ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

Design and Analysis of Low Power Digital Signal Processor Architechture for WSN Using Folded Tree

A.Aayathullah [1], P.SaravanaKumar [2], A.Sathish Kumar [3] , S.Saravanan [4]

P.G. Student, Department of ECE, MahaBarathi Engineering College, Chinnasalem, Tamil Nadu , India[1]

Assistant Professor, Department of ECE, MahaBarathi Engineering College, Chinnasalem, Tamil Nadu, India[2]

Assistant Professor, Department of ECE, MahaBarathi Engineering College, Chinnasalem, Tamil Nadu, India[3]

Professor & Head, Department of EEE, Muthyammal Engineering college, Rasipuram, Tamil Nadu, India[4]

ABSTRACT: Radio communication exhibits the highest energy consumption in wireless sensor nodes. Given their limited energy supply from batteries or scavenging, these nodes must trade data communication for on-the-node computation. Currently, they are designed around off-the-shelf low-power microcontrollers. But by employing a moreappropriate processing element, the energy consumption can be significantly reduced. My paper describes the design and implementation of the newly proposed folded-tree architecture for on-the-node data processing in wireless sensor networks, using parallel prefix operations and data locality in hardware. Measurements of the silicon implementation show an improvement of 10–20× in terms of energy as compared to traditional modern micro-controllers found in sensor nodes. This can be visualized as a binary tree of processing elements (PEs) across which input data flows from the leaves to the root. This topology will form the fixed part of our approach, but in order to serve multiple applications, flexibility is also required. The tree-based data flow will, therefore, be executed on a data path of programmable PEs, which provides this flexibility together with the parallel prefix concept.

I. INTRODUCTION

Since radio transmissions are very expensive in terms of energy, they must be kept to a minimum in order to extend node lifetime. Data communication must be traded for on-the-node computation to save energy, so many sensor readings can be reduced to a few useful data values. Application-Specific: A “one-size-fits-all” solution does not exist since a general purpose processor is far too power hungry for the sensor node’s limited energy budget. ASICs, on the other hand, are more energy efficient but lack the flexibility to facilitate many different applications. Apart from the above characteristics of WSNs, two key requirements for improving existing processing and control architectures can be identified.

Prefix operations can be calculated in a number of ways, but chose the binary tree approach because its flow matches the desired on the-node data aggregation. This can be visualized as a binary tree of processing elements (PEs) across which input data flows from the leaves to the root. This topology will form the fixed part of our approach, but in order to serve multiple applications, flexibility is also required. The tree-based data flow will, therefore, be executed on a data path of programmable PEs, which provides this flexibility together with the parallel prefix concept.

Pulse modulation schemes aim at transferring a narrowband analog signal over an analog baseband channel as a two-level signal by modulating a pulse wave. Some pulse modulation schemes also allow the narrowband analog signal to be transferred as a digital signal with a fixed bit rate, which can be transferred over an underlying digital transmission system, for example some line code. These are not modulation schemes in the conventional sense since they are not channel coding schemes, but should be considered as source coding schemes, and in some cases analog-to-digital conversion techniques.

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

Related work

Design Of Data Processing Architecture For Wsn Nodes Wireless sensor nodes require low-energy components given their limited energy supply from batteries or scavenging. Currently they are designed around off-the-shelf low power micro-controllers for on-the-node processing. However, by employing more appropriate hardware the energy consumption can be significantly reduced. This paper identifies that many WSN applications employ algorithms which can be solved by using parallel prefix-sums. Therefore an alternative architecture is proposed to calculate them energy-efficiently. It consists of several parallel processing elements (PEs) structured as a folded tree. Profiling System C models of the design with Activate SC helps to improve data-locality. Measurements of the fabricated chip confirm an improvement of 10–20x in terms of energy as compared with traditional MCUs found in sensor nodes.

Exploring The Processor And Isa Design For Wireless Sensor Network Applications says Power consumption, physical size and architecture design of sensor node processors have been the focus of sensor network research in the architecture community. What lies at the foundation for these researches is the hardware level design which determines the boundaries for achievable utility and performance. Architecture design and evaluation, however, cannot be accomplished independent of the applications and software that run on these sensor nodes. On one hand some researchers have proposed architectures that can cater to a variety of application classes while trading off on some performance improvements. On the other hand, a set of application-specific architectures have been proposed which perform certain operations extremely well but are not versatile enough to run a variety of applications. It provides a design space exploration and optimizations platform to characterize the processor and ISA design tailored for a particular application or a class of applications. Here collect a wide variety of sensor network applications to create a comprehensive benchmark suite called the WiSeNBench. Then present a careful profiling of these benchmark applications using an ARM simulator to identify some of the key characteristic behaviors. This also opens up avenue for a possible re-look at the classes of applications that could be supported on next-generation sensor networks and efficient architectural designs to enable these applications.

Parallel Prefix (Scan) Algorithms for MPI describe and experimentally compare three theoretically well-known algorithms for the parallel prefix (or scan, in MPI terms) operation, and give a presumably novel, doubly-pipelined implementation of the in-order binary tree parallel prefix algorithm. Bidirectional interconnects can benefit from this implementation. Present results from a 32 node AMD Cluster with Myrinet 2000 and a 72-node SX-8 parallel vector system. On both systems observe improvements by more than a factor two over the straight-forward binomial-tree algorithm found in many MPI implementations. Also discuss adapting the algorithms to clusters of SMP nodes.
Survey Of Hardware Systems For Wireless Sensor Networks says Wireless sensor networks have been gaining interest as a platform that changes how interact with the physical world. Applications in medicine, military, inventory management, structural and environmental monitoring, and the like can benefit from low-power wireless nodes that communicate data collected via a variety of sensors. Current deployments of wireless sensor networks (WSN) rely on off-the-shelf commodity-based microcontrollers, but the un optimized energy consumption of these systems can limit the effective lifetimes. Ideally, researchers would like to deeply embed wireless sensor network nodes in the physical world, relying on energy scavenged from the ambient environment. This paper provides a survey of ultra low power processors specifically designed for WSN applications that have begun to emerge from research labs, which require detailed understanding of tradeoffs between application space, architecture, and circuit techniques to implement these low-power systems.

An Accelerator Based Wireless Sensor Network Processor In 130nm CMOS Networks of ultra-low-power nodes capable of sensing, computation, and wireless communication have applications in medicine, science, industrial automation, and security. Over the past few years, deployments of wireless sensor networks (WSNs) have utilized nodes based on off-the-shelf general purpose microcontrollers. Reducing power consumption requires the development of System on-chip (SoC) implementations that must provide both energy efficiency and adequate performance to meet the demands of the long deployment lifetimes and bursts of computation that characterize WSN applications. This work takes a holistic approach and, thus, studies all layers of the design space, from the applications and architecture, to process technology and circuits. This paper introduces the emerging application space of wireless sensor networks and describes the motivation and need for custom system architecture. The proposed design fully embraces the accelerator-

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

based computing paradigm, including acceleration for the network layer (routing) and application layer (data filtering). Moreover, the architecture can disable the accelerators via VDD-gating to minimize leakage current during the long idle times common to WSN applications. Have implemented system architecture for wireless sensor network nodes in 130nm CMOS. It operates at 550 mV and 12.5 MHz. Our system uses 100x less power when idle than a traditional microcontroller, and 10-600x less energy when active.

Tinybench: The Case For A Standarized Benchmark Suite Or Tinyos Based Wireless Sensor Network Devices The growing wireless sensor network research community lacks a standard method for evaluating hardware platforms. Traditional benchmark suites do not sufficiently address the needs of sensor network designers. This work provides motivation for a benchmark suite and details an approach for benchmarking Tiny OS compatible hardware. To aid the development of future hardware architectures propose the creation of a standard single node benchmark suite, based on both real applications and "stress marks." Present sample benchmark results and call for further work in this area.

II. SYSTEM MODEL

VHDL (VHSIC Hardware Description Language):

VHDL (VHSIC hardware description language) is a hardware description language used in electronic design automation to describe digital and mixed-signal systems such as field-programmable gate arrays and integrated circuits.VHDL was originally developed at the behest of the U.S Department of Defense in order to document the behavior of the ASICs that supplier companies were including in equipment. That is to say, VHDL was developed as an alternative to huge, complex manuals which were subject to implementation-specific details. The idea of being able to simulate this documentation was so obviously attractive that logic simulators were developed that could read the VHDL files. The next step was the development of logic synthesis tools that read the VHDL, and output a definition of the physical implementation of the circuit. Because the Department of Defense required as much of the syntax as possible to be based on Ada, in order to avoid re-inventing concepts that had already been thoroughly tested in the development of Ada, VHDL borrows heavily from the Ada programming language in both concepts and syntax. The initial version of VHDL, designed to IEEE standard 1076-1987, included a wide range of data types, including numerical (integer and real), logical (bit and boolean), character and time, plus arrays of bit called bit_vector and of character called string. The updated IEEE 1076, in 1993, made the syntax more consistent, allowed more flexibility in naming, extended the character type to allow ISO-8859-1 printable characters, added the xnor operator, etc. Minor changes in the standard (2000 and 2002) added the idea of protected types (similar to the concept of class in C++) and removed some restrictions from port mapping rules.In addition to IEEE standard 1164, several child standards were introduced to extend functionality of the language. IEEE standard 1076.2 added better handling of real and complex data types. IEEE standard 1076.3 introduced signed and unsigned types to facilitate arithmetical operations on vectors. IEEE standard 1076.1 (known as VHDL-AMS) provided analog and mixed-signal circuit design extensions. Some other standards support wider use of VHDL, notably VITAL (VHDL Initiative Towards ASIC Libraries) and microwave circuit design extensions. In February 2008, Accellera approved VHDL 4.0 also informally known as VHDL 2008, which addressed more than 90 issues discovered during the trial period for version 3.0 and includes enhanced generic types. In 2008, Accellera released VHDL 4.0 to the IEEE for balloting for inclusion in IEEE 1076-2008. The VHDL standard IEEE 1076-2008 was published in September 2008.

Design

VHDL is commonly used to write text models that describe a logic circuit. Such a model is processed by a synthesis program, only if it is part of the logic design. A simulation program is used to test the logic design using simulation models to represent the logic circuits that interface to the design. This collection of simulation models is commonly called a test bench. VHDL has constructs to handle the parallelism inherent in hardware designs, but these constructs (processes) differ in syntax from the parallel constructs in Ada (tasks). Like Ada, VHDL is strongly typed and is not case sensitive. In order to directly represent operations which are common in hardware, there are many features of VHDL which are not found in Ada, such as an extended set of Boolean operators including nand and nor. VHDL also allows arrays to be indexed in either ascending or descending direction; Both conventions are used in hardware, whereas in Ada and most programming languages only ascending indexing is available.

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

VHDL has file input and output capabilities, and can be used as a general-purpose language for text processing, but files are more commonly used by a simulation test bench for stimulus or verification data. There are some VHDL compilers which build executable binaries. In this case, it might be possible to use VHDL to write a test bench to verify the functionality of the design using files on the host computer to define stimuli, to interact with the user, and to compare results with those expected. However, most designers leave this job to the simulator.

One can design hardware in a VHDL IDE (for FPGA implementation such as Xilinx ISE, Altera Quartus, Synopsys Simplify or Mentor Graphics HDL Designer) to produce the RTL schematic of the desired circuit. After that, the generated schematic can be verified using simulation software which shows the waveforms of inputs and outputs of the circuit after generating the appropriate test bench. To generate an appropriate test bench for a particular circuit or VHDL code, the inputs have to be defined correctly. For example, for clock input, a loop process or an iterative statement is required.

A final point is that when a VHDL model is translated into the "gates and wires" that are mapped onto a programmable logic device such as a CPLD or FPGA, and then it is the actual hardware being configured, rather than the VHDL code being "executed" as if on some form of a processor chip.

Advantages

The key advantage of VHDL when used for systems design is that it allows the behavior of the required system to be described (modeled) and verified (simulated) before synthesis tools translate the design into real hardware (gates and wires). Another benefit is that VHDL allows the description of a concurrent system. VHDL is a Dataflow language, unlike procedural computing languages such as BASIC, C, and assembly code, which all run sequentially, one instruction at a time.

FIELD-PRORAMMABLE GATE ARRAY (FPGA)

A Field-programmable Gate Array (FPGA) is an integrated circuit designed to be configured by the customer or designer after manufacturing—hence "field-programmable". The FPGA configuration is generally specified using a hardware description language (HDL), similar to that used for an application-specific integrated circuit (ASIC) (circuit diagrams were previously used to specify the configuration, as they were for ASICs, but this is increasingly rare). FPGAs can be used to implement any logical function that an ASIC could perform.

FPGAs contain programmable logic components called "logic blocks", and a hierarchy of reconfigurable interconnects that allow the blocks to be "wired together"—somewhat like many (changeable) logic gates that can be inter-wired in (many) different configurations. Logic blocks can be configured to perform complex combinational functions, or merely simple logic gates like AND and XOR. In most FPGAs, the logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memory.

In addition to digital functions, some FPGAs have analog features. The most common analog feature is programmable slew rate and drive strength on each output pin, allowing the engineer to set slow rates on lightly loaded pins that would otherwise ring unacceptably, and to set stronger, faster rates on heavily loaded pins on high-speed channels that would otherwise run too slow. Another relatively common analog feature is differential comparators on input pins designed to be connected to differential signaling channels. A few "mixed signal FPGAs" have integrated peripheral Analog-to-Digital Converters (ADCs) and Digital-to-Analog Converters (DACs) with analog signal conditioning blocks allowing them to operate as a system-on-a-chip. Such devices blur the line between an FPGA, which carries digital ones and zeros on its internal programmable interconnect fabric, and field-programmable analog array (FPAA), which carries analog values on its internal programmable interconnect fabric.

MODERN DEVELOPMENTS

A recent trend has been to take the coarse-grained architectural approach a step further by combining the logic blocks and interconnects of traditional FPGAs with embedded microprocessors and related peripherals to form a complete "system on a programmable chip". This work mirrors the architecture by Ron Perlof and Hana Potash of Burroughs Advanced Systems Group which combined a reconfigurable CPU architecture on a single chip called the SB24. That

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

work was done in 1982. Examples of such hybrid technologies can be found in the Xilinx Virtex-II PRO and Virtex-4 devices, which include one or more PowerPC processors embedded within the FPGA's logic fabric. The Atmel FPSLIC is another such device, which uses an AVR processor in combination with Atmel's programmable logic architecture. The Actel Smart Fusion devices incorporate an ARM architecture Cortex-M3 hard processor core (with up to 512kB of flash and 64kB of RAM) and analog peripherals such as a multi-channel ADC and DACs to their flash-based FPGA fabric. An alternate approach to using hard-macro processors is to make use of soft processor cores that are implemented within the FPGA logic.

As previously mentioned, many modern FPGAs have the ability to be reprogrammed at "run time," and this is leading to the idea of reconfigurable computing or reconfigurable systems CPUs that reconfigure themselves to suit the task at hand. The Mitrion Virtual Processor from Mitrionics is an example of a reconfigurable soft processor, implemented on FPGAs. However, it does not support dynamic reconfiguration at runtime, but instead adapts itself to a specific program.

Xilinx claims that several market and technology dynamics are changing the ASIC/FPGA paradigm:
1.Integrated circuit costs are rising aggressively
2.ASIC complexity has lengthened development time
3.R&D resources and headcount are decreasing
4.Revenue losses for slow time-to-market are increasing.
5.Financial constraints in a poor economy are driving low-cost technologies
These trends make FPGAs a better alternative than ASICs for a larger number of higher-volume applications than they have been historically used for, to which the company attributes the growing number of FPGA design starts (see History).

Versus Complex Programmable Logic Devices
The primary differences between CPLDs (Complex Programmable Logic Devices) and FPGAs are architectural. A CPLD has a somewhat restrictive structure consisting of one or more programmable sum-of-products logic arrays feeding a relatively small number of clocked registers. The result of this is less flexibility, with the advantage of more predictable timing delays and a higher logic-to-interconnect ratio.
The FPGA architectures, on the other hand, are dominated by interconnect. This makes them far more flexible (in terms of the range of designs that are practical for implementation within them) but also far more complex to design for. Another notable difference between CPLDs and FPGAs is the presence in most FPGAs of higher-level embedded functions (such as adders and multipliers) and embedded memories, as well as to have logic blocks implements decoders or mathematical functions.

Security considerations
With respect to security, FPGAs have both advantages and disadvantages as compared to ASICs or secure microprocessors. FPGAs' flexibility makes malicious modifications during fabrication a lower risk. For many FPGAs, the loaded design is exposed while it is loaded (typically on every power-on). To address this issue, some FPGAs support bit stream encryption.

Decoding convolutional codes

Several algorithms exist for decoding convolutional codes. For relatively small values of k, the Viterbi algorithm is universally used as it provides maximum likelihood performance and is highly parallelizable. Viterbi decoders are thus easy to implement in VLSI hardware and in software on CPUs with SIMD instruction sets.

Longer constraint length codes are more practically decoded with any of several sequential decoding algorithms, of which the Fano algorithm is the best known. Unlike Viterbi decoding, sequential decoding is not maximum likelihood but its complexity increases only slightly with constraint length, allowing the use of strong, long-constraint-length codes. Such codes were used in the Pioneer program of the early 1970s to Jupiter and Saturn, but gave way to shorter,

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

Viterbi-decoded codes, usually concatenated with large Reed-Solomon error correction codes that steepen the overall bit-error-rate curve and produce extremely low residual undetected error rates.

Both Viterbi and sequential decoding algorithms return hard-decisions: the bits that form the most likely codeword. An approximate confidence measure can be added to each bit by use of the Soft output Viterbi algorithm. Maximum a posteriori (MAP) soft-decisions for each bit can be obtained by use of the BCJR algorithm.

Popular convolutional codes

An especially popular Viterbi-decoded convolutional code, used at least since the Voyager program has a constraint length k of 7 and a rate r of 1/2. Longer constraint lengths produce more powerful codes, but the complexity of the Viterbi algorithm increases exponentially with constraint lengths, limiting these more powerful codes to deep space missions where the extra performance is easily worth the increased decoder complexity. Mars Pathfinder, Mars Exploration Rover and the Cassini probe to Saturn use a k of 15 and a rate of 1/6; this code performs about 2 dB better than the simpler k=7 code at a cost of 256x in decoding complexity (compared to Voyager mission codes). Puncturing is a technique used to make a m/n rate code from a "basic" rate 1/2 code. It is reached by deletion of some bits in the encoder output. Bits are deleted according to puncturing matrix. The following puncturing matrices are the most frequently used: code rate puncturing matrix free distance (for NASA standard K=7 convolutional code) Error correction may generally be realized in two different ways:

Automatic repeat request (ARQ)

This is an error control technique whereby an error detection scheme is combined with requests for retransmission of erroneous data. Every block of data received is checked using the error detection code used, and if the check fails, retransmission of the data is requested – this may be done repeatedly, until the data can be verified.

Error detection schemes

Error detection is most commonly realized using a suitable hash function (or checksum algorithm). A hash function adds a fixed-length tag to a message, which enables receivers to verify the delivered message by recomputing the tag and comparing it with the one provided. There exists a vast variety of different hash function designs. However, some are of particularly widespread use because of either their simplicity or their suitability for detecting certain kinds of errors (e.g., the cyclic redundancy check's performance in detecting burst errors). Random-error-correcting codes based on minimum distance coding can provide a suitable alternative to hash functions when a strict guarantee on the minimum number of errors to be detected is desired. Repetition codes, described below, are special cases of error-correcting codes: although rather inefficient, they find applications for both error correction and detection due to their simplicity.

Repetition code

A repetition code is a coding scheme that repeats the bits across a channel to achieve error-free communication. Given a stream of data to be transmitted, the data is divided into blocks of bits. Each block is transmitted some predetermined number of times. For example, to send the bit pattern "1011", the four-bit block can be repeated three times, thus producing "1011 1011 1011". However, if this twelve-bit pattern was received as "1010 1011 1011" – where the first block is unlike the other two – it can be determined that an error has occurred. Repetition codes are not very efficient, and can be susceptible to problems if the error occurs in exactly the same place for each group (e.g., "1010 1010 1010" in the previous example would be detected as correct). The advantage of repetition codes is that they are extremely simple, and are in fact used in some transmissions of numbers stations.

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

Applications

Applications of FPGAs include digital signal processing, software-defined radio, aerospace and defense systems, ASIC prototyping, medical imaging, computer vision, speech recognition, cryptography, bioinformatics, computer hardware emulation, radio astronomy, metal detection and a growing range of other areas.

FPGAs originally began as competitors to CPLDs and competed in a similar space, that of glue logic for PCBs. As their size, capabilities, and speed increased, they began to take over larger and larger functions to the state where some are now marketed as full systems on chips (SoC). Particularly with the introduction of dedicated multipliers into FPGA architectures in the late 1990s, applications which had traditionally been the sole reserve of DSPs began to incorporate FPGAs instead.

FPGAs especially find applications in any area or algorithm that can make use of the massive parallelism offered by their architecture. One such area is code breaking, in particular brute-force attack, of cryptographic algorithm. The inherent parallelism of the logic resources on an FPGA allows for considerable computational throughput even at a low MHz clock rates. The flexibility of the FPGA allows for even higher performance by trading off precision and range in the number format for an increased number of parallel arithmetic units. This has driven a new type of processing called reconfigurable computing, where time intensive tasks are offloaded from software to FPGAs.
The adoption of FPGAs in high performance computing is currently limited by the complexity of FPGA design compared to conventional software and the turn-around times of current design tools.

CRYPTOGRAPHIC

It was the need to synchronize the scramblers that suggested to James H. Ellis the idea for non-secret encryption which ultimately led to the invention of the RSA encryption algorithm. The latest scramblers are not scramblers in the truest sense of the word, but rather digitizers combined with encryption machines. In these systems the original signal is first converted into digital form, and then the digital data is encrypted and sent. Using modern public-key systems, these "scramblers" are much more secure than their earlier analog counterparts. Only these types of systems are considered secure enough for sensitive data.

Voice inversion scrambling can be as simple as inverting the frequency bands around a static point to various complex methods of changing the inversion point randomly and in real time and using multiple bands. The "scramblers" used in cable television are designed to prevent casual signal theft - not to provide any real security. Early versions of these devices simply "inverted" one important component of the TV signal, re-inventing it at the client end for display. Later devices were only slightly more complex, filtering out that component entirely and then adding it by examining other portions of the signal. In both cases the circuitry could be easily built by any reasonably knowledgeable hobbyist. See Television encryption

Frame-Based Processing

If the input is frame-based, then both it and the Puncture vector parameter must be column vectors. The length of the Puncture vector parameter must divide K. The block repeats the puncturing pattern, if necessary, to cover all input elements.

ARCHITECTURE

The most common FPGA architecture consists of an array of logic blocks (called Configurable Logic Block, CLB, or Logic Array Block, LAB, depending on vendor), I/O pads, and routing channels. Generally, all the routing channels have the same width. Multiple I/O pads may fit into the height of one row or the width of one column in the array. An application circuit must be mapped into an FPGA with adequate resources. While the number of CLBs/LABs and I/Os required is easily determined from the design, the number of routing tracks needed may vary considerably even among designs with the same amount of logic. For example, a crossbar switch requires much more routing than a systolic array with the same gate count. Since unused routing tracks increase the cost (and decrease the performance) of the part without providing any benefit, FPGA manufacturers try to provide just enough tracks so that most designs that will fit

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

in terms of LUTs and IOs can be routed. This is determined by estimates such as those derived from Rent's rule or by experiments with existing designs.

In general, a logic block (CLB or LAB) consists of a few logical cells (called ALM, LE, Slice etc). A typical cell consists of a 4-input Lookup table (LUT), a Full adder (FA) and a D-type flip-flop, as shown below. The LUT are in this figure split into two 3-input LUTs. In normal mode those are combined into a 4-input LUT through the left mux. In arithmetic mode, their outputs are fed to the FA. The selection of mode are programmed into the middle mux. The output can be either synchronous or asynchronous, depending on the programming of the mux to the right, in the figure example. In practice, entire or parts of the FA are put as functions into the LUTs in order to save space.
ALMs and Slices usually contains 2 or 4 structures similar to the example figure, with some shared signals. CLBs/LABs typically contains a few ALMs/LEs/Slices.

In recent years, manufacturers have started moving to 6-input LUTs in their high performance parts, claiming increased performance. Since clock signals (and often other high-fanout signals) are normally routed via special purpose dedicated routing networks in commercial FPGAs, they and other signals are separately managed.

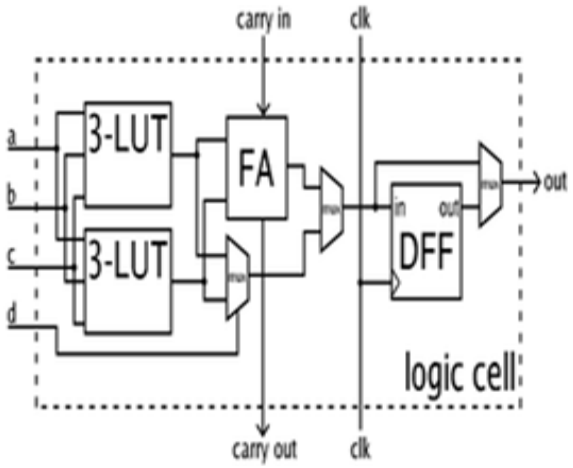

Fig Simplified example illustration of a logic cell

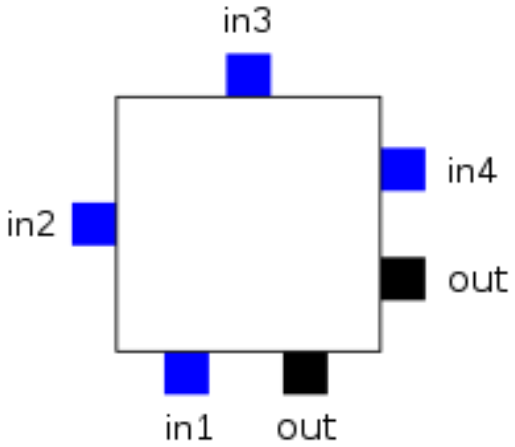

Fig The locations of the FPGA logic block pins

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

DIGITAL MODULATION METHODS

In digital modulation, an analog carrier signal is modulated by a digital bit stream. Digital modulation methods can be considered as digital-to-analog conversion, and the corresponding demodulation or detection as analog-to-digital conversion. The changes in the carrier signal are chosen from a finite number of M alternative symbols (the modulation alphabet).

A simple example: A telephone line is designed for transferring audible sounds, for example tones, and not digital bits (zeros and ones). Computers may however communicate over a telephone line by means of modems, which are representing the digital bits by tones, called symbols. If there are four alternative symbols (corresponding to a musical instrument that can generate four different tones, one at a time), the first symbol may represent the bit sequence 00, the second 01, the third 10 and the fourth 11. If the modem plays a melody consisting of 1000 tons per second, the symbol rate is 1000 symbols/second, or baud. Since each tone represents a message consisting of two digital bits in this example, the bit rate is twice the symbol rate, i.e. 2000 bits per second. According to one definition of digital signal, the modulated signal is a digital signal, and according to another definition, the modulation is a form of digital-to-analog conversion. Most textbooks would consider digital modulation schemes as a form of digital transmission, synonymous to data transmission; very few would consider it as analog transmission.

Fundamental digital modulation methods

These are the most fundamental digital modulation techniques:

1) In the case of PSK, a finite number of phases are used.
2) In the case of FSK, a finite number of frequencies are used.
3) In the case of ASK, a finite number of amplitudes are used.
4) In the case of QAM, a finite number of at least two phases, and at least two amplitudes are used.

LOGIC BLOCK PIN LOCATIONS

Each input is accessible from one side of the logic block, while the output pin can connect to routing wires in both the channel to the right and the channel below the logic block. Each logic block output pin can connect to any of the wiring segments in the channels adjacent to it. Similarly, an I/O pad can connect to any one of the wiring segments in the channel adjacent to it. For example, an I/O pad at the top of the chip can connect to any of the W wires (where W is the channel width) in the horizontal channel immediately below it.

Generally, the FPGA routing is un segmented. That is, each wiring segment spans only one logic block before it terminates in a switch box. By turning on some of the programmable switches within a switch box, longer paths can be constructed. For higher speed interconnect, some FPGA architectures use longer routing lines that span multiple logic blocks. Whenever a vertical and a horizontal channel intersect, there is a switch box. In this architecture, when a wire enters a switch box, there are three programmable switches that allow it to connect to three other wires in adjacent channel segments. The pattern, or topology, of switches used in this architecture is the planar or domain-based switch box topology. In this switch box topology, a wire in track number one connects only to wires in track number one in adjacent channel segments, wires in track number 2 connect only to other wires in track number 2 and so on. The figure below illustrates the connections in a switch box.

Switch box topology

Modern FPGA families expand upon the above capabilities to include higher level functionality fixed into the silicon. Having these common functions embedded into the silicon reduces the area required and gives those functions increased speed compared to building them from primitives. Examples of these include multipliers, generic DSP blocks, embedded processors, high speed IO logic and embedded memories. FPGAs are also widely used for systems validation including pre-silicon validation, post-silicon validation, and firmware development. This allows chip companies to validate their design before the chip is produced in the factory, reducing the time-to-market.

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

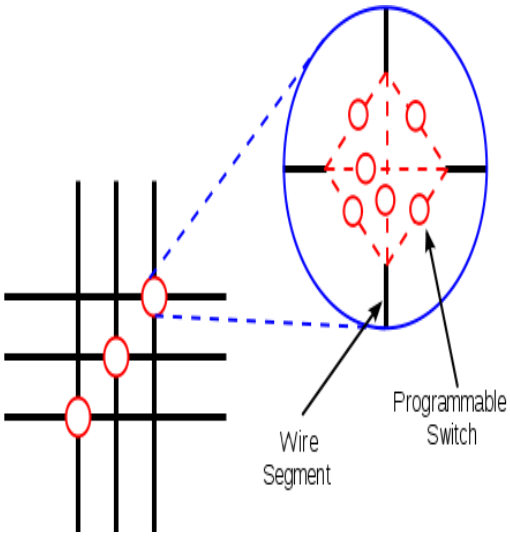

Fig The connections in the switchbox

FPGA DESIGN AND PROGRAMMING

To define the behavior of the FPGA, the user provides a hardware description language (HDL) or a schematic design. The HDL form is more suited to work with large structures because it's possible to just specify them numerically rather than having to draw every piece by hand. However, schematic entry can allow for easier visualization of a design. Then, using an electronic design automation tool, a technology-mapped net list is generated. The net list can then be fitted to the actual FPGA architecture using a process called place-and-route, usually performed by the FPGA Company's proprietary place-and-route software. Going from schematic/HDL source files to actual configuration: The source files are fed to a software suite from the FPGA/CPLD vendor that through different steps will produce a file.
This file is then transferred to the FPGA/CPLD via a serial interface (JTAG) or to an external memory device like an EEPROM.

The most common HDLs are VHDL and Verilog, although in an attempt to reduce the complexity of designing in HDLs, which have been compared to the equivalent of assembly languages, there are moves to raise the abstraction level through the introduction of alternative languages. National Instrument's LabVIEW graphical programming language (sometimes referred to as "G") has an FPGA add-in module available to target and program FPGA hardware. The LabVIEW approach drastically simplifies the FPGA programming process.

Switching State Determination Mechanism

Mathematical models were widely used to estimate the circuit delay and served well as the basics of static timing analysis (STA). In this work, instead of calculating the circuit delay, a switching state determination mechanism is proposed. The concept of the proposed mechanism is to distinguish between switching and stable states of the circuit. Obviously, only leakage current exists in the stable state. Therefore, by monitoring the change of drained current, the circuit state can determined.

IV. CONCLUSION

This paper presented the folded tree architecture of a digital signal processor for WSN applications. The design exploits the fact that many data processing algorithms for WSN applications can be described using parallel-prefix operations, introducing the much needed flexi bility. Energy is saved thanks to the following: 1) limiting the data set by pre-processing with parallel-prefix operations; 2) the reuse of the binary tree as a folded tree; and 3) the combin ation of

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

data flow and control flow elements to introduce a local distributed memory, which removes the memory bottleneck while retaining sufficient flexibility.

The simplicity of the programmable PEs that constitute the folded tree network resulted in high integration, fast cycle time, and lower power consumption. Finally, measurements of a 130-nm silicon implementation of the 16-bit folded tree with eight PEs were measured to confirm its performance. It consumes down to 8 pJ/cycle. Compared to existing commer-cial solutions, this is at least 10 × less in terms of overall energy and 2–3× faster

FUTURE ENHANCEMENT

The design exploits the fact that many data processing algorithms for WSN applications can be described using serial prefix operations the main goal is to design an ultra low energy WSN digital signal processor and minimize the memory access. Total dynamic energy -298.8(pj) 213.7(uw) total power is used. For the hardware requirement FPGA Trainer kit Spartan 3E Altera processor with Device:204pin package version 3E.

REFERENCES

[1] Cedric Walrvens, Member,IEEE,and Wim Denhance Senior Member,IEEE "Low Power digital signal architecture for wireless sensor nodes" vol 22 no 2,Feb 2014.
[2] C. Walravens and W. Dehaene, "Design of a low-energy data processing architecture for wsn nodes," in Proc. Design, Automat. Test Eur. Conf. Exhibit., Mar. 2012, pp. 570–573.
[3] N. Weste and D. Harris, CMOS VLSI Design: A Circuits and Systems Perspective. Reading, MA, USA, Addison Wesley, 2010.
[4] S. Mysore, B. Agrawal, F. T. Chong, and T. Sherwood, "Exploring the processor and ISA design for wireless sensor network applications," in Proc. 21th Int. Conf. Very-Large-Scale Integr. (VLSI) Design, 2008, pp. 59–64.
[5] M. Hempstead, J. M. Lyons, D. Brooks, and G.-Y. Wei, "Survey of hardware systems for wireless sensor networks," J. Low Power Electron., vol. 4, no. 1, pp. 11–29, 2008.
[6] J. Hennessy and D. Patterson, Computer Architecture A Quantitative Approach , 4th ed. San Mateo, CA: Morgan Kaufmann, 2007.
[7] P. Sanders and J. Träff, "Parallel prefix (scan) algorithms for MPI," in Proc. Recent Adv. Parallel Virtual Mach. Message Pass. Interf., 2006, pp. 49–57.
[8] H. Karl and A. Willig, Protocols and Architectures for Wireless Sensor Networks , 1st ed. New York: Wiley, 2005.
[9] V. N. Ekanayake, C. Kelly, and R. Manohar, "BitSNAP: Dynamic significance compression for a low energy sensor network asynchronous processor," in Proc. IEEE 11th Int. Symp. Asynchronous Circuits Syst., Mar. 2005, pp. 144–154.
[10] V. N. Ekanayake, C. Kelly, and R. Manohar "SNAP/LE: An ultra-low-power processor for sensor networks," ACM SIGOPS Operat. Syst. Rev. -ASPLOS, vol. 38, no. 5, pp. 27–38, Dec. 2004.