributed smart camera environment that performs real-time video processing for different types of applications ranging from surveillance to gesture recognition and ambient intelligence.

The main contributions of this paper are two-folds. First, we use the ZigBee™ MAC layer (IEEE 802.15.4) to enable really low-power standalone wireless smart cameras network. Second, we design a universal packet-based protocol to ease the communication in the distributed smart vision applications.

Figure 1. WiCa: Wireless Smart Camera.

2. WiCa HW Architecture

The WiCa smart camera consists of one or two color VGA sensors, a high-performance, low-power, vision processor and a DSP host. The vision system is fully programmable in C++ and can analyze the scene for event detection and/or event description. Its block diagram is shown in Figure 2.

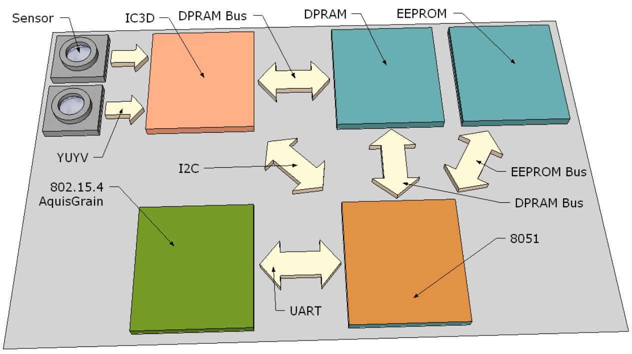

Figure 2. WiCa Block Diagram.

We use the IC3D as a vision processor (pixel processor), which is a straight successor of Xetal-I with improved performance. The IC3D has 320 RISC processors; it is a highly parallel SIMD machine and thus low-power [1]. The maximum performance of this processor is about 50GOPS. IC3D runs the low-level part of an application – it collects and analyzes raw video data.

To process video events data we use a well-known upper class Atmel 8051 processor [8]. This processor has an internal EEPROM to store IC3D programs; it is also connected to the I^2C bus and handles interrupts from the IC3D.

Both the IC3D and the 8051 are connected via a dual-port RAM (DPRAM), large enough to store video information [7].

For communication purposes we use the AquisGrain (AG) module developed at Philips Research, Aachen [9]. At this moment only the ZigBee IEEE 802.15.4 MAC layer is implemented in the AG devices. We have built all network maintenance ourselves. There is already a next version of the AG available, which holds a complete ZigBee stack. We plan to use this in the next WiCa generations.

3. SW Architecture

Applications will retrieve data from the IC3D and bundle it into events and features, which are shared over the network for distributed scene analysis. The basic idea of the SW architecture is to provide a framework for applications and to automate general functionality.

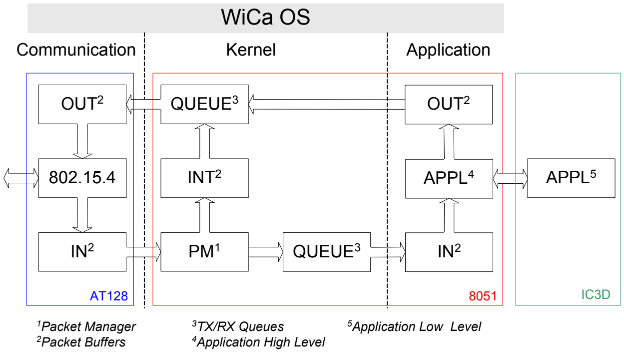

Figure 3. SW Modules.

The WiCa software stack consists of four modules. The first module is responsible for the wireless communication part and controls the IEEE 802.15.4 MAC layer. The second module is the kernel controlling the WiCa HW. The third and fourth modules are the application code divided into high-level processing and low-level processing.

The last two modules are executed together in the 8051 and the IC3D pixel processor. All four modules are linked in a chain with each other; the first three form the WiCa OS. The WiCa OS manages the general packet-flow of the smart camera platform (see Figure 3).

High-level and low-level processes in applications interact either via the DPRAM or via the I^2C registers and the interrupt (see 3.3).

The first module controls the wireless communication and is executed on the AG devices. The main responsibility for the module is to setup and maintain the network and to forward packets in/out of the WiCa. Packets are sent/received with the IEEE 802.15.4 MAC layer. The module handles three general types of packets. Communication packets are sent between devices to maintain the network structure, ISP packets are sent to reprogram the 8051 and WiCa OS packets are simply forwarded to the next module.

Each AG module has a unique hard coded number (UID) to identify it on the network. The WiCa itself has no UID – if another AG module is attached, there will be another UID assigned to this WiCa. Besides the UID, every AG module has a network NID; this NID is assigned to every AG module by a network coordinator, similarly to DHCP on the Internet. If one resets the WiCa, there will be another NID assigned, but the UID of this WiCa remains the same. In short, UID is needed to run the distributed application, while NID is needed to maintain the network.

The kernel and the high-level part of the application are both executed in the 8051 but are separated as different tasks. The task switch is a round-robin system with an extra feature to enable/disable switching.

The kernel is a high priority task, which cannot be preempted by the application. In practice, the kernel immediately disables task switching when it is executing and re-enables it upon finished work. The kernel task forwards packets in/out from the application and due to the none-preemption it can guarantee packet delivery. The application works with blocking calls to the rx/tx-queues, which removes the issues of buffer overflows. Internal packets (i.e. IC3D commands, IC3D code and administrative packets) are handled internally in the kernel and possible responses are put in the tx-queue and handled by the normal kernel process.

The application module is more a template than an implementation. The template provides possibilities to setup tasks and to send/receive packets stamped with the application type. Function calls to the WiCa OS are available and build the basis for the applications communication with the IC3D. In other words: the template is an empty working application.

3.1 Packet-based approach

Information in a WiCa based sensor network flows through different HW subsystems and the communication interfaces differ from each other. A layered model abstracting the HW would be a natural way to implement the communication. In our approach we designed a small packet header, which represents different events. The same packet header is used throughout the entire data-flow. Figure 4 illustrates the different steps in the packet-flow between the HW, which use the general event packet.

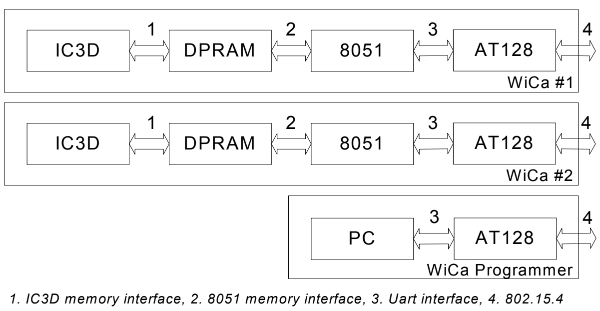

Figure 4. Packet Flow.

This approach minimizes the communication overhead and data transfers, which leads to reduced power consumption due to less processing and wireless transmissions. The fact that each subsystem of a sensor network can communicate with each other on the same basis widens the overall intelligence of the system. The possibility for a subsystem to communicate eases the distribution of computational power over the nodes[1]. Each node or part of a node co-works with other units on the network forming a distributed sensor network. The network is not only distributed over the nodes but even between subsystems of nodes. The distribution can be either internal, external or a combination of the two (see Figure 5).

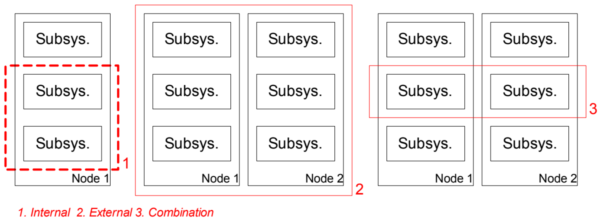

Figure 5. Network Distribution.

The single header approach takes the peer-to-peer concept one step further, from camera-to-camera to subsystem-to-subsystem. This is done without setting up any communication channel. Instead message headers are defined and packets are forwarded through the different subsystems to their destinations. Nodes are addressed by their UIDs and specific subsystems are reached with message types or flags. It is up to the receiving subsystem of a node to define which and how packets are forwarded to the other subsystems of the node.

3.2 WiCa OS

The round-robin task switch system is a software module based on the functionality of TinyOS [11]. The module is controlled from one of the internal timers in the 8051. To implement the functionality of the SW framework, WiCa OS extends this module with an enable/disable function. Theses two functions actually control the HW timer and not the task switching itself. This approach is a simple but robust way to guarantee the kernel execution time.

3.3 Application Execution

Application execution is divided in two parts: the low-level part executed in the IC3D due to its massive parallel processor array; and a high-level part executed in the 8051 that sorts the data, interprets the result, and passes important information to the network.

To provide a good base for these types of applications the SW framework offers three types of internal communication between the IC3D and the 8051, illustrated in Figure 6.

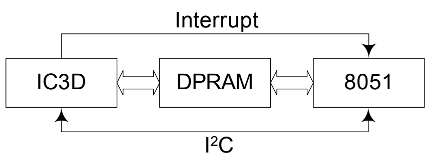

Figure 6. Application Communication.

The DPRAM is a shared data area where the information exchange between the IC3D and the 8051 occurs. The SW framework provides an easily used interface for the DPRAM. The interrupt is used for the IC3D to notify the

[1] Every WiCa as part of a network is called a ”node”.

application code in the 8051 about special events. The framework around the interrupt service routine is provided by the template, but the functionality is left out for the application developer. The I^2C bus is typically used to synchronize the code in the IC3D and the code in the 8051. A low-level I^2C driver is provided to abstract the HW from the application.

3.4 PC-Side Control Environment

The WiCap (Wireless Camera Programmer) is a PC side interface to the WiCas. It consists of a win32 application written in C++ and the same communication HW (AG) as used in the WiCas. The HW for the PC comes as either a USB-stick or a UART based PCB. The system provides functionality for re-programming of WiCas, administration of the network, management of on-board EEPROMs and application control.

Basically the WiCap translates a command into a sequence of packets/messages and delivers them to the network thru the communication HW. The extra HW runs the exact same software module for communication as the WiCas.

4. Results

In this paper we have presented the WiCa software stack, which is based on novel communication approach. To share resources in wireless smart camera networks we used a universal packet-based approach. The packet header is common for all WiCa subsystems and therefore there is no need for additional SW interlayer interfaces. This saves power and increases performance.

We also developed a universal application template, which provides all necessary mechanisms of interaction between different WiCas and between different subsystems of one WiCa and even between different subsystems of differents WiCas.

The only restriction of the WiCa communication is that nothing besides a star-type network is supported. Two WiCa nodes do not directly have to see each other, but the coordinator must see all nodes. This puts the restriction on the maximum distance between the two nodes.

5. Future Work

To increase the communication range we plan to use the latest AquisGrain II (AG2) the next WiCa generation. AG2 provides a full ZigBee stack with all the consequences.

We are also planning to build a WiCa GUI to more effectively interact with a running PC-application and to be able to browse the memory.

ACKNOWLEDGMENTS

Thanks to Ben Zwaans from 3D Solutions for his support on IC3D.

6. REFERENCES

[1] Kleihorst, R.P. et al., "Xetal: a low-power high-performance smart camera processor," *Circuits and Systems, 2001. ISCAS 2001. The 2001 IEEE International Symposium on* , vol.5, no.pp.215-218 vol. 5, 2001

[2] Wolf, W. Ozer, B. Lv, T. "Smart Cameras as Embedded Systems". COMPUTER -LOS ALAMITOS- 2002, VOL 35; PART 9, pages 48-53.

[3] HH Fatemi, RP Kleihorst, H Corporaal, P Jonker "REAL-TIME FACE RECOGNITION ON A SMART CAMERA". Proceedings of Acivs 2003, Ghent, Belgium, September 2-5, 2003.

[4] Bramberger, M. Brunner, J. Rinner, B. Schwabach, H. "Real-time video analysis on an embedded smart camera for traffic surveillance". Real-Time and Embedded Technology and Applications Symposium, 2004. Proceedings. RTAS 2004. 10th IEEE, pages 174- 181.

[5] Chang Hong Lin; Lv, T.; Wolf, W.; Ozer, I.B., "A peer-to-peer architecture for distributed real-time gesture recognition," *Multimedia and Expo, 2004. ICME '04. 2004 IEEE International Conference on* , vol.1, no.pp. 57- 60 Vol.1, 27-30 June 2004.

[6] Wolf, W.; Ozer, B.; Lv, T., "Architectures for distributed smart cameras," *Multimedia and Expo, 2003. ICME '03. Proceedings. 2003 International Conference on* , vol.2, no.pp. II- 5-8 vol.2, 6-9 July 2003.

[7] R. Kleihorst, B. Schueler, Alexander Danilin, Marc Heijligers, "Smart Camera Mote With High Performance Vision System", submitted to DSC-06.

[8] http://www.atmel.com/products/8051/

[9] J. Espina, T. Falck, and O. Mülhens. "Network topologies, communication protocols and standards", in Body Sensor Networks, 145-182, Springer, 2006.

[10] R. Kleihorst, A. Abbo, V. Choudhary, B. Schueler, "DESIGN CHALLENGES FOR POWER CONSUMPTION IN MOBILE SMART CAMERAS", COGIS'06—Cognitive Systems with Interactive Sensors, March 15–17, 2006; Paris (France)

[11] http://www.keil.com/support/man/docs/tr51/

[12] Huang Lee and Hamid Aghajan, "Vision-Enabled Node Localization in Wireless Sensor Networks". COGIS'06, March 15–17, 2006; Paris (France)

[13] Laura Savidge, Huang Lee, Hamid Aghajan, Andrea Goldsmith, "Event-Driven Geographic Routing for Wireless Image Sensor Networks". COGIS'06, March 15–17, 2006; Paris (France)