

Rapid design of complex DSP cores

J V McCanny¹, D Trainor², Y Hu², T J Ding¹

1. DSiP™ Laboratories, Dept. of Electrical and Electronic Engineering
The Queen's University of Belfast, Belfast, BT9 5AH, Northern Ireland.

2. Integrated Silicon Systems Limited, 29 Chlorine Gardens,
Belfast, BT9 5DL, Northern Ireland.

Methods are presented for the rapid design of Digital Signal Processing chips which are based on the use of hierarchical VHDL libraries. These capture advanced, parameterised DSP architectures ("silicon intellectual property") at various levels of complexity. Resulting designs can be implemented in ASIC, PLD and FPGA technologies and are portable across many silicon foundries. Complex DSP designs are developed in a fraction of the time normally required and results compete very favourably with conventional methods. The approach is illustrated through its application to the design of FFT, DCT and Reed Solomon processors..

1. Introduction

The increasing complexity of modern integrated circuits, coupled with ever decreasing times-to-market, has created major challenges in terms of design complexity, productivity and correctness. This underlies the need to further develop advanced systems on silicon design methodologies. One important approach is through design re-use and through the development of so called "silicon intellectual property" (IP). At present, this tends to take one of two forms:

- (1) specific cores, usually developed in-house or sourced under licence (e.g. RISC processors or programmable DSP processors).
- (2) VHDL descriptions of system level functions captured at the RTL level.

The first category of designs are usually rather inflexible, expensive and tend to require considerable engineering resource in migrating these from process to process. The second approach, which although more portable, often produces silicon solutions which are far from optimal in terms of area, performance and power consumption. The purpose of this paper is to describe new methods for the design of application specific DSP chips which, we believe, incorporate a much higher level of sophistication than current IP based methods. This produces designs which are highly flexible, portable and comparable to ones developed using manual methods. A major attraction is a very significant reduction in typical chip design time.

The approach is based on the use of a series of hierarchical, synthesisable VHDL libraries. Blocks within these libraries are parameterised and incorporate a variety of architectural templates, to cover a wide range of application areas. The resulting algorithm-to-architecture-to-chip design flow mirrors in hardware methods used for DSP software algorithm development e.g. using tools such as SPW or MATLAB.

2. Library Structure

The basic philosophy which underlies the approach adopted is that, whilst from a user point of view, many DSP applications vary widely, experience [1-3] has shown that many typical blocks required in such applications can be constructed from a finite library of pre-designed, parameterised (e.g. in terms of word-lengths) templates. These, in turn, can be used to construct higher level blocks (e.g. arithmetic blocks) which can then be used to construct functions such as FFT devices, FIR filters etc. This approach provides an efficient means for capturing silicon “intellectual property” in a manner (a) which allows non-specialists to rapidly create advanced DSP chips and (b) allows more experienced silicon systems engineers to tailor designs to required specifications.

The current library comprises five sub-libraries (figure 1), with examples of typical contents as shown. Four of these are DSP specific, whilst the fifth (support devices) is included to provide interfaces to other external non-DSP components. In many instances blocks in the higher level libraries have been created using those from a lower level. This means that the optimisation of structures at the lower levels is carried through to higher level functions and this results in designs which are highly efficient in terms of performance, area and power consumption.

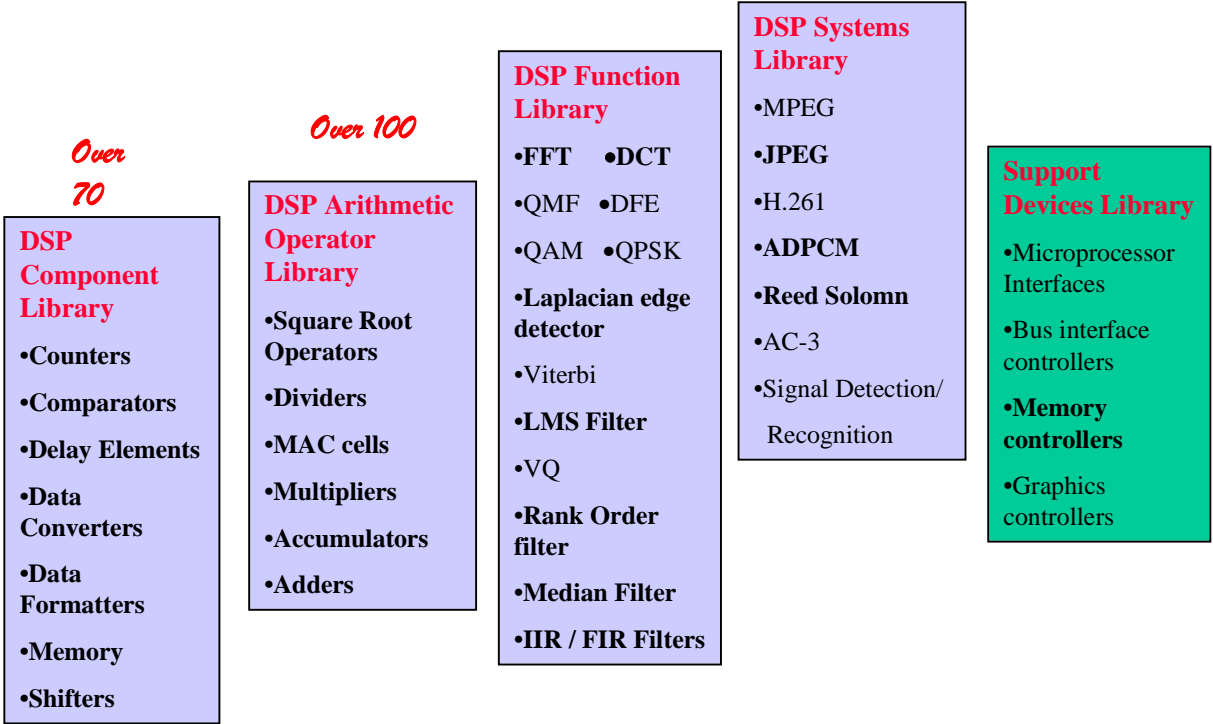


Fig. 1. DSiP™ Library hierarchy

DSP Component Library This contains over 70 blocks covering the range of word and data formats described in section 3. Here only a limited degree of parameterisation is needed e.g. arithmetic type, word length or counter size.

The DSP Arithmetic Library (over 100 optimised blocks) contains operators such as multipliers, adders, dividers and square root processors. Typical parameters include word-length and type of arithmetic used (e.g. carry-save, carry-look-ahead and Wallace tree implementations). Numerous architectural variants of each block allow many different arithmetic and data word formats to be used.

The DSP Function Library incorporates many core DSP functions. A variety of parameters are used, the details being function specific. These provide flexibility in tailoring designs to performance requirements and allow numerous design alternatives to be quickly explored for a given specification. For instance, it is possible to rapidly create many parameterisable FIR and IIR filter circuit types using different permutations of the multiplier/accumulators from the arithmetic operator library. It is also possible to directly use one or other of the pre-defined library filter functions, in which case, such details are transparent to the user. Examples include FIR and IIR filters, filters and FFT's.

DSP System Library This represents the highest level in hierarchy and contains complete system level blocks. Here parameterisation allows different variants to be created. For example, the ADPCM CODEC allows implementations to be produced in which the number of duplex channels required can be changed through the supply of appropriate values [4].

3. Data Word Formats

The type of arithmetic used in many different DSP applications can vary considerably. A broad degree of flexibility has been incorporated through the provision of blocks with different forms of arithmetic. These include both floating and fixed point versions. Each word format is capable of being transmitted and processed in a bit parallel, bit serial or digit serial manner. Bit parallel data tends to give the highest performance, at the expense of silicon area and is useful for high bandwidth applications; bit serial data enables reductions in silicon area for low bandwidth applications; digit serial versions provide flexibility and allow the designer freedom in the matching of bandwidth with circuit architecture. Two floating point formats have been included. These are the IEEE standard floating point number format and an internal custom format. This provides the same accuracy, but enables the development of functions using less silicon area and lower power consumption. Other formats are based on fixed point arithmetic with numerous alternatives available. These include versions based on lsb and msb first arithmetic. The use of msb first arithmetic is highly effective in the design of low latency multipliers, which can be used in high performance, pipelined IIR filters [2], for example.

4. Design examples

FFT Cores

The methods described have been used to create hardware description language generators for the automated design and synthesis of FFT cores. Such generators have been parameterised in terms (a) transform size (b) wordlengths and (c) levels of pipelining. Designs have been synthesised for a range of transform sizes, wordlengths and different types of I/O interfaces. The design in figure 2 performs a 24 bit, 64-point FFT in 1.3 μ s. This is based on a 0.35 μ m CMOS technology. It has a clock rate of around 100 MHz, contains 120,000 gates, has a power consumption of around 1.3W. These figures compare very favourably with a recent manual design [3], but with very significant reductions in design time. The methods described have also been used to develop 2K and 8K complex FFT cores, compliant with European Digital Video Broadcast (DVB) standards. In a 0.35 μ m process, these operate at clock rate of 80MHz. Both devices contain 25,000 gates and incorporate 128K and 256K dual-port RAMs respectively. The 2K design performs an FFT in 0.1ms, the 8K design in 0.4ms. Design times basically depend on the times required to perform synthesis. These typically range from a day to a week. This is significantly less than if conventional design methods had been used.

The methods described have also been used in PLD implementations, a recent example being a 128 point, 12 bit, FFT processor implemented using an Altera FLEX 10K100-3 device. This requires around 65% of available cells, can be clocked at over 30 MHz and performs an FFT in less than 10 μ s.

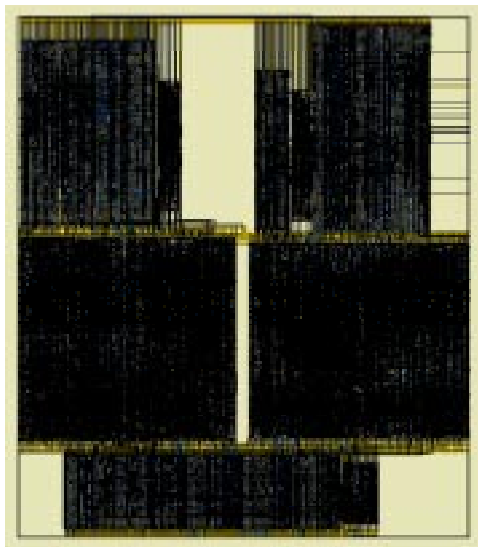


Fig 2. 64 point FFT core

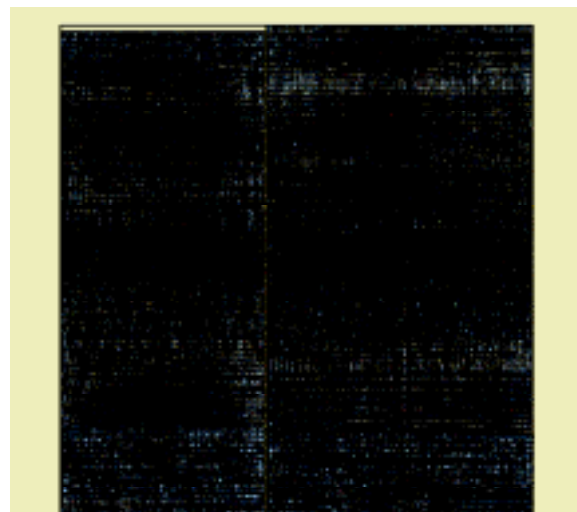


Fig 3. Reed Solomon (DVB compliant) core

Parameterised DCT core A second example is a parameterised DCT core targeted at 8 x 8 DCT implementations in which word-lengths are parameterised. Input values are in the range 8-16 bits, coefficient values in the range 10-18 bits with an internal word-length between 12 and 24 bits. As an example, a 0.35 μ design with 12 bit I/O data requires 56.5K gates, occupies an area of 3.66 x 3.5 mm² and has a critical path of 12ns. This provides a sampling rate of up to 85 Megasamples per second with a power consumption of 0.6W. Design times are similar to those described above. A similar DCT function, with 12 bit I/O and 12 bit coefficients, implemented using an Altera FLEX10K50-3 device has a 50ns critical path (20 MSPS) and uses 4386 logic elements (87% cells).

Reed Solomon Encoders/Decoders Figure 3 shows a 0.35 μ m CMOS Reed Solomon decoder design developed for Digital Video Broadcasting applications ($n = 204$, $k = 188$). This contains 20K gates (plus RAM), operates on 8 bit symbols, has a 20ns critical path and hence processes 50 MSPS (i.e. 400 MBPS). This design has a 361 symbol clock cycle latency. A similar function implemented on an Altera FLEX10K50-3 is capable of processing 13MSPS (104MBPS), using 81% of available logic elements. This has a 564 symbol clock cycle latency. A comparable specification has been achieved using a Xilinx XC4025 chip, with several related Xilinx based designs currently under development.

5. References

- [1] J V McCanny and J G McWhirter "Some Systolic Array Developments in the UK", IEEE Computer, 1987, pp 51-63
- [2] O C McNally, J V McCanny, R F Woods, "The Design of a highly Pipelined Second Order IIR filter chip" Intl. Jour High Speed Electronics and Systems" vol. 4, no 1,1993, pp 65-84
- [3] C Hui, T J Ding, J V McCanny, R F Woods "A New FFT Architecture and Chip Design for Motion Compensation based on Phase Correlation" IEEE Trans. on Solid State Circuits, vol.31, no. 11, pp1751-1761, Nov 1996
- [4] J V McCanny, D Ridge, Y Hu, J Hunter "Hierarchical VHDL Libraries for DSP ASIC Design", Proc. of the IEEE Intl. Conf. on Acoustics Speech and Signal Processing, Munich, April 1997.