IMPLEMENTING A RECEIVER FOR TERRESTRIAL DIGITAL VIDEO BROADCASTING IN SOFTWARE ON AN APPLICATION-SPECIFIC DSP

M. Hosemann, G. Cichon, P. Robelly, H. Seidel, T. Dräger, T. Richter, M. Bronzel, and G. Fettweis

Vodafone Chair Mobile Communication Systems
Dresden University of Technology
01062 Dresden, Germany

ABSTRACT

Terrestrial Digital Video Broadcasting (DVB-T) is currently beeing introduced in many European countries and planned to supplement or replace current analogue broadcasting schemes in a large part of the world. It is also considered as an additional downlink medium for third generation UMTS mobile phones, where a special variant, DVB-H, is under development. Current DVB-T receivers still are built upon dedicated application specific integrated circuits (ASICs). However, designing ASICs is a tedious and expensive task. We will show that it is possible to implement a DVB-T receiver in software on an application-specific digital signal processor (AS-DSP). We analyze the computational requirements of a DVB-T receiver and investigate its potential for parallization. Further, we present our AS-DSP, the M5-DSP, which is based on a novel architectural and design paradigm, and report on implementing the core algorithms of a DVB-T receiver on it.

1. INTRODUCTION

Terrestrial Digital Video Broadcasting (DVB-T) [1] is currently beeing introduced in many European countries and planned to supplement or replace current analogue broadcasting schemes in a large part of the world [2]. It is operational in the whole U.K. and has replaced analogue broadcasting completely in the area of Berlin in Germany. It is also considered as an additional downlink medium for third generation UMTS mobile phones [3] in case multiple users request the same data, e.g. video streams from a sports event or news. For handheld devices a new standard, DVB-H, is under development [4]. DVB-H is a time-sliced version of DVB-T, where data are not transmitted at all times. Hence, data rates and power consumption are reduced.

Most DVB-T receivers are still stationary devices but the first mobile phone (Samsung SGH-P700) featuring a DVB-T receiver has just been introduced. All these devices are built upon application specific integrated circuits (ASICs) which require large design, verification, and manufacturing efforts and expenses. Furthermore, they can not easily incorporate design changes which are required frequently in evolving standards such as DVB-H. We suggest that application-specific digital signal processors (AS-DSPs) could be a suitable mean to implement flexible software solutions for various transmission standards [5, 6]. In this paper, we demonstrate, how a computationally demanding receiver for DVB-T can be implemented on the M5-DSP, an AS-DSP which we designed for a DVB-T receiver. The M5-DSP was designed using a novel design methodology which allows for automatically generating the DSP cores as presented in [7]. A scaled-down version of the M5-DSP featuring less data paths could also be used as a receiver for DVB-H.

We first introduce DVB-T in section 2, analyze the computational requirements and the potential for parallization. In section 3 we describe the architecture of the M5-DSP and explain implementation results in section 4.

2. TERRESTRIAL DIGITAL VIDEO BROADCASTING

DVB-T is an OFDM-based broadcasting system which allows for employing a single-frequency-network (SFN). Also, multiple video and audio streams can be transmitted over one frequency channel. Both techniques help save frequency resources over analogue broadcasting. The block diagram of a DVB-T transmission system is shown in figure 1 with the transmitter on top and the receiver below. We will focus on the receiver, since only very few transmitters have to be implemented for a broadcasting scheme. Some parameters of DVB-T are summarized in table 1. So far, only ASIC solutions for DVB-T receiver chips are available, e.g. [8, 9]. Also, a professional meassurement receiver employing multiple DSPs, FPGAs, and ASICs has been presented [10].

T. Dräger is now with Signalion GmbH, Dresden, Germany
T. Richter is now with ADIT GmbH, Hildesheim, Germany
This work was sponsored in part by Deutsche Forschungsgemeinschaft (DFG) within SFB358-A6

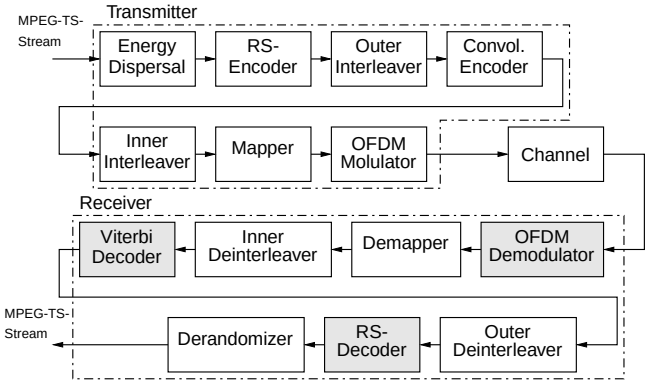

Fig. 1. DVB-T Transmission System

RF Bandwidth	7,8,9 MHz
# of carriers	2k, 8k
Symbol duration T_S	224μs, 896μs
Guard Interval (in T_S)	1/4, 1/8, 1/16, 1/32
Modulation	QPSK, 16, 64 QAM
Code Rate	1/2, 2/3, 3/4, 5/6, 7/8
Net data rate	3..34MBit/s

Table 1. Parameters of DVB-T

2.1. Receiver Structure

The receiver consists of the algorithms depicted in the bottom part of figure 1. The computationally most demanding algorithms are marked in grey. They comprise the OFDM demodulation, Viterbi decoding, and Reed-Solomon decoding.

- The **OFDM demodulation** consists mainly of an FFT over 2k or 8k points, depending on the number of carriers [11]. The critical arithmetic operation in an FFT is the multiply-accumulate (MAC) operation.
- The **Viterbi decoder** requires decoding a 64-state convolutional code at a net data rate of 3 to 34 Mbit/s. The critical operation in the Viterbi decoder is the add-compare-select (ACS) operation for calculating the state metrics.
- A **Reed-Solomon decoder** decoder is used as a second stage of error correction. It requires Galois-field arithmetic over a Galois-field of GF(2^8). The critical operations are the GF-multiply operations.

Since the computational requirements of these algorihms will have a strong influence on the achitecture of an AS-DSP for these algorithms, we will analyze the computational requirements in the following section.

2.2. Computational Requirements

Algorithm	*Required Operations/s*
OFDM Demodulation	720M
Viterbi Trellis Calculation	2765M
Viterbi Traceback	140M
RS-Decoder	11M
Derandomizer	2M
SUM	3668

Table 2. Computational Requirements of a DVB-T Receiver for Mode 8k, 1/4, 16QAM, 2/3, 14.4MBit/s

The operation counts per second for the critical operations of the respective algorithms explained in the previous section are summarized in table 2. It can be seen that the OFDM demodulation and the Viterbi decoder have such high computational requirements that they exceed the computational power of most general purpose DSPs. The parameters for table 2 were choosen for the operation mode which is employed currently in Germany. The requirements for the Viterbi- and RS-decoders can be even higher for higher modulation schemes and data rates.

Since the clock rates for an AS-DSP which could perform all operations of the receiver algorithms serially would be excessive, we will analyze the potential for parallization to find out, if parallization can be used to achieve lower clock rates.

2.3. Parallization

Algorithmus	*Required Operations/s*
OFDM Demodulation	45
Viterbi Trellis Calculation	180
Viterbi Traceback	8
RS-Decoder	2
Derandomizer	0.125
Sum	$\approx$235

Table 3. Computational Requirements of DVB-T Receiver Algorithms for Parallel Processing with $P = 16$ Data Paths

Performing computations in parallel allows for lower clock rates. However, computing algorithms in parallel is only possible if the algorithms bear no inherent data-dependencies which limit parallel computation.

Of the three main algorithms, the FFT [12] and the Viterbi-decoder [13] can be performed in parallel while only the syndrom calculation of the Reed-Solomon-decoder can

be parallized. The Derandomizer can also be computed in parallel but requires very little computational power. For an assumed number of data paths $P = 16$ we estimated the number of operations per second as presented in table 3. This number of data paths was choosen since it results in a target clock rate for the processor of about 200MHz. This is a reasonable clock rate for a standard-cell design with a simple three-stage pipeline, as will elaborated later on.

3. THE M5-DSP

The M5-DSP was designed following a platform-based hardware-software-codesign methodology introduced in [7]. The platform, depicted in figure 2 consists of a fixed control processing part and a scalable signal processing part where the functionality of the data paths can be tailored to suit the application.

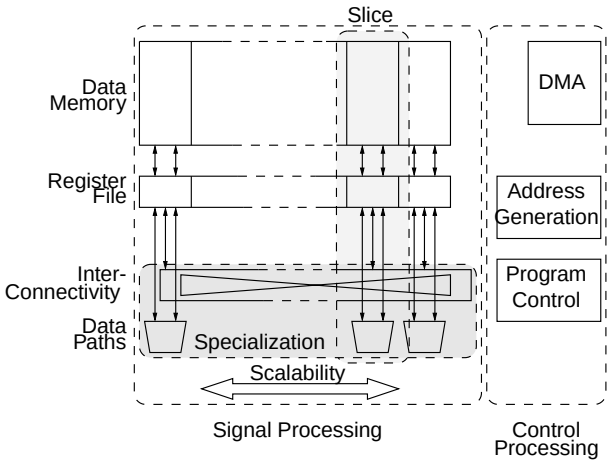

Fig. 2. Platform Architecture

3.1. Architecture

The control processing part consists of the *program control unit* (PCU) which performs operations like jumps, branches, and loops. It features a zero-overhead loop mechanism supporting two nested loops.

Two *address generation units* (AGUs) are available. They serve two purposes: When the processor performs parallel computations on the signal processing part, they generate addresses for the dual-port data memory. When the processor executes serial control code, one AGU still performs address calculations while the other AGU performs microcontroler tasks.

The signal processing part consists of P slices where each slice comprises data memory, a register file, and a data path including an *arithmetic-logic-unit* (ALU), multiplier (MUL), and shifter (BS). An *interconnectivity unit* (ICU) connects the slices with each other and the control part of the processor.

All slices are controlled using the *single-instruction multiple-data* (SIMD) paradigm. This allows for efficiently controlling a large number of slices since only very little control overhead is required.

The M5-DSP features a simple three-stage pipeline with stages fetch, decode, and execute. No stages for read and write-back to and from the register file are required due to the special architecture of the data paths which is outlined in the following section.

3.1.1. Data Paths

The schematic of one of the data paths of the M5-DSP is depicted in figure 3. It consists of a *multiplexer network*, marked light grey at the top, the *functional units* (FUs), marked medium grey, and dedicated *output accumulators*, marked in dark grey, at the bottom.

Each of the FU writes its output into its dedicated output accumulator located below. This allows to operate the FUs fully orthogonally, since no shared registers or busses could pose structural hazards for parallel operation. As input operands, each FU can select the output accumulators of other FUs via its input multiplexers. If each FU can access each other FUs output, the resulting multiplexer network can become quite large. However, the connectivity can be reduced to those connections which are required in the application code. We first wrote our application code assuming full connectivity. As a second step we profiled the application code and removed all connections which were not used. The resulting connectivity can be seen in figure 3. Only connections where an arrow meets the input multiplexer ($\overline{\uparrow}$) are available. We also created a compiler-based tool to extract the required connections directly from the application in C [14]. This multiplexer networks resembles the bypass network of superscalar processors. However, it is controled fully in software and thus needs no control hardware which consumes much area and power in superscalar processors.

Despite the usual FUs ALU, MUL, and BS, we also included a load-store-unit (LST), a unit for register file accesses (REG), and an interconnectivity unit (ICU) into the data path. Including the LST unit into the data path allows for load operations to bypass the register file. This allows for a smaller register file with less ports compared to RISC processor's data paths. The REG unit acts similar to the LST unit but provides three independent accesses to the register file per cycle.

The ALU features special instructions for accelerating the calculation of FFTs and Viterbi decoders, which require a second output accumulator. These features were described in [13] and help reduce the instruction count of FFT and Viterbi-decoder below the instruction counts in table 3.

With these features the number of instructions for one ACS operation can be reduced from 6 down to 2. Hence, the operations count for Viterbi trellis calculation is reduced from 180 down to 60. Also, the multiplier supports Galois-field arithmetic as in [15] for the Reed-Solomon decoder.

3.1.2. Instruction Set Architecture (ISA)

Our M5-DSP features a *very long instruction word* (VLIW) *instruction set architecture* (ISA) which allows for controlling each FU of the data path in each cycle in parallel. The programm control and adress generation unit of the control part of the processor can be controled by *functional instruction words* (FIWs) within the VLIW as well. The size of one VLIW instruction is 170 bits.

3.1.3. Memory

The M5-DSP features a Harvard-style memory architecture with separate data and program memory. The data memory is organized in slices. The required size of the data memory depends on the target application. For DVB-T, the trellis data of the Viterbi decoder require the largest intermediate storage with about 384kbit. Since also state and transition metrics and incoming data need to be stored, we choose a data memory size of 1Mbit. For a high data througput for the FFT we employ dual-port memory, allowing two independent read or write accesses in each cycle.

The program memory needs to store about 1k lines of code as can be found in the implementation results in the following section 4. Each line consists of a VLIW instruction of size 170 bit. Hence, the size of the single-port program memory of the M5-DSP is 180kbit. For implementing more control code of the DVB-T receiver, the programm memory will have to be extented to about 2k lines.

4. IMPLEMENTATION RESULTS

We implemented both hardware and software for the DVB-T receiver algorithms.

4.1. Hardware

For the hardware implementation we created a VHDL model of the processor using our gencore tools as in [16]. The VHDL description was synthesized for a 130nm 8-metal-layer standard-cell library by UMC using Synopsys Design Compiler™. For place and route and back-annotation we used Cadence SoC Encounter™. For ALUs and Multipliers we used Synopsys DesignWare™ components. For memories, SRAM macros by UMC are used.

The resulting layout can be seen in figure 4. The data memories are located closely to the data paths of their respective slices. The ICU, located in between all slices' data paths, connects the data paths. The control processing part (PCU, AGU, and decoder) are hardly visible due to their small size. This confirms the efficiency of the SIMD paradigm.

Clock Rate Our M5-DSP achieves a clock rate of 250MHz which exceeds our initial assumption of 200MHz. The critical path is in the ALU.

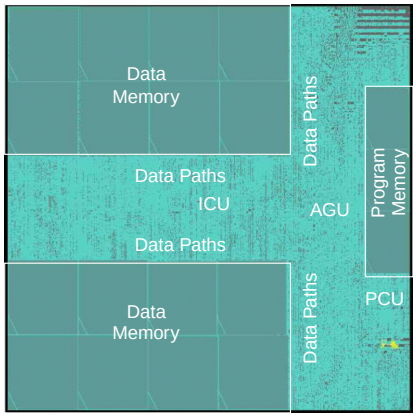

Fig. 4. Layout of M5-DSP

Die Size The die size is dominated by the data memories as can be seen in figure 4 and table 4. The 16 data paths require only a small portion of the design. Due to the high number of routing layers the design achieves a place and route utilization of about 88%.

Component	*Size in mm²*	*Size in %*
Data paths	1.6	17
Program Control	0.1	1
Address Generation	0.2	2
Data Memory	7.1	73
Program Memory	0.7	7
SUM	9.7	100

Table 4. Die Size of the M5-DSP

Power Consumption We estimated the power consumption using Synopsys Power Compiler to be about 300mW. Again, a large portion of this power is consumed in the memories. This compares favorably to the power consumption of an ASIC like the LSI Logic L64782 [9] which consumes about 800mW but also includes parts of the analog front-end. However, it does not provide any flexibilty to accomodate changes in the standard as our M5-DSP does.

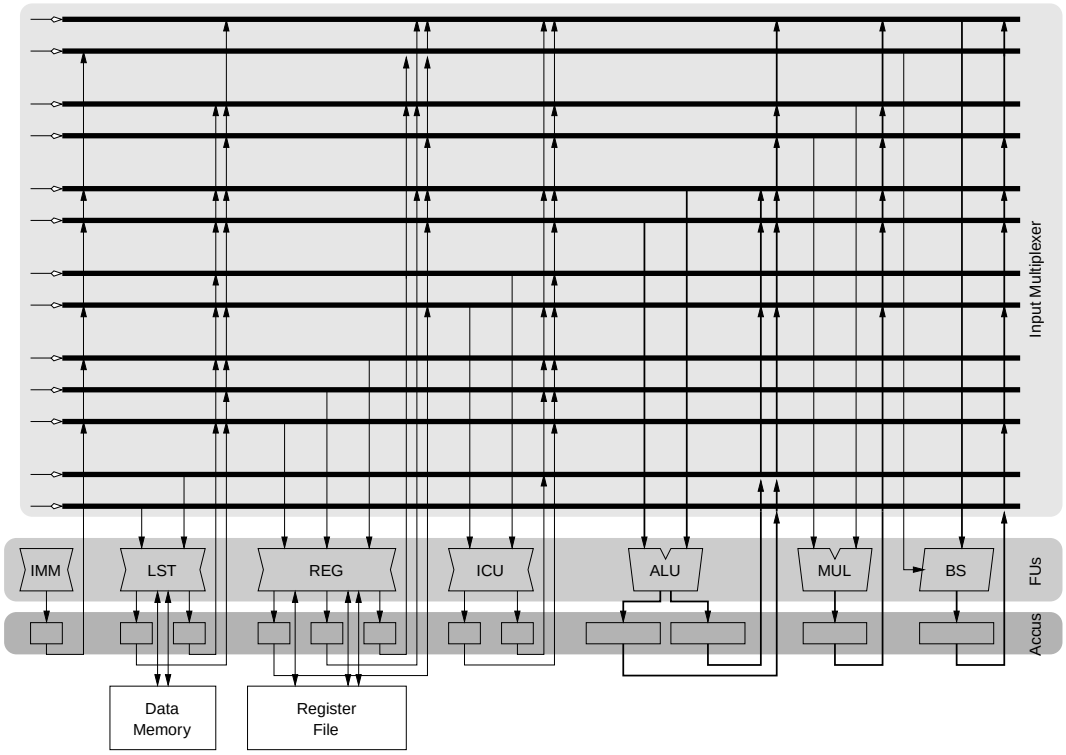

Fig. 3. Schematic of one Data Path of the M5-DSP, Customized for DVB-T Receiver Application

4.2. Software

For the software implementation we created software development tools (Assembler, Linker, Simulator, and Debugger) using the EDGE^{TM} toolsuite by LISAtek/ CoWare and wrote the application software in assembly language. Work on a compiler to speed-up this tedious task is under way. This section shall summarize the performance data of our implemented algorithms.

Algorithm	*Cycles*
FFT 2k/8k	4675/22283
Viterbi Trellis	120768
Viterbi Traceback	11427
RS-Decoder	3700

Table 5. Cycle Counts of Receiver Algorithms

Table 5 shows the cycle counts for computing one FFT symbol, the Viterbi decoder for 16 blocks of 204 bytes, and the Reed-Solomon decoder for one block of 204 bytes. Please note that the application code for the Reed-Solomon decoder is not fully optimized yet. It should be possible to get the cycle count down to about 2000 cycles.

These cycle counts yield a workload of our processor of about 150 MIPS. Considering the clock rate of 250MHz and the still to improve RS-decoder, this leaves enough computational ressources for implementing channel estimation and equalization, interleavers,and control code.

Algorithm	*VLIW Instructions*
FFT 2k/8k	265/310
Viterbi Trellis	224
Viterbi Traceback	113
RS-Decoder	233
SUM	835/880

Table 6. Program Code Size in VLIW Instructions of Receiver Algorithms

The required lines of code and hence program memory for the respective receiver algorithms are shown in table 6.

5. CONCLUSIONS

We presented the implementation of a receiver for DVB-T on an application-specific DSP which we designed for this application. The DSP is capable of performing the computationally intense algorithms in software by means of parallel execution and specially tailored data paths for Viterbi decoding, FFT and Galois-filed multiplication.

The implementation results show that the costs of the DSP are on par with commercially available ASICs but our

M5-DSP also provides flexibility for accomodating future upgrades or changes of the standard by requiring only rather simple software changes instead of redesigns of an ASIC. A scaled-down version of the M5-DSP could also be used for implementing a receiver for the upcoming DVB-H standard.

6. ACKNOWLEDGEMENTS

We would like to acknowledge the help of many students, in particular Rene Habendorf, Rene Beckert, Karsten Todtermuschke and Carsten Köckritz who wrote the assembly code for the receiver algorithms and Thomas Schuster who helped with the VHDL coding of the data paths.

7. REFERENCES

[1] European Telecommunications Standards Institute (ETSI), "Digital Video Broadcasting (DVB); Framing Structure, Channel Coding and Modulation for Digital Terrestrial Television," Jan. 2001, EN 300 744 V1.4.1.

[2] "Digital terrestrial tv standards adoption," map on website, www.dvb.org.

[3] "COMCAR - Communication and Mobility by Cellular Advanced Radio," 2002, www.comcar.de.

[4] DVB Project, "Digital Video Broadcasting (DVB) Transmission System for Handheld Terminals DVB-H," Jun. 2004, DVB Document A081.

[5] Gerhard P. Fettweis, "DSP Cores for Mobile Communications: Where are we going?," in *Proceedings of ICASSP*, München, 21.-24. April 1997, pp. 279–282.

[6] T. Richter, W. Drescher, F. Engel, S. Kobayashi, V. Nikolajevic, M. Weiss, and G. Fettweis, "A Platform-Based Highly Parallel Digital Signal Processor," in *Proc. CICC 2001*, San Diego, USA, 6.-9. May 2001, pp. 305–308.

[7] J.P. Robelly, G. Cichon, H.Seidel, and G. Fettweis, "A HW/SW Design Methodology for Embedded SIMD Vector Signal Processors," *International Journal of Embedded Systems,*, vol. 1, no. 11, Jan. 2005, to appear.

[8] M. Christian, B. Martin, G. Krampl, and F. Kuttner, "0.35um CMOS COFDM Receiver Chip for Terrestrial Digital Video Broadcasting," in *Proc. of Int. Solid State Circuits Conference*, 2000.

[9] LSI Logic, *L64782 Single-Chip COFDM Receiver*, 2002, data sheet.

[10] F. Frescura, E. Sereni, M. Vagheggini, C.Gnudi, and P. Antognoni, "C6000 Based DVB-T Receiver," Tech. Rep., Digilab2000 s.r.l. and Universita di Perugia, 2002.

[11] Michael Speth, Stefan Fechtl, Gunnar Fock, and Heinrich Meyr, "Optimum Receiver Design for OFDM-Based Broadband Transmission — Part II: A Case Study," *IEEE Transactions on Communications*, vol. 49, no. 4, pp. 571–578, April 2001.

[12] Matthias H. Weiss, *Rahmenwerk zur Automatisierung des Entwurfs von digitalen Signalprozessoren (Framework for Auitomated Design of Digital Signal Processors)*, Ph.D. thesis, Dresden University of Technology, June 2003.

[13] Michael Hosemann, Rene Habendorf, and Gerhard P. Fettweis, "Hardware-Software Codesign of a 14.4 Mbit - 64 - State Viterbi Decoder for an Application-Specific Digital Signal Processor," in *Proceedings of IEEE Workshop on Signal Processing Systems 2003 (SIPS'03)*, Seoul, Korea, 27.-29. Aug. 2003, pp. 45–50.

[14] Jie Guo, Michael Hosemann, and Gerhard Fettweis, "Employing Compilers for Determining Architectural Features of Application-Specific DSPs," in *Proceeding of PARELEC'04*, 2004, accepted for publication.

[15] W. Drescher, K. Bachmann, and G. P. Fettweis, "VLSI Architecture for Datapath Integration of Arithmetic Over GF(2^m) on Digital Signal Processors," in *Proceedings of ICASSP'97*, 21.-24. April 1997, pp. 631–637.

[16] Hendrik Seidel and Gerhard P. Fettweiss, "Generated DSP Cores for OFDM-transmission Systems," in *Proc. of SAMOS'04*. 2004, Lecture Notes on Computer Science, Springer Verlag.