

Chapter 2

RECONFIGURABLE HARDWARE EXPLOITATION IN WIRELESS MULTIMEDIA COMMUNICATIONS

Konstantinos Masselos^{1,2} and Nikolaos S. Voros¹

¹ *INTRACOM S.A., Hellenic Telecommunications and Electronics Industry, Greece*

² *Currently with Imperial College of Science Technology and Medicine, United Kingdom*

Abstract: This chapter presents cases where reconfigurable hardware can be exploited for the efficient realization of wireless multimedia communication systems. The various scenarios described are referring to (a) the DLC/MAC layer and the baseband part of the physical layer of HIPERLAN/2 and IEEE 802.11a WLAN protocols, and (b) the application layer of a sophisticated personal device. The goal of this chapter is to provide an insight on the advantages reconfigurable hardware may bring in real life applications.

Key words: Reconfiguration, WLAN, application layer, wireless multimedia communications

1. RECONFIGURABLE HARDWARE BENEFITS FROM A SYSTEM'S PERSPECTIVE

The presence of reconfigurable hardware resources in a system can be exploited in two major directions:

- To create space for post-fabrication functional modifications e.g. to upgrade system functionality or for software like bug fixing. Software realizations allow post-fabrication functional modifications, however for complex tasks software realizations might be inefficient. This feature may allow important time-to-market improvement.
- To allow sharing of hardware resources among tasks that are not active simultaneously thus reducing the total area cost of the system. Such

tasks may belong to different modes of operation of a given system, to different applications or standards realized on the same platform or even to time non-overlapping tasks of a single system.

Given an application, tasks that are suitable for realization on reconfigurable hardware are those that may share hardware resources with other tasks over time or are likely to be modified/upgraded in the future and also have high computational complexity (that prevents efficient realization on instruction set processors).

In the rest of this chapter, reconfiguration scenarios are discussed from the wireless communications and multimedia domains. Real life complex systems are used for this analysis namely the HIPERLAN/2 and IEEE 802.11a WLAN systems (covering MAC and physical layers functionality) and the MPEG system (covering the application layer).

2. RECONFIGURATION SCENARIOS FOR HIPERLAN/2 AND IEEE 802.11a WLAN SYSTEMS

In this section reconfiguration scenarios for the HIPERLAN/2 and IEEE 802.11a WLAN systems are discussed. The two systems targeted functionalities cover the DLC/MAC layer and the baseband part of the physical layer.

2.1 HIPERLAN/2 and IEEE 802.11a systems

HIPERLAN/2 [1] is a connection-oriented time-division multiple access (TDMA) system. Physical layer is based on coded OFDM modulation scheme [2]. The physical layer is multi-rate type allowing control of link capability between access point and mobile terminal according interference situations and distance.

The flow graph of the HIPERLAN/2 transmitter is shown in Figure 2-1. The blocks in the inputs and outputs of the different tasks give the input and output rates of the tasks respectively. The input rate of a given task corresponds to the minimum amount of data required for the task to produce a given output (output rate).

The computational complexity and the type of processing of the transmitter tasks are analytically presented in Table 2-1. The analysis of computational complexity is done by estimating the number of required basic operations per output data item in each function. The basic operations include arithmetic, logic and memory read/write operations. It is assumed,

that a processing of transmitted or received data should be possible at a sustained nominal data rate of each physical layer mode. The input and output operations included in this complexity analysis correspond to data coming from previous tasks and being passed to following tasks (in a real implementation these operations are likely corresponding to accesses to data storage locations).

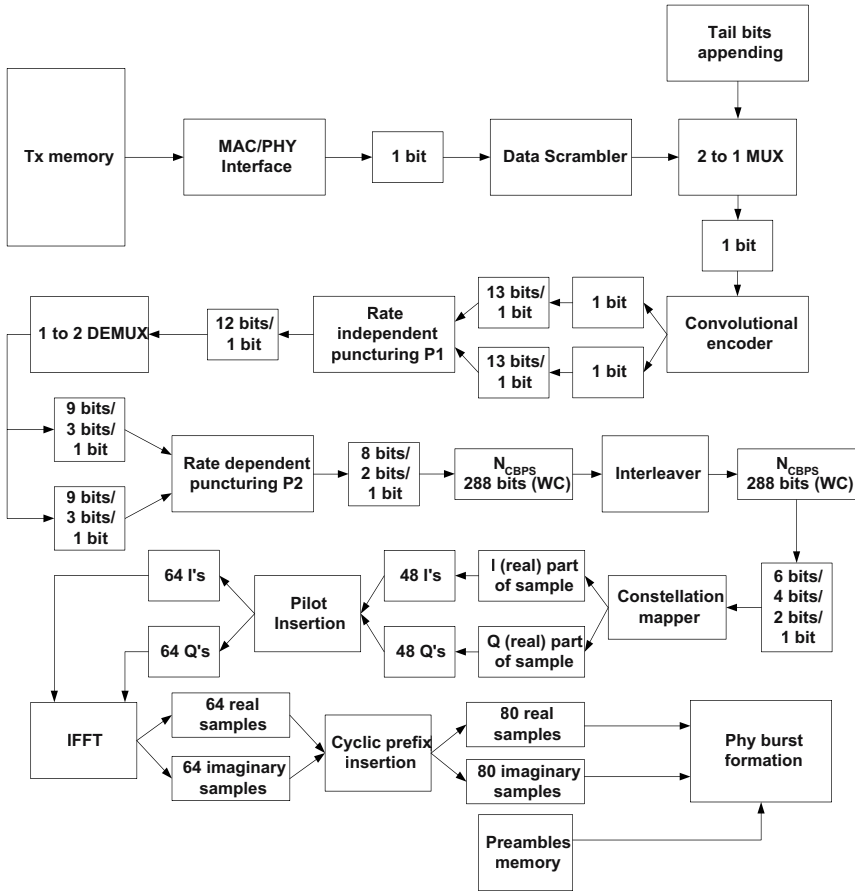


Figure 2-1. HIPERLAN/2 transmitter

From the computational complexity analysis it can be seen that there are some algorithms that generate a constant computational complexity in all physical layer modes. The most important is IFFT that is dominating the overall transmit side complexity in the low bit rate modes. The complexities of channel coding functions are naturally related to the used bit rate.

Task	Type of processing	Computational complexity (MOPS) / PHY mode (Mb/s)						
		6	9	12	18	27	36	54
Scrambling	bit level - shift register, XOR	108	162	216	324	486	648	972
Convolutional encoding	bit level - shift register, XOR	174	261	348	522	783	1044	1566
Puncturing (Rate dependent)	bit level – logic operations	0.31	0.31	0.31	0.31	0.31	0.31	0.31
Puncturing (Rate dependent)	bit level – logic operations	0	33	0	66	105	132	198
Interleaving	Group of bits – LUT accesses	48	48	96	96	192	192	288
Constellation mapping	Group of bits – LUT accesses	30	45	36	54	54	72	90
Pilot insertion	Word level - memory accesses	56	56	56	56	56	56	56
IFFT	Word level – multiplications, additions, memory accesses	922	922	922	922	922	922	922
Cyclic prefix insertion	Word level - memory accesses	72	72	72	72	72	72	72
Sum		1410	1599	1746	2112	2670	3138	4164

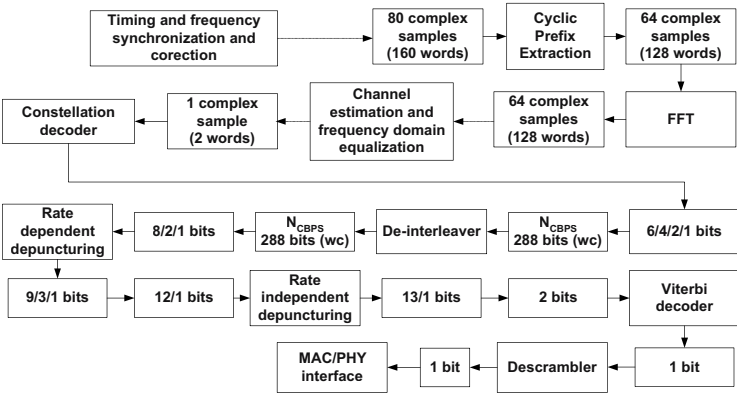


Figure 2-2. HIPERLAN/2 receiver

The flow graph of a reference HIPERLAN/2 receiver is presented in Figure 2-2. The receiver chain of the HIPERLAN/2 is left open by the standard so there is more freedom for algorithm selection for certain blocks such as the timing and frequency synchronization and the channel estimation (different chains of tasks can be adopted for these two generic blocks). The computational complexity and the type of processing of the receiver tasks are analytically presented in Table 2-2.

Table 2-2. Computational complexity of receiver tasks in different physical layer modes

Task	Type of processing	Computational complexity (MOPS) / PHY mode (Mb/s)						
		6	9	12	18	27	36	54
Cyclic prefix extraction	Word level – memory accesses	96	96	96	96	96	96	96
Frequency error correction	Word level – multiplications, additions, memory accesses	208	208	208	208	208	208	208
FFT	Word level – multiplications, additions, memory accesses	922	922	922	922	922	922	922
Frequency domain equalization	Word level – multiplications, additions, memory accesses	132	132	132	132	132	132	132
Constellation demapping	Group of bits – LUT accesses	48	48	240	240	288	288	336
Deinterleaving	Group of bits – LUT accesses	48	48	96	96	192	192	288
Depuncturing (Rate dependent)	bit level – logic operations	0	50	0	99	118	198	297
Depuncturing (Rate independent)	bit level – logic operations	0.16	0.20	0.16	0.20	0.28	0.20	0.20
Viterbi decoding	Bit level I/O – word level additions, comparisons	1170	1755	2340	3510	5265	7020	10530
Descrambling	bit level – shift register, XOR	108	162	216	324	486	648	972
Sum		2732	3421	4250	5627	7707	9704	13781

As it can be deduced, the Viterbi decoding dominates the overall complexity figures in all physical layer modes. It can be also seen that the receiver side processing is up to three times more complex than transmit side processing.

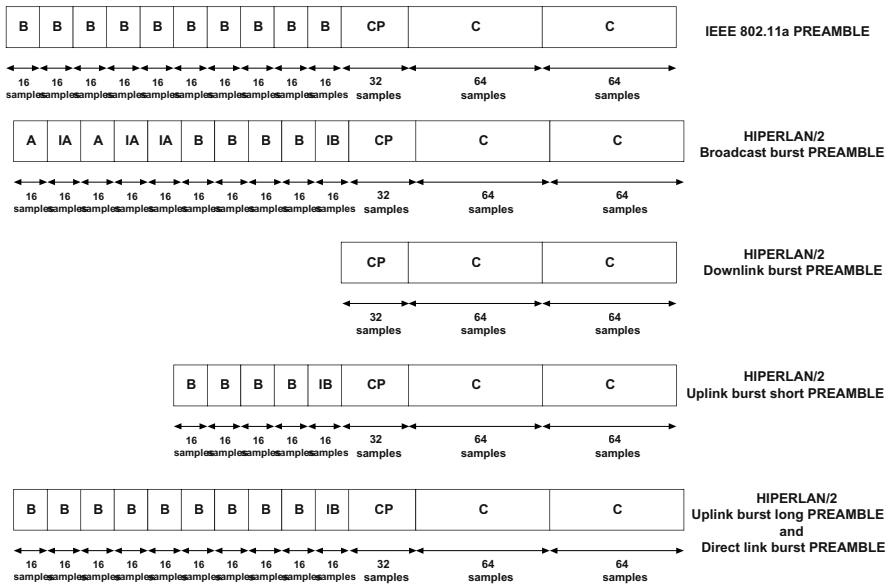


Figure 2-3. IEEE 802.11a and HIPERLAN/2 preambles

The baseband part of the IEEE 802.11a system [3] is almost similar to that of HIPERLAN/2 system. Only some minor differences exist. IEEE 802.11a uses only one preamble sequence (shown in Figure 2-3) of 320 samples. HIPERLAN/2 uses 4 different types of preamble sequences for the different types of PDUs with sizes ranging from 160 samples to 320 samples. The contents of the first half of the PREAMBLE sequences of HIPERLAN/2 are always different to that of IEEE 802.11a. From an implementation point of view this may affect the synchronization block of the receiver.

Different sequences are used by the two systems for the initialization of the (de)scrambler. In IEEE 802.11a the initialization is performed using the first 7 bits of the service field which are always set to zero. In HIPERLAN/2 the initial state of the scrambler is set to pseudo random non-zero 7-bit state determined by the frame counter field in the BCH (first four bits of BCH) at the beginning of the corresponding MAC frame. The initial state is derived

by appending the first four bits of BCH to the fixed binary number $(111)_2$. This difference is small from an implementation point of view.

In the encoder side, IEEE 802.11a supports 1/2, 3/4 and 2/3 code rates while HIPERLAN/2 supports 1/2, 3/4 and 9/16 code rates. Two code rates are in common while each system supports a third different extra one. HIPERLAN/2 applies two puncturing stages (a rate independent one followed by a rate dependent one) while IEEE 802.11a applies a single puncturing stage. The puncturing patterns applied by the two systems to achieve the different code rates are presented in Figure 2-4 (no puncturing pattern is required for 1/2 code rate). The difference from an implementation point of view is small.

The combinations of modulation, coding rate and achieved nominal bit rate (physical modes of operation) supported by IEEE 802.11a and HIPERLAN/2 are presented in Table 2-3. Six modes of operation are common, IEEE 802.11a supports two extra modes while HIPERLAN/2 supports one extra mode. From an implementation point of view the number of modes of operation supported affects the modem controller from which the modem control words are issued.

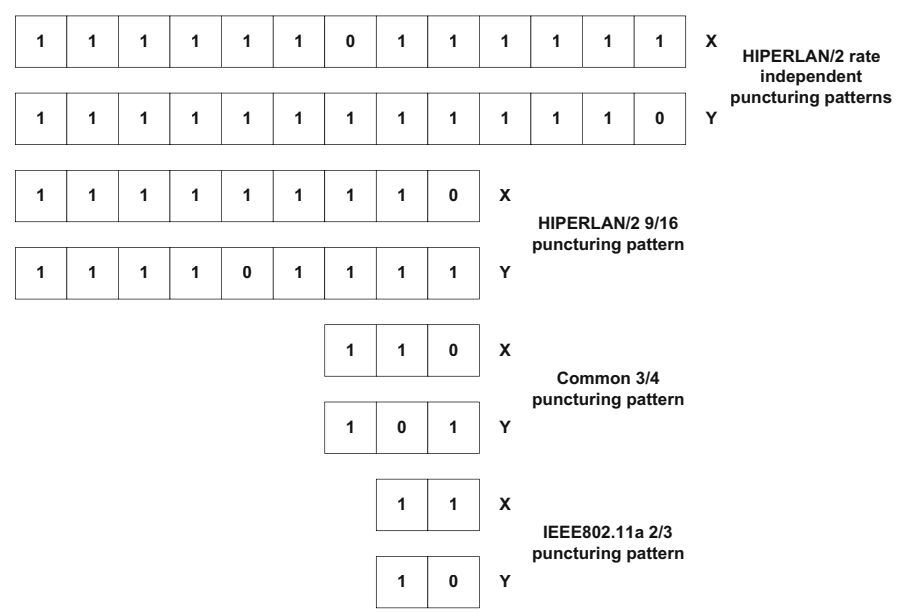


Figure 2-4. Puncturing patterns used by IEEE 802.11a and HIPERLAN/2

The MAC frame duration of the HIPERLAN/2 is fixed to 2 ms. The HIPERLAN/2 MAC frame structure described in Figure 2-5 comprises time

slots for broadcast control (BCH), frame control (FCH), access feedback control (ACH) and data transmission in downlink (DL), uplink (UL) and directlink (DiL) phases, which are allocated dynamically depending on the need for transmission resources. A mobile terminal (MT) first has to request capacity from the access point (AP) in order to send data. This can be done in the random access channel (RCH), where contention for the same time slot is allowed. Downlink, uplink and directlink phases consist of two types of PDUs. The long PDUs have a size of 54 bytes and contain control or user data. The payload is 49.5 bytes and the remaining 4.5 bytes are used for the PDU Type (2 bits), a sequence number (10 bits, SN) and cyclic redundancy check (CRC-24). Long PDUs are referred to as the long transport channel (LCH). Short PDUs contain only control data and have a size of 9 bytes. They may contain resource requests, ARQ messages etc and they are referred to as the short transport channel (SCH). A physical burst is composed of the PDU train payload and a preamble and is the unit to be transmitted via the physical layer.

Table 2-3. Physical modes of operation of IEEE 802.11a and HIPERLAN/2

Modulation	Coding Rate R	Nominal bit rate (Mbit/s)	Coded bits per OFDM symbol
BPSK	1/2	6	48
BPSK	3/4	9	48
QPSK	1/2	12	96
QPSK	3/4	18	96
16 QAM (HL/2 only)	9/16	27	192
16 QAM (IEEE 802.11a only)	1/2	24	192
16 QAM	3/4	36	192
64 QAM	3/4	54	288
64 QAM (IEEE 802.11a only)	2/3	48	288

The structure of the IEEE 802.11a PPDU frame is described in Figure 2-6. The header contains information about the length of the exchanged data and the transmission rate. The RATE field conveys information about the type of the modulation and the coding rate used in the rest of the packet. The LENGTH field takes a value between 1 and 4095 and specifies the number of bytes to be exchanged (PSDU). The six tail bits are used to reset the convolutional encoder and to terminate the code trellis in the decoder. The first 7 bits of the service field are set to zero and are used to initialise the (de)scrambler. The remaining 9 bits are reserved for future use.

The pad bits are used to ensure that the number of bits in the PPDU frame maps to an integer number of OFDM symbols. A cyclic redundancy check (CRC-32) is included in the IEEE 802.11a PSDU.

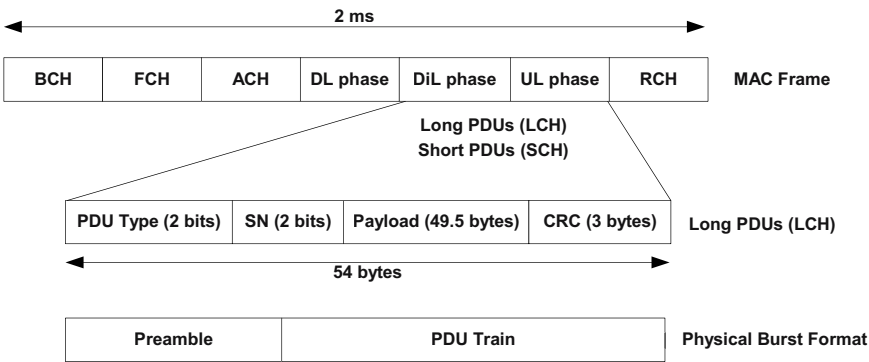


Figure 2-5. HIPERLAN/2 MAC frame, Long PDU and Physical Burst format

An important issue is that the transmission duration (TXTIME) for a PPDU frame in IEEE 802.11a is not fixed but a function of LENGTH field as shown in the following equation:

$$TXTIME = T_{PREAMBLE} + T_{SIGNAL} + T_{SYM} \times Ceiling((16 + 8 \times LENGTH + 6) / N_{DBPS}) \tag{1}$$

where N_{DBPS} is the number of data bits per symbol and can be derived from the DATARATE parameter. From an implementation point of view this fact imposes a strict timing requirement to the MAC/PHY interface for the decoding of the SIGNAL symbol in order to determine the number of OFDM symbols to be exchanged.

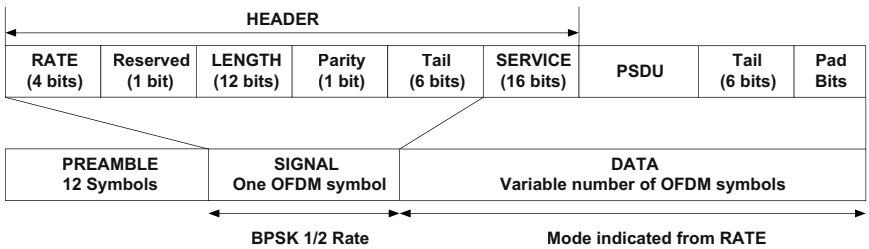


Figure 2-6. IEEE 802.11a PPDU frame format

The major differences between IEEE 802.11a and HIPERLAN/2 systems occur in the MAC sublayer. In HIPERLAN/2 the medium access is based on a TDD/TDMA approach. The control is centralized to an AP, which informs the MTs at which point in time in the MAC frame they are allowed to transmit their data. IEEE 802.11a uses a distributed MAC protocol based on Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA).

2.2 WLAN Reconfiguration scenarios

Some reconfiguration scenarios for the MAC and baseband parts of the HIPERLAN/2 and IEEE 802.11a WLAN systems are described in this section. HIPERLAN/2 and IEEE 802.11a baseband processing algorithms are quite simple as far as control flow is concerned and their functionality does not depend in principle on the physical layer mode that is used in transmission or reception. The baseband processing computational complexity depends very much on the used physical layer mode in the transmission or reception.

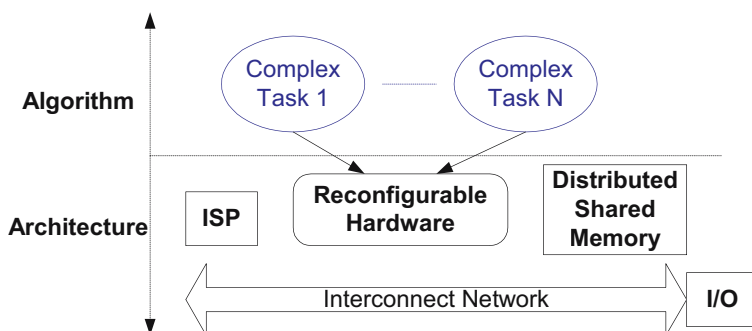


Figure 2-7. Realization on a highly flexible platform

The most computationally complex tasks are the Viterbi decoding and the FFT on the receiver side and the IFFT in the transmitter side. Assuming a highly flexible implementation using instruction set processors (ISP) and reconfigurable hardware (alongside interconnect, memory, I/Os etc.) these tasks should be assigned to reconfigurable hardware (for increased speed and reduced power). This scenario is illustrated in Figure 2-7. However almost no flexibility is required for these tasks on a stand-alone basis (no different candidate implementation choices exist). If ASIC blocks were included in the target implementation platform these tasks should be preferably moved to them.

Reconfigurable hardware resources can be shared among baseband processing tasks that are not active simultaneously. This may lead to silicon area optimization (taking into consideration reconfiguration related overheads). This scenario is described in Figure 2-8. For example under a half duplexing scenario the transmitter and the receiver will not be active simultaneously. In this case, tasks of the transmitter and the receiver may share the same reconfigurable hardware resources.

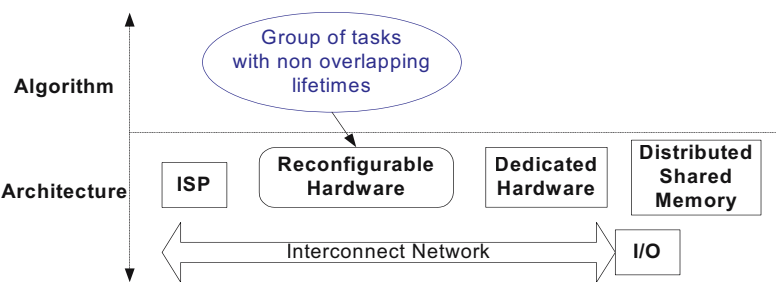


Figure 2-8. Reconfigurable hardware sharing among tasks with non-overlapping lifetimes

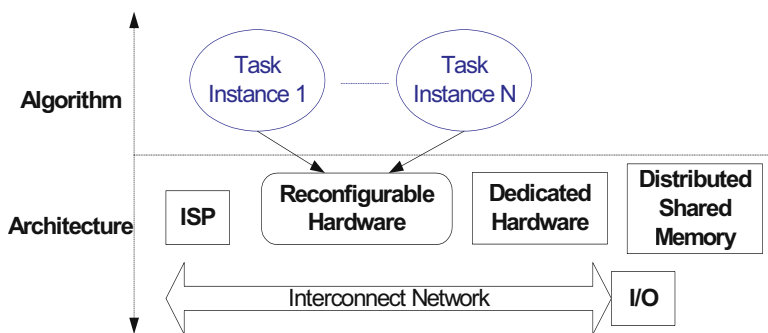


Figure 2-9. Realization of different algorithmic instances of the same task on reconfigurable hardware

Certain tasks in the receiver chain of the baseband processing allow different algorithmic implementations with different trade-offs between algorithmic performance and computational complexity (e.g. channel estimation). Lower algorithmic performance requirements (e.g. SNR, BER) may allow the use of less sophisticated and computational complex algorithmic instances leading to improved implementation efficiency (speed,

power). Furthermore realization of different algorithmic instances for the same task in a given system may be beneficial e.g. allowing adaptation to different operating conditions. Such tasks are good candidates for implementation on reconfigurable hardware (with their different instances sharing the same reconfigurable hardware resources) if their complexity is high (preventing efficient realization on instruction set processors). This scenario is described in Figure 2-9.

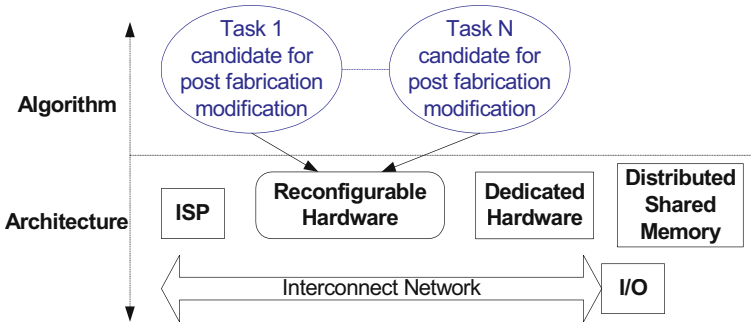


Figure 2-10. Post shipment modification scenario

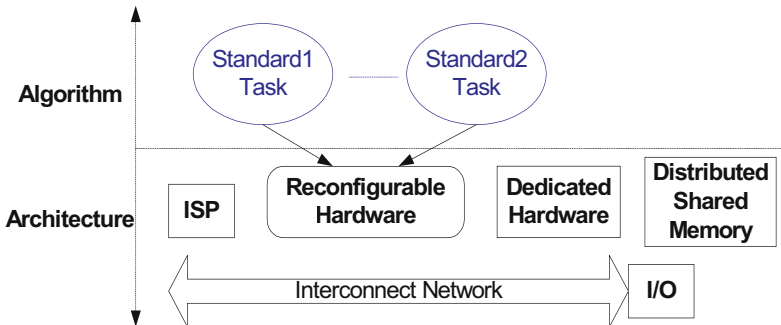


Figure 2-11. Multi-standard realization scenario

Another opportunity for reconfigurable hardware exploitation is towards post-shipment modification/enhancement of the system's functionality (e.g. with more sophisticated realizations of certain tasks). Baseband processing tasks that are candidates for being upgraded are those that are left open by the standard. This scenario is described in Figure 2-10.

More opportunities for reconfiguration and reconfigurable hardware sharing exist in the case of realization of multiple standards on the same reconfigurable implementation platform. This scenario is described in Figure 2-11. Let assume a HIPERLAN/2 – IEEE 802.11a dual standard

realization with the two systems not being active simultaneously. Given that the major differences between the two standards are in the MAC layers reconfigurable hardware can be used for the realization of the most complex and performance demanding parts of the MAC layers (and the MAC to baseband interfaces) of the two systems.

3. RECONFIGURATION SCENARIOS AT THE APPLICATION LAYER

As portable devices become more powerful, it also becomes possible to run more computationally intensive services on these appliances. Due to the increasing flexibility requirements that are imposed by these applications, the devices need to be highly adaptable to the running applications. At the other hand, efficient realizations of these applications are required, especially in the resources they use during deployment, where power consumption must be traded against perceived quality of the application. To be able to realize a variety of applications or services, the implementation platform needs to be highly adaptable.

Assume a wireless communication terminal as is shown in Figure 2-12, which consists out of instruction set processors (ISP) and reconfigurable hardware that are connected to a common interconnect network and to memory. This device is powerful enough to support various applications, including video. Because of the high computational demand of such a video application, it will be run on the reconfigurable hardware (see Figure 2-12) as that part can be configured for optimal performance for a given application.

When the user decides to view the video in a small window and to start up a 3D game, the situation changes. Then the video application can be run with much less resources, while the game becomes the most computationally intensive application. This means that this 3D game will need to be run on the reconfigurable hardware. To enable that, the video application is moved to run further in software on an instruction set processor (ISP). The hardware is then reconfigured for the 3D game and that application is started (see Figure 2-13).

By moving the video application to software and running it in a smaller window also implies that a lower data rate can be used on the wireless terminal interconnect. This means that the wireless appliance should send back to the server that a lower resolution (and thus a lower bit-rate) is allowed for the video application. The application quality as perceived by the user is still satisfying.



Figure 2-12. A video application is running on the reconfigurable hardware



Figure 2-13. A 3D application is running on the reconfigurable hardware, while the video application continues in a reduced window and on a software processor

From the application scenario above, it is clear that it must be possible to run many different applications on the reconfigurable hardware. This means that general reconfigurable hardware is needed, in contrast to incorporating dedicated hardware blocks, like FFT processor, FIR filter etc. Also we notice that applications are very different in nature, as already described in the case of video streaming and interactive 3D applications. A selection of the

reconfiguration characteristics is also based on general characteristics of the multi-media applications and on the usage scenario above.

Requirements on reconfiguration time are modest: because reconfiguration is user-initiated, fast reconfiguration times (< 1 msec) are not needed. When e.g. switching a video application from hardware to software, it is not important that a numbers of frames are not decoded. As soon as the application is running in software, it decodes the next incoming frame.

Requirements on the reconfiguration granularity are complicated by the unknown nature of the application, the granularity should be fine enough so that for each application an optimal implementation in reconfigurable hardware is possible. However due to power requirements, word level coarse grain reconfiguration is more appropriate than bit-level reconfiguration. This is especially the case when the word-lengths are matched to the application at hand.

Table 2-4. Operational power requirements for MPEG2 video decoding

MPEG-2 MP@ML Decoder			
<i>Function</i>	<i>MOPS</i>	<i>Input</i>	<i>Output</i>
Bitstream parsing and VLD	12	4	40
Dequantization and IDCT	105	40	70
Motion Compensation	273	70	70
YUV to RGB color conversion	299	70	35
Total	689	184	215

Table 2-5. Operational power requirements for a 3D application

Quality	CPU time	#triangles	#pixels	Architecture
31 dB	40 ms	5000	5 %	SW
31 dB	2 ms	5000	5 %	HW
25 dB	70 ms	5000	19%	SW
30 dB	80 ms	8000	19%	SW
43 dB	118 ms	17500	19 %	SW
43 dB	21 ms	17500	19 %	HW

To summarize the requirements on applications, it is not only emphasized that different applications must be able to run on the wireless LAN platform, but also that they can have huge computational demands for which dedicated or reconfigurable hardware is needed. To have an indication of the required operational power, we refer to literature [4, 5] the results of which are summarized in Table 2-4 for MPEG2 and in Table 2-5 for a 3D application. In the latter application the CPU time, and thus the frame rate, is closely

related to the required quality (application QoS) but also depends on the architecture, be it a hardware or a software realization.

REFERENCES

1. ETSI (2000), Broadband Radio Access Networks (BRAN); HIPERLAN type 2; Physical (PHY) layer, v1.2.1
2. Van Nee R, Prasad R (1999) OFDM for Mobile Multimedia Communications. Boston: Artech House
3. IEEE Std 802.11a/D7.0 (1999) Part 1: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: High Speed Physical Layer in the 5 GHz Band
4. Zhou CG, Kabir I, Kohn L, Jabbi A, Rice D, Hu XP (1995) MPEG video decoding with the UltraSPARC visual instruction set. In: Proceedings of the 40th IEEE Computer Society International Conference, pp. 470—477
5. Lafruit G, Nachtergaele L, Denolf K, Bormans J (2000) 3D Computational Graceful Degradation. In: Proceedings of ISCAS—Workshop and Exhibition on MPEG-4, vol. 3, pp. 547-550

System Level Design of Reconfigurable Systems-on-Chip

Voros, N.; Masselos, K. (Eds.)

2005, 231 p., Hardcover

ISBN: 978-0-387-26103-4