# ABSTRACT

Designing and manufacturing of X-ray imaging systems have been carried out for several decades. A wide range of industrial products, such as medical CT scanners, luggage security scanners, container scanners, industrial CT scanners, etc., have been developed. How to make these X-ray scanners smaller, cheaper, and faster is an important subject for the industry.

The aim of the thesis work is to design the X-ray line-scan camera with low cost, high flexibility, high performance and excellent reliability. In order to achieve these requirements, special attentions have been given to the design of the software of X-ray line-scan camera system, especially the digital process the pixel data from the control unit board.

In this thesis various digital design techniques have been studied for lower noise, higher processing speed and higher reliability. In most commercial X-ray line-scan cameras, the frame grabber card and RS-232 port are used to transfer image and control signals to computer. In this thesis new interfaces such as USB 2.0 and Ethernet are implemented in order to meet the cost and flexibility requirements from different customers. Reliability is a critical issue for real-time processing systems such as X-ray line-scan cameras. This has been thoroughly studied in this thesis in order to make the system more reliable with the implemented USB 2.0 and Ethernet interfaces. The throughput can reach 25 Mbit/s with Ethernet interface. The X-ray line scan camera prototype has been delivered for industrial tests, and the image results are presented and analyzed with mathematical software.

*Keywords:* X-ray line-scan camera, digital image process, control unit

# ACKNOWLEDGEMENT

This thesis was carried out in the Appons Technology Inc. CHINA, during the years of 2014. I would like to express my deepest gratitude to my supervisor Professor Yanne, for his warm advice and continuous encouragement. I would like also thank Mr. DianLu Gu for his kind help and valuable advices.

I have benefited significantly over the years from discussions with my gifted colleagues, especially Mr. Zheng Jian who is the FPGA and board designer of the X-ray line-scan camera, and Mr. Mikko Matikkala. They helped me a lot in carrying out this thesis work.

Finally, I would like to give special thanks to my wife, thanks, for your support and understanding all the time.

Oulu, June 2014

Yi Zheng

**TABLE OF CONTENT**

ABSTRACT 1

ACKNOWLEDGEMENT 2

LIST OF FIGURES 3

LIST OF TABLES 5

ABBREVIATIONS 6

1.1 Background 8

1.2 Scope of This Thesis Work 10

1.3. The Author’s Contribution 10

1.4 Report Organization 11

2. INTRODUCTION TO THE X-RAY LINE-SCAN CAMERA 12

2.1 Introduction to the Imaging System 12

2.2 Operation Principle of the X-ray Line-Scan Camera 14

2.3 Block-Diagram and Functional Description 15

2.3.1 Detector Board 16

2.3.2 Front-End Board 18

2.3.3 Control Unit Board 19

2.3.4 Power Supply Unit 21

2.3.5 Host PC 21

3. INTRODUCTION TO THE CONTROL UNIT BOARD 22

3.1 Overview 22

3.2 Architecture Discussion for the Control Unit Board 23

3.2.1 Background Technology 23

3.2.2 Primary Discussion 27

3.2.3 Comparison between DSP Microprocessor and FPGA 28

3.2.3 Final Conclusion 30

3.3 Block-Diagram and Functional Description 31

3.3.1 Pixel DSP Processor 32

3.3.2 Microcontroller and USB Controller 39

3.3.3 Fast Ethernet Chips 40

3.3.4 Memories and Other Components 43

3.4 Conclusion 44

3.4.1 Low Cost 45

3.4.2 High Flexibility 45

3.4.3 High Performance 45

4. IMPLEMENTATION OF THE PIXEL DSP PROCESSOR 47

4.1 Overview 47

4.2 Noise Reduction and Signal Enhancement 48

4.2.1 Fixed Pattern Noise Correction 48

4.2.2 Noise Reduction Ratio 53

4.2.3. FIR Moving Average Filter to Reduce White Noise 54

4.2.4. Hardware Realization of the Digital Filter 55

4.3 Output Buffer Control 60

4.3.1 The Output Buffer (1st level buffer) Control Process 62

4.3.2 The Bank Memory (2nd-level buffer) Control Process 63

5. MEASURED RESULTS AND DISCUSSIONS 69

5.1. Overview 69

5.2. Main Applications and Sample Pictures 69

5.2.1. Main Specifications 69

5.2.2. Main Applications and Sample Images 71

6. Conclusions 76

References 77

# LIST OF FIGURES

Figure 1 A typical X-ray line scan camera (Courtesy of Detection Technology Inc.) 8

Figure 2 A typical X-ray line-scan camera’s mechanics. 9

Figure 3 The operation principle of a typical imaging system 12

Figure 4 X-ray application measurement environment 13

Figure 5 The operation principle of X-ray line-scan camera 14

Figure 6 The block diagram of the X-ray line-scan camera 15

Figure 7 Operation principle of photodiode array [13] 16

Figure 8 The block diagram of detector board 17

Figure 9 The block diagram of front-end board 18

Figure 10 The simplified block diagram of control unit board 19

Figure 11 The structure of an FPGA 24

Figure 12 SRAM-controlled programmable switches 24

Figure 13 A representative conventional fixed-point DSP processor data path (from the Motorola DSP560xx, a 24-bit, fixed-point processor family). 25

Figure 14 The comparison between FPGA and DSP based on the sample data rate []. 28

Figure 15 Using an FPGA as a coprocessor rates special attention [] 29

Figure 16 The top level block diagram of control unit board 30

Figure 17 The block diagram of pixel DSP processor 31

Figure 18 The block diagram of FE controller 32

Figure 19 The block diagram of decoder 33

Figure 20 The block diagram of SRAM interface 34

Figure 21 The block diagram of output controller 35

Figure 22 The block diagram of DSP processor 36

Figure 23 Block diagram of FX2 USB controller 38

Figure 24 The block diagram of i2Chip W3100A 39

Figure 25 The block diagram of RTL8201BL 40

Figure 26 A simplified signal flow in X-ray line-scan camera 43

Figure 27 The DSP data flow diagram of the pixel DSP processor 45

Figure 28 Offset calibration (a) a blank image without offset calibration (b) a blank image with offset calibration. 48

Figure 29 Gain calibration (a) a blank image without gain calibration (b) a blank image with gain calibration. 48

Figure 30 X-ray scanning images (a) without calibration (b) with calibration. 48

Figure 31 Architecture of a FIR filter. 51

Figure 32 Wrapped linear delay-line buffer. 52

Figure 33 Circular delay-line buffer. 53

Figure 34 The noise of the offset values (a) before filtering (b) A four order moving average filter is used; (c) An eight order moving average filter is used. 54

Figure 35 Block diagram of buffer control 56

Figure 36 The output buffer (1st-level buffer) state diagram 57

Figure 37 The control process of the output buffer 58

Figure 38 Simplified timing of DSP processing using bank memory 59

Figure 39 The state of 2nd level buffer 60

Figure 40 The write control of the bank buffer 61

Figure 41 The read control of bank buffer 62

Figure 42 400μm line pairs can be discerned. 64

Figure 43 Bread package with metal wires on the bottom. 65

Figure 44 Sample pictures on industrial food quality inspection: (a) Chocolate bar with alien materials (b) Sausages package with alien material (c) Juice box with metal wire (d) Chicken with bones. 67

Figure 45 Custom security inspection. 68

Figure 46 Industrial inspection (a) Air fresher bottle with a metal ball inside (b) a bulb inside paper package. 68

Figure 47 Electronic assembly industry quality inspection (a) A multi-meter (b) A telephone. 69

Figure 48 Pseudo-colored picture. 69

# LIST OF TABLES

Table 1 Comparison of FPGAs and DSP Processors 28

Table 2 Cyclone FPGA features 31

Table 3 Comparison between the theoretical values and the measured results 53

Table 4 Comparison of data lost for different buffer size 63

# ABBREVIATIONS

AC Alternating Current

AD Analog to Digital

ADC Analog Digital Converter

AGND Analog ground

ARP Address Resolution Protocol

API Application Programming Interface

ASIC Application-specific integrated circuit

BGA Ball Grid Array

CCD Charge-coupled device

CISC Complex Instruction Set Computer

CMOS Complementary metal oxide semiconductor

CPLD Complicated programmable logical device

CU Control Unit

DAS Data Acquisition System

DC Direct Current

DGND Digital ground

DSP Digital signal processing

EMC Electromagnetic compatibility

FE Front-end

FIFO First-in First-out

FIR Finite impulse response

FPGA Field programmable gate logic

FPIC Field-Programmable Interconnect Component

FPN Fixed pattern noise

GND Ground

GPIF General Programmable Interface

GPP General-purposed Processor

HDL Hardware Description Language

IC Integrated circuit

ICMP Internet Control Message Protocol

IEEE Institute of Electrical and Electronics Engineers, Inc.

IIR Infinite impulse response

IP Internet Protocols

ISP In System Programming

I/O Input/output

I2C Inter－Integrated Circuit

LAN Local Area Network

LINAC Linear accelerator

LSB Least Significant Byte

LSI Large Scale Integration

MAC Multiplier accumulator

MAC Media Access Control

MACs Multiply-accumulators

MII Media Independent Interface

MPGA Mask-programmable Gate Arrays

MUX Multiplexer

NDT Non-destructive testing

NRE Non Recurring Engineering

PAL Programmable Array Logic

PC Personal Computer

PCB Printed Circuit Board

PCI Peripheral Component Interconnect

PCM Processing control monitor

PCS Physical Coding Sub-layer

PGA Programmable Gain Amplifier

PMA Physical Medium Attachment

PLL Phase Lock Loop

RAM Random Access Memory

RISC Reduced Instruction Set Computer

RMS Root Mean Square

SIE Serial Interface Engine

SoC System-on-Chip

SNI Serial Network Interface

SNR Signal Noise Ratio

SRAM Static Random Access Memory

S/H Sample and hold

S/N Signal/noise ratio

TCP Transmission Control Protocol

TTL Transistor-Transistor Logic

USB Universal serial bus

USART Universal Synchronous Asynchronous Receiver / Transmitter

VHDL VHSIC Hardware Description Language

VLIW Very Long Instruction Word

UDP User Datagram Protocol

**1. INTRODUCTION**

## 1.1 Background

Nowadays X-ray applications are more and more important in industrial inspections. Its ability to inspect the contents of packages non-destructively and without contact makes the X-ray line-scan camera ideal for a wide range of applications, including detection of foreign objects in food products and electronic components.

X-ray line-scan cameras are very common in various quality inspections and non-destructive testing (NDT) applications. They are used for finding foreign objects in food products, checking for missing products in packages, checking the fill-level, and inspecting contents of passenger’s luggage. X-ray line-scan cameras have been available for more than ten years and during these years the technology has developed towards higher scanning speeds, better dynamic range, and smaller pixel sizes [[[1]](#endnote-2)]. The linear arrays have become more and more common in the field of NDT. A typical X-ray line-scan camera is illustrated in Figure 1.

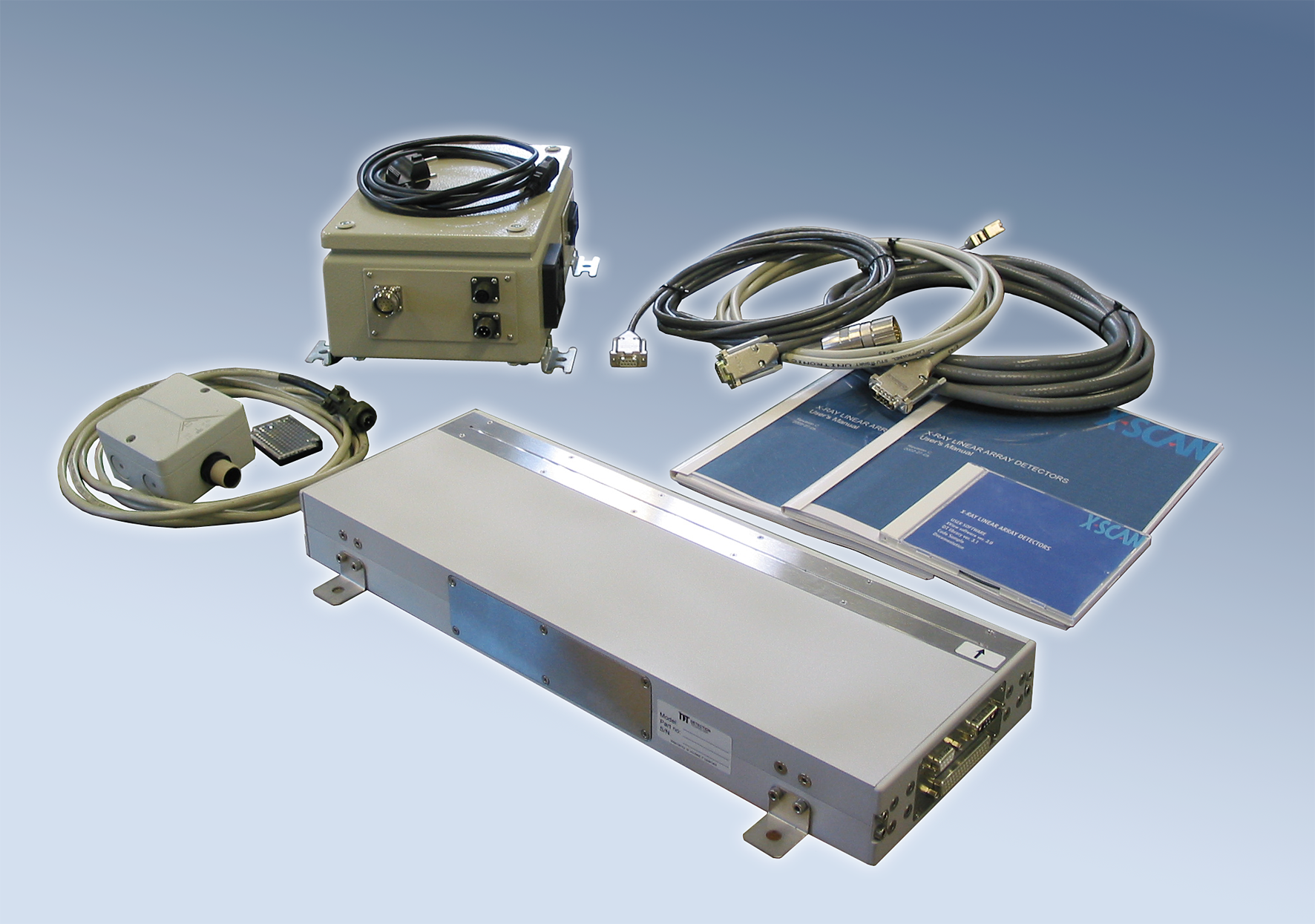


Figure 1 A typical X-ray line scan camera (Courtesy of Detection Technology Inc.)

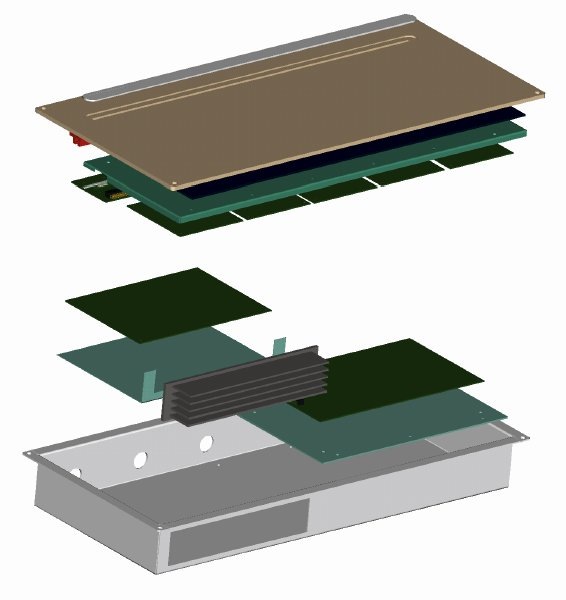
X-ray line-scan cameras are complicated devices with large amount of electronics. A typical modern X-ray line-scan camera can be divided into the following main parts: scintillator, photodiode array, detector front-end, data acquisition system, control unit, mechanics, power supply, accessories, frame grabber card, and software.

Most of the X-ray line-scan cameras use certain type of scintillators to transform the incoming radiation into visible light. The reason is that the absorption efficiency of plain photodiode is not high enough for X-ray energies above 30-35 keV. Photodiode arrays [[[2]](#endnote-3)], [[[3]](#endnote-4)], [[[4]](#endnote-5)] are used to measure the amount of visible light generated by the scintillators [[[5]](#endnote-6)].

The front-end part includes pre-amplifying and multiplexing electronics [[[6]](#endnote-7)], [[[7]](#endnote-8)], [[[8]](#endnote-9)], [[[9]](#endnote-10)]. The purpose of the pre-amplifying electronics is to integrate and amplify the very low current output from the photodiode arrays. Usually the scintillator, photodiode array and front-end electronics are made into a detector board. After the pre-amplifying and multiplexing electronics, there is a part called DAS (Data Acquisition System). The DAS typically contains other amplifier stages, such as adjustable gain amplifiers and buffer amplifiers, which amplify the signal further until it is suitable for the A/D-conversion. The main function of the DAS is A/D-conversion of the signal.

The Ethernet interfaces electronics take care of the communications between the X-ray line-scan camera and computer. The control unit transforms and buffers the data in order for Ethernet client to be able to read it.

The mechanics include the metal enclosure, which also acts as the electromagnetic shielding [[[10]](#endnote-12)], X-ray beam collimator, X-ray entrance window, and the X-ray shielding for electronics. The enclosure may be sealed or include a fan for forced-air cooling. There are separate enclosures for the power supply and the detector. The control unit may also be in its own enclosure. Figure 2 shows an example of the X-ray line-scan camera’s mechanics.



**Cover**

**Aluminum window that can pass X-ray**

###### Lead cover to protect electronics

###### Metal enclosure

###### Heatsink

Figure 2 A typical X-ray line-scan camera’s mechanics.

There are also small accessories, such as cables, connectors, etc. During the assembly of these parts, electromagnetic compatibilities and radiation protection should be carefully considered.

X-ray Line-Scan cameras are ideal for in-time X-ray photography, enabling capture of high resolution X-ray images of objects moving on equipment such as belt conveyors.

## 1.2 Scope of This Thesis Work

During the last few years, intensive research work has been carried out to develop X-ray imaging sensors and systems. The availability of higher performance PC, density FPGA and more powerful DSP has made it possible to build low cost bust high performance x-ray line-scan camera, which can be easily customized. Optimizing the X-ray line-scan camera with digital technique for a specific application becomes easier. Due to the improved flexibility and the diversified application requirements, there will most probably be, in the future, wider variety of options for the X-ray line-scan camera.

The purpose of this thesis is to study and implement the software of the X-ray line-scan camera, and especially for digital signal processing and flexible interfaces design. The digitalized signal processing can reduce system noise and make certain customized data processing for system normalization and signal enhancement. Nowadays system operating speed is very crucial, so some methods have also been implemented in this work to increase the processing speed of the control unit board.

In this thesis, a complete design of X-ray line-scan camera has been presented first. The purpose of this study is to develop an X-ray line-scan camera with good performance, better potential to be optimized in the future, and also with lower cost and better reliability.

## 1.3. The Author’s Contribution

The the X-ray line-scan camera from Appons Technology Inc. has been developed during last one and half years. Currently, the X-ray line-scan camera has been delivered to the industrial tests, and the production line has been started.

The thesis will focus on the software design part of the X-ray line-scan camera, which consists of design of firmware running in uCLinux, PC side library and application. The author independently designed the firmware and PC side software. The author’s contribution includes the following:

1. Whole software architecture design
2. Firmware design and implementation;
3. Support Library design and implementation;
4. Application design and implementation;
5. Hardware debugging and verification;
6. CPLD timing design of the DAS board;

This thesis emphasizes the software design of the X-ray line-scan camera. Great efforts have been made to meet the target requirements. Most of the original design ideas in this thesis have been carefully verified.

## 1.4 Report Organization

Chapter one is a general introduction to this thesis work, it shows the background information as well as the scope of the thesis work; introduction to the X-ray line-scan camera is given in chapter two; it gives the basic knowledge for a good understanding of the thesis work. The introduction of the firmware in the X-ray line-scan camera is given in chapter three. Architecture of the control board is also discussed in this chapter; Chapter four describe data process algorithms and software implementations of the pixel processor in detail, followed by an analysis. Chapter five describes the architecture of the library and application. Chapter six gives the conclusion.

# 2. INTRODUCTION TO THE X-RAY LINE-SCAN CAMERA

## 2.1 Introduction to the Imaging System

X-ray line-scan camera is a kind of linear-array x-ray detector, which is designed to be the imaging detector component for inspection, testing and quality control instruments. The imaging is based on moving the object at a constant speed relative to an x-ray source and the detector, and to collect and store a two-dimensional x-ray image line by line.

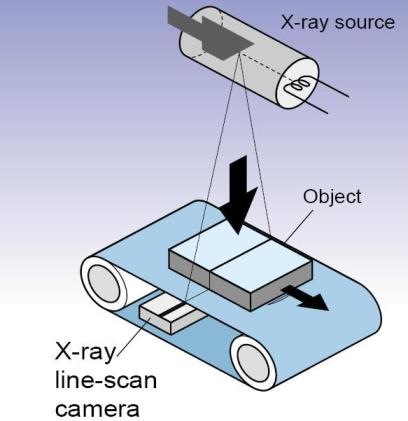


Figure 3 The operation principle of a typical imaging system

The operation principle of a typical imaging system is presented in Figure 3. A X-ray line scan camera works according to the same principle as a fax machine, also sometimes referred to as a “push broom” operation. It captures a single line at a time (horizontal), and requires either the object or the camera to be moving in order to create the vertical direction of the image.

The X-ray line-scan camera can take a high-sensitivity, high-resolution transparent X-ray image of an inspected object transported on a belt conveyor or similar apparatus. Since the defects or foreign objects in an object, which is not visible with the naked eye, can be inspected without contact or destruction, this kind of cameras are suitable for broad interior X-ray observation, enabling the detection of a foreign body mixed in food, electronic components, etc.

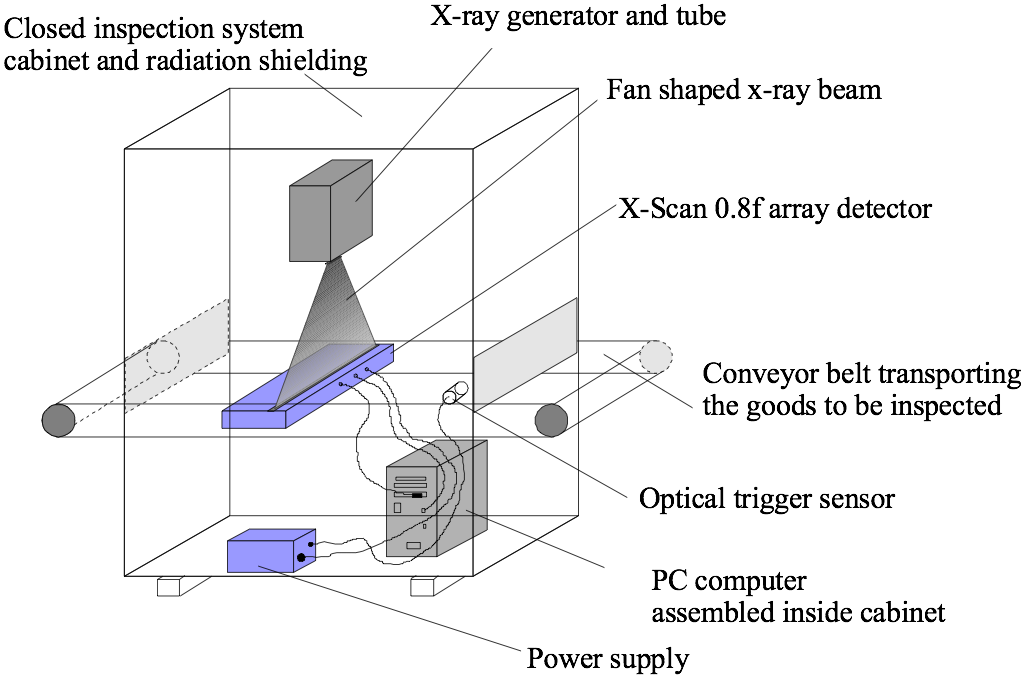


Figure 4 X-ray application measurement environments

The main components of a typical imaging system are shown in Figure 4 above:

1. A conveyor where the object is transported across the detector’s field-of-view.

2. An X-ray tube, a high-voltage generator and a control system.

3. A host PC equipped with an image acquisition electronics board (frame grabber) or Ethernet or USB 2.0 interfaces.

4. Necessary mechanics for system integration and radiation shielding.

5. An X-ray line-scan camera and a power supply unit.

6. An optical trigger sensor for an image frame trigger.

The X-ray tube is placed on the top of the cabinet of the X-ray inspection instrument. It projects a fan-shaped X-ray beam. The X-ray line-scan camera with photodiode array is assembled in a line inside the metal enclosure. The control unit board with digital signal processor and other circuits are also assembled in the same metal enclosure. The X-ray line-scan camera is placed under the conveyor belt. When the scanned objects are moving on the conveyor belt, the X-ray beam will penetrate into the objects and the images are created in the X-ray line-scan camera. The power supply unit and computer are assembled in the same cabinet. Different equipments are connected together with cables []. [[11]](#endnote-13)

2.2Operation Principle of the X-ray Line-Scan CameraFigure 5 The operation principle of X-ray line-scan camera

We could easily understand an X-ray line-scan camera system with its data flow. The principle of data flow is presented in Figure 5 above for a typical X-ray line-scan camera. The detector board, the front-end board and the control unit board are three key components to convert X-rays to digital signals that can be identified by the computer software.

Since only visible light could be detected by the silicon photodiodes, the incoming X-rays need to be first converted to visible light by using scintillators in an intensifying screen or a crystal array. Then the visible light is further detected and accurately measured by a number of silicon photodiode arrays.

A scintillation screen is mounted on the top of photodiode array. The scintillator material and an individual photodiode form a single sensitive element, which is so called pixel in the detector. The spacing of the sensitive elements is usually from 0.2 to 2.5 mm depending on the applications; this spacing is called pixel pitch.

Each sensitive element of the detector provides a signal current that is proportional to the flux and energy of the X-rays absorbed in this sensitive element of the detector. The photocurrent from the silicon photodiode sensor is already an electrical signal (instead of a physical signal such as X-ray) that is loaded with the information of the scanned objects. The geometric shapes, thickness, inside structures and non-uniformities of the objects are detected by the photodiodes. The photocurrent itself cannot be recognized by the computer so it has to be sent to a front-end CMOS integrated circuit to be sampled and processed. So the signal current is integrated by a highly sophisticated front-end electronics that is generally implemented in an ASIC chip. The sensitivity of the front-end electronics can be decided by using different capacitance of the feedback capacitor that is used to integrate the signal current.

The integration time of the front-end electronics is defined by the timing control. The output of the front-end electronics is an analogue voltage proportional to the total integrated current, i.e., the charge. The voltage outputs from multiple front ends are multiplexed into an A/D converter. The voltage amplitude before the A/D converter can be multiplied by a configurable factor that is the analog gain by using a programmable gain amplifier (PGA). The output digital parallel data from an A/D converter is then read by a timing controller of the front-end board, it is also responsible to generate all timing to control detector board and AD converter. After the timing controller, the output digital parallel data is converted into differential serial digital data and sent to the control unit board.

In the control unit board, the serial digital is read by a pixel DSP processor, and converted to the parallel data that can be real-time processed in the pixel DSP processor. After processing, the digital data is sent to the frame grabber, or the FIFO of the USB controller, or the buffer of Ethernet controller chip. Necessary internal timing signals of the control unit board are generated in a timing control block of the pixel DSP processor. The internal settings of the camera are controlled by a microcontroller. The microcontroller receives the control commands from the computer by one of the three interfaces: the RS-232, the USB 2.0, or the Ethernet interface. Necessary operational voltages for all electronics inside the X-ray line-scan camera are generated from an external DC voltage by power distribution electronics [12].

## 2.2 Block-Diagram and Functional Description

A principle block diagram of the X-ray line-scan camera is presented below in Figure 6.

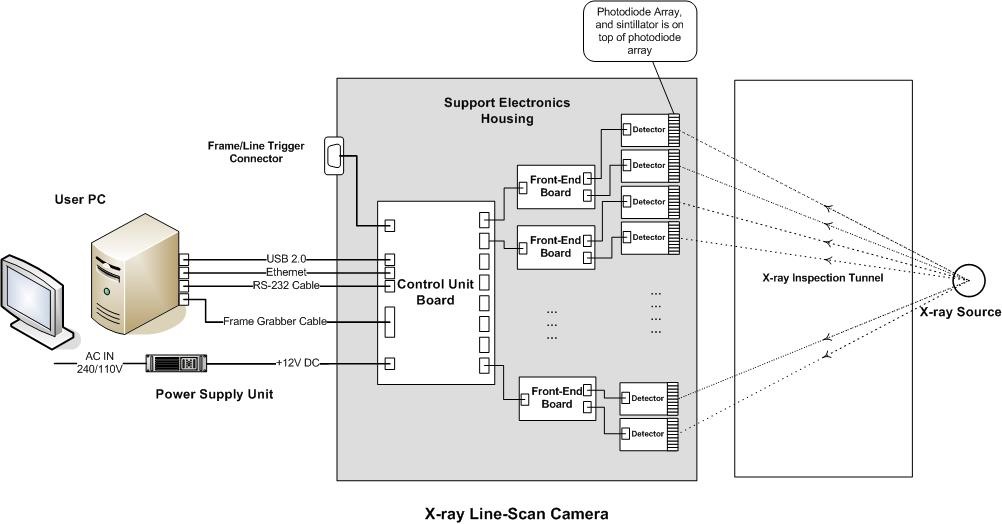


Figure 6 The block diagram of the X-ray line-scan camera

Typically, an X-ray line-scan camera consists of detector boards, A/D or Front-End boards (FEs), a control unit board (CU) and optionally a DC power supply. Cabling between these units is also schematically shown in Figure 6.

A number of detector boards are mounted in the mechanic, they are under a collimator along with the window of the camera. The active area plane of the detector elements is under the X-ray beam from focal spot of the X-ray source. Every two detector boards are connected to a single front-end board while all of front-end boards are connected to the control unit board. At most eight FE boards could be connected to one CU board because of the physical area limitation of the CU board. All of detector boards work in parallel. Every FE board multiplexes the analog signals from two detector boards, and converts them to digital signals. After that, the FE boards send the digital signals to the CU board. All of these FE boards work in parallel, too. The CU board receives the data from these FE boards, and arranges the order of the pixel data according to the physical pixel positions. Then it performs the necessary digital signal processing for the re-sequenced pixel data and sends the processed data to the user defined interface. The computer software performs two functions, one is to further process or display these processed data, and the other is to send commands to the CU board to control the whole system.

2.3.1

### 2.3.3 Control Unit Board

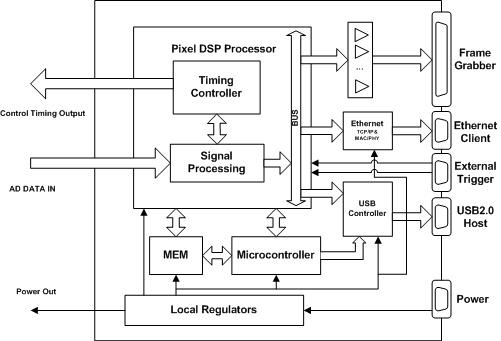


Figure 10 The simplified block diagram of control unit board

The control unit board is responsible for collecting the data from FE boards, arranging the data to proper format and performing the necessary signal processing for each pixel value. After signal processing, the processed data could be sent to the frame grabber or to the USB 2.0/Ethernet interfaces so that the data could be received by the user computer. The key component in the control unit board is the pixel DSP processor. In the thesis the pixel DSP processor is realized by a FPGA chip. Each pixel is handled as a 16-bit data value inside the control unit. The operations of the control unit and whole system are controlled by user computer via RS-232 serial link, USB 2.0 or Ethernet link.

The simplified block diagram of the control unit board is presented in Figure 10. The main components of the control unit board are the pixel DSP processor, the microcontroller, the USB controller, Ethernet chips, the power circuit and related buffers and drivers.

There are altogether eight connectors that could be used to connect FE boards. Thus at most eight FE boards can be connected to one control unit board. The output data from each FE board is the multiplexed serial digital signal that is controlled by a clock for serial link. There are altogether three interfaces to transfer image data to the host PC.

### 2.3.4 Power Supply Unit

The power supply unit converts the supply voltage (AC 110/240 V) into +12V DC voltage, which can be used by the electronics of X-ray line-scan cameras. The control unit board and FE boards generate internal voltages using local regulators from the +12V DC power supply.

### 2.3.5 Host PC

1. Remote host PC is needed in our system to control the X-ray line-scan camera. The host PC should be capable to handle the data interface at required rate and to process the data from the X-ray line-scan camera. Customized software is needed in the host PC to control the X-ray line-scan camera. The software takes care of the image grabbed into the host PC (or frame grabber) memory, displays the continuous image and sends the control commands to X-ray line-scan camera. Control Detector
2. Image Receiving
3. Image Processing

# 3. INTRODUCTION TO THE CONTROL UNIT BOARD

## 3.1 Overview

In our X-ray line-scan camera system, the control unit board is used for digital signal processing purpose. It is the key component of the digital X-ray line-scan camera. The main functions of the control unit board consist of controlling and digital signal processing. They are listed below:

* Get the raw data from AD or FE board.
* Provide all control timing for the whole system.
* Digital signal processing to improve the performance.
* Provide power for the control unit board and front-end boards.
* Communicate with host PC.

The main components of the control unit board are the pixel DSP processor, the microcontroller, the USB controller, Ethernet chips, memories, buffers and necessary drivers. The key component is the pixel DSP processor, which implements all the real-time digital signal processing operations.

Since one of my aims in this thesis work is to design the control unit board especially with low cost, high flexibility, high performance and excellent reliability, we will focus on how to meet these targets in the architecture design of control unit board in the following sections.

## 3.2 Hardware Software partition discussion for the Control Unit Board

As mentioned in the last section, we have certain design targets for the control unit board. Since pixel DSP processor is the most important component in this board, its implementation techniques turn out to be the most critical issues for our designs of the control unit board.. And the dsp processor will occupy the most part of FPGA resource. To make a

### 3.2.1 Background Technology

Based on the design requirements, the pixel DSP processor should be programmable for a variety of signal processing operations, such as filtering, spectrum estimation, and other DSP algorithm. It should also generate all the timing signals to control the whole system including the control unit board and front-end or AD boards. For such applications there are numerous software/hardware candidates, including general purpose processors, DSP processors, custom ASICs and field programmable gate arrays (FPGAs). These alternatives offer varying degrees of performance benefits that must be weighed against factors such as cost, power consumption and the design time.

Before we could start to discuss the right kind of circuit that should be deployed for realization, the background technologies need to be introduced first for understanding the advantages and disadvantages of these candidates.

#### 3.2.1.1. FPGA Technology

Field programmable gate arrays (FPGA) are a technology between programmable array logic (PAL) and custom gate arrays such as mask-programmable gate arrays (MPGA). PALs were developed in the 1970's and found many applications in replacing one to ten discrete TTL devices in hardware designs. These devices use a fuse technology that is one-time programmable. On the other extreme, fully custom integrated circuits like MPGAs were designed for much higher densities. MPGAs consist of an array of pre-fabricated transistors that can be customized into the user’s logic circuit by connecting the transistors with custom wires. Customization is performed during chip fabrication by specifying the metal interconnect, and this means that in order for a user to employ an MPGA a large setup cost is involved and manufacturing time is long.

Field programmable gate arrays take this technology a step further. Like MPGAs, FPGAs comprise an array of uncommitted circuit elements, called logic blocks, and interconnect resources. The interconnects in FPGAs are user definable, either through a fuse technology as in PALs and therefore only one-time programmable, or as static RAM cells that can be reprogrammed. The design process is similar to that of a gate array. Due to its high density and programmability, FPGAs are becoming popular for designing the logic circuits for relatively small quantities. Especially SRAM-based FPGAs have great advantages over other types of FPGAs for quick prototyping; they can be re-programmed at users' site in a short time period.

Today’s FPGAs have millions of logic cells and embedded microprocessors, making them incredibly powerful and inexpensive for large-scale computational tasks. Market dynamics, technology innovation, cost models, and tools have all been affected by the deep submicron era. All these factors have made FPGAs a very attractive solution in many areas. FPGAs are now often the heart of the system, being designed into mainstream as well as state-of-the-art high volume products.

A typical FPGA is composed of three major components: logic modules, routing resources, and input/output (I/O) modules. Figure 11 depicts the conceptual FPGA model. In an FPGA, an array of logic modules is surrounded or overlapped by general routing resources bounded by I/O modules. The logic modules contain combinational and sequential circuits that implement logic functions. The routing resources comprise pre-fabricated wire segments and programmable switches. The interconnections between the logic modules and the I/O modules are user programmable. A logic circuit is implemented in an FPGA by partitioning logic into individual logic modules and then interconnecting the modules by programming 2 switches. A large circuit that cannot be accommodated into a single FPGA is divided into several parts; each part is realized by an FPGA and these FPGAs are then interconnected by a Field-Programmable Interconnect Component (FPIC).

For commercial FPGAs there are two major programming technologies to program the switches: Static Random Access Memory (SRAM) and antifuse. Each technology has its strengths as well as weaknesses. The following sections detail important properties of the programming technologies.

The SRAM programming technology employs SRAM cells to control pass transistors or multiplexers as shown in Figure 12 SRAM-controlled programmable switches []. An example of usage of SRAM-controlled switches is illustrated in this figure, showing these two applications of SRAM cells: for controlling the gate nodes of pass-transistor switches and to control the select lines of multiplexers that drive logic block inputs. The figure gives an example of the connection of one logic block (represented by the AND-gate in the upper left corner) to another through two pass-transistor switches, and then a multiplexer, all controlled by SRAM cells. Whether an FPGA uses pass-transistors or multiplexers or both depends on the particular product.  
[[12]](#endnote-15)

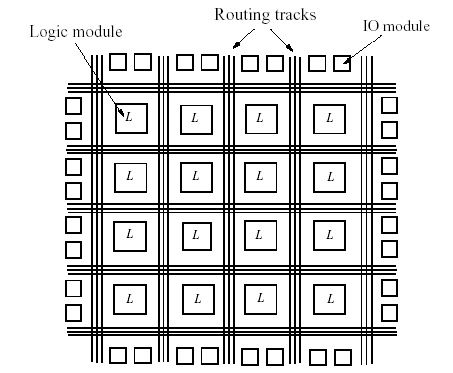


Figure 11 The structure of an FPGA

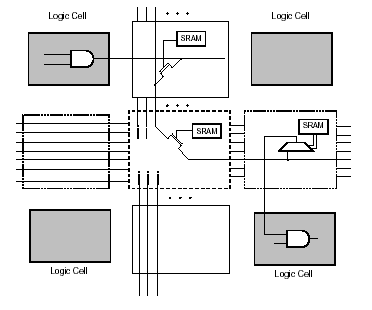


Figure 12 SRAM-controlled programmable switches

Currently the biggest vendors Xilinx and Altera provide their own EDA software for design, verification, simulation and synthesis, so the design can be done at a block-diagram level. Many blocks can be very high level – ranging from a single gate to an FIR or FFT. Their performance is limited by the number of gates they have and the clock rate. Recent FPGAs have included Multipliers especially for performing DSP tasks more efficiently.

#### 3.2.1.2. DSP Processor

DSP processors are microprocessors designed to perform digital signal processing—the mathematical manipulation of digitally represented signals. Most DSP processors share some common basic features designed to support high performance, repetitive, numerically intensive tasks. The most often cited of these feature is the ability to perform one or more multiply-accumulate operations (often called “MACs”) in a single instruction cycle. The multiply-accumulate operation is useful in DSP algorithms that involve computing a vector dot product, such as digital filters, correlation, and Fourier transforms. To achieve a single-cycle MAC, DSP processors integrate multiply-accumulate hardware into the main data path of the processor, as shown in Figure 1[[[13]](#endnote-16)]. Some recent DSP processors provide two or more multiply-accumulate units, allowing multiply-accumulate operations to be performed in parallel. In addition, to allow a series of multiply-accumulate operations to proceed without the possibility of arithmetic overflow (the generation of numbers greater than the maximum value the processor’s accumulator can hold), DSP processors generally provide extra “guard” bits in the accumulator. For example, the Motorola DSP processor family examined in Figure 13 offers eight guard bits [15].

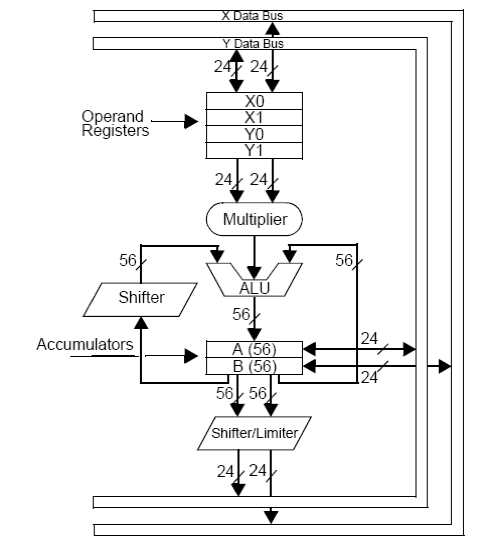


Figure 13 A representative conventional fixed-point DSP processor data path (from the Motorola DSP560xx, a 24-bit, fixed-point processor family).

The DSP processor is a specialized microprocessor – typically can be programmed in C language, perhaps with assembly code for performance. It is well suited to extremely complex maths-intensive tasks, with conditional processing. It is limited in performance by the clock rate, and the number of useful operations it can do per clock. As an example, a TMS320C6201 who is the fixed-point digital signal processor of Texas Instruments has two multipliers and a 200MHz clock – so can achieve 400M multiplies per second.

#### 3.2.1.3. ASICs

An ASIC is an application-specific integrated circuit that is customized or tailored to perform specific functions to a particular system or application; they are typically employed as bus interfaces, glue logic, functional accelerators, and/or a System-On-Chip (SoC).

The main advantage of ASICs is that they are designed with a specific application in mind and thus they are optimized for that application from the viewpoint of performance. ASICs also benefit from the fact that their manufacturing process can be optimized to exclude any extra circuitry due to the fixed functionality and this reduces unit costs [].  
[[14]](#endnote-17)

However, ASICs have two main disadvantages. One is the long design cycle which includes circuit design and the manufacturing process setup. This translates into much higher upfront cost compared to FPGAs and the gap is still increasing. In addition, upgrades and design corrections take a similarly large amount of time and cost so the investment risk is higher when one opts for ASICs.

#### 3.2.1.4. General-Purpose Processor

General-purpose processor (GPP) is a family of microprocessors and microcontrollers whose architecture are represented by complex instruction set computer (CISC), reduced instruction set computer (RISC), or the very long instruction word (VLIW). GPPs are best suited for performing a broad array of tasks that are not specifically tailored for any particular application. It can also be programmed with C language and assembly language.

### 3.2.2 Primary Discussion

Depending on the speed and computational requirements of different applications, the digital signal processor may be realized by a general-purposed processor, microprocessor, ASIC, FPGA or DSP microprocessor. The X-ray line-scan camera is a real-time processing system, and the processing speed need to be relatively fast. In this sense, as the core processing component of the system, the pixel DSP processor needs to have the ability to perform a variety of signal processing operations with enough fast speed. Regarding our X-ray line-scan camera system, the output data rate is not only specified by 20 mega bytes/ second, and there are also several real-time DSP Algorithms needs to be implemented in the pixel DSP processor: real-time filtering, and real-time fix pattern noise correction etc. The operations of addition or subtraction and multiplication are needs to be used for DSP functions.

Obviously, although the general-purposed processor has the advantages of low cost and easy implemented with advanced language, but its processing data rate is too slow to reach the specification, because the pixel data needs to be real-time processed with operations of subtraction and multiplication pixel by pixel.

The custom ASIC offers the advantage of highly optimized logic and dedicated metal interconnect. As such it can match or even get better the functionality and performance than FPGA and DSP processor with significant attendant cost reduction, but because non recurring engineering (NRE) costs for tools and masks are very high and very long timescale make it only suitable for medium and higher volumes production. ASIC is too expensive to implement on a small scale. And another disadvantage of custom ASIC is that it has very low flexibility to be redesigned. The X-ray line-scan camera is relatively low volume but high price system, and need to be designed flexibly to meet the requirements of different X-ray applications, so custom ASIC can’t be employed for pixel DSP processor.

Recent reductions in the cost of high-density FPGAs, combined with advances in software-oriented FPGA design tools, have led to a corresponding rise in the use of these devices to handle functions that are traditionally the domain of DSP processors, while at same time dramatically reducing the risks and up-front costs of custom ASIC solutions.

In this sense, the discussions above lead to get the conclusion that only FPGA and DSP microprocessor could be the possible solutions for the application.

### 3.2.3 Comparison between DSP Microprocessor and FPGA

A DSP processor is optimized for use of external memory, so a large data set can be used in the processing. FPGAs have a limited amount of internal storage so need to operate on smaller data sets, however FPGA modules with external memory can be used to eliminate this restriction [[[15]](#endnote-18)].

A DSP processor is designed to offer simple re-use of the processing units, for example a multiplier used for calculating an FIR can be re-used by another routine that calculates FFTs. This is much more difficult to achieve in an FPGA, but in general there will be more multipliers available in the FPGA.

If a major context switch is required, the DSP can implement this by branching to a new part of the program. In contrast, an FPGA needs to build dedicated resources for each configuration. If the configurations are small, then several can exist in the FPGA at the same time. Larger configurations mean the FPGA needs to be reconfigured – a process which can take some time.

The DSP can take a standard C program and run it. This C code can have a high level of branching and decision making – for example, the protocol stacks of communications systems. This is difficult to implement within an FPGA.

Most signal processing systems start life as a block diagram of some sort. Actually translating the block diagram to the FPGA may well be simpler than converting it to C code for the DSP.

When sample rates grow above a few mega hertz, a DSP has to work very hard to transfer the data without any loss. This is because the processor must use shared resources like memory busses, or even the processor core which can be prevented from taking interrupts for some time. An FPGA on the other hand dedicates logic for receiving the data, so can maintain high rates of I/O [[[16]](#endnote-19)].

Based on the descriptions above, the comparison of FPGAs and DSP processors is shown in Table 1 as follow.

Table 1 Comparison of FPGAs and DSP Processors

|  |  |  |
| --- | --- | --- |
|  | **FPGAs** | **DSP Processors** |
| **Max Clock Rate** | About 1 GHz | About 400MHz |
| **Max number of multipliers** | Normally 4 (16x16) | Over several hundrends |
| **Max instructions per clock** | 4 or 8 | 100s to 1000s |
| **Programming** | Hardware flow | Software flow |
| **Language** | VHDL or Verilog HDL | C or assembly language |
| **I/O flexibility** | Limited | Flexible |
| **Memory Management** | Built-In | Manual |
| **Difficulty of Design** | Relatively difficult | Easy |
| **Design Time** | Longer timescales | Shorter timescales |
| **Design flow** | Relatively complicated of design flow | Simple |
| **Work method** | Massive Parallelism | Less Massive Parallelism |
| **Cost** | Relatively High | Low |
| **DSP data rate** | High | Relatively low |
| **Difficulty for Customized Timing design** | Easy to implement | Difficult |

After reviewing Table 1, it can be known that FPGAs are suitable to perform mathematical operations on an entire vector or matrix at the same time. Furthermore, FPGAs are ideal for connecting multiple processing nodes together, distributing the data between digital signal processors and collecting and recombining the sub-calculations into a single output stream.

The ability of FPGA to manipulate the logic at the gate level means the user can construct a custom processor to efficiently implement the desired function. By simultaneously performing all of the algorithm’s subfunctions, the FPGA can outperform a DSP processor by as much as 1000:1.

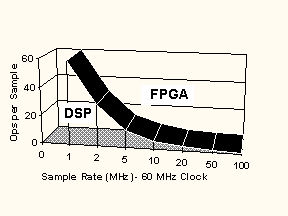


Figure 14 The comparison between FPGA and DSP based on the sample data rate [[[17]](#endnote-20)].

DSP performance is limited by the serial instruction stream. FPGAs are a better solution in the region above the curve. Note that actual performance gains depend on algorithm efficiency, clock rates, degree of parallelism and other factors. Typical gains lie between 10:1 and 1000:1

### 3.2.3 Final Conclusion

Based on the discussion in last section, FPGA should be the best solution for the pixel DSP processor. Without doubt, the highest performance (algorithm programmed into FPGA hardware) could be reached by using FPGA, but still the decision to use an FPGA would be on expense of flexibility, which increases design risk in most cases. The co-processing architecture could solve the problem by using lower-cost microprocessor as a co-processor, which could handle offloading computationally intensive work.

Figure 15 shows Altera's view of the DSP/FPGA solutions universe. It places special emphasis on co-processing.

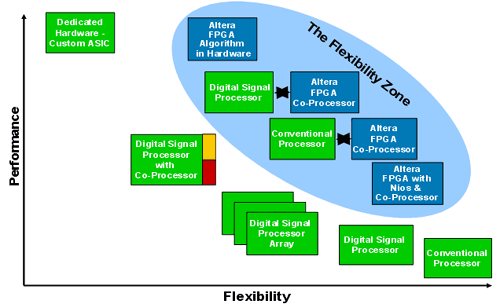


Figure 15 Using an FPGA as a coprocessor rates special attention [[[18]](#endnote-21)]

The area of most interest in Figure 15 is described by Altera as "the flexibility zone." It also defines the "profitability zone" for FPGA vendors and the "uncertainty zone" for design teams.

Finally, the combination of microcontroller (conventional processor) and FPGA is used in my control unit board design in terms of optimizing cost, time to market, and performance.

The requirement of data throughput is reached by using FPGA, the co-processor 8051 microcontroller is also employed to make the system configuration and offload computation.

## 3.3 Block-Diagram and Functional Description

As we discussed on the architecture of control unit board, the block diagram is shown as follow in Figure 16.

The pixel DSP processor is realized by using Altera Cyclone FPGA; The USB controller is implemented in the control unit board using Cypress’s EZ-USB FX2 chip, and the co-processor (8051 microcontroller) is also embedded in the same chip. To make the design easily, the LSI of hardware protocol stacks is chosen to implement the TCP/IP protocol in the system, and the fast Ethernet phyceiver is used together with the LSI as the physical layer chip. Using LSI of TCP/IP stacks and the phyceiver together is an easy, low-cost solution for high-speed Internet connectivity.

In the block diagram of control unit board, the 256K x 32 bits synchronous SRAM is also another key component, 133M or 150M Hz reading and writing speed make the system is possible to make the real-time DSP processing. The synchronous SRAM is divided into many 8K blocks that is called “ bank” in the system, so all of histograms of correction data and processed data finally are stored in the banks.

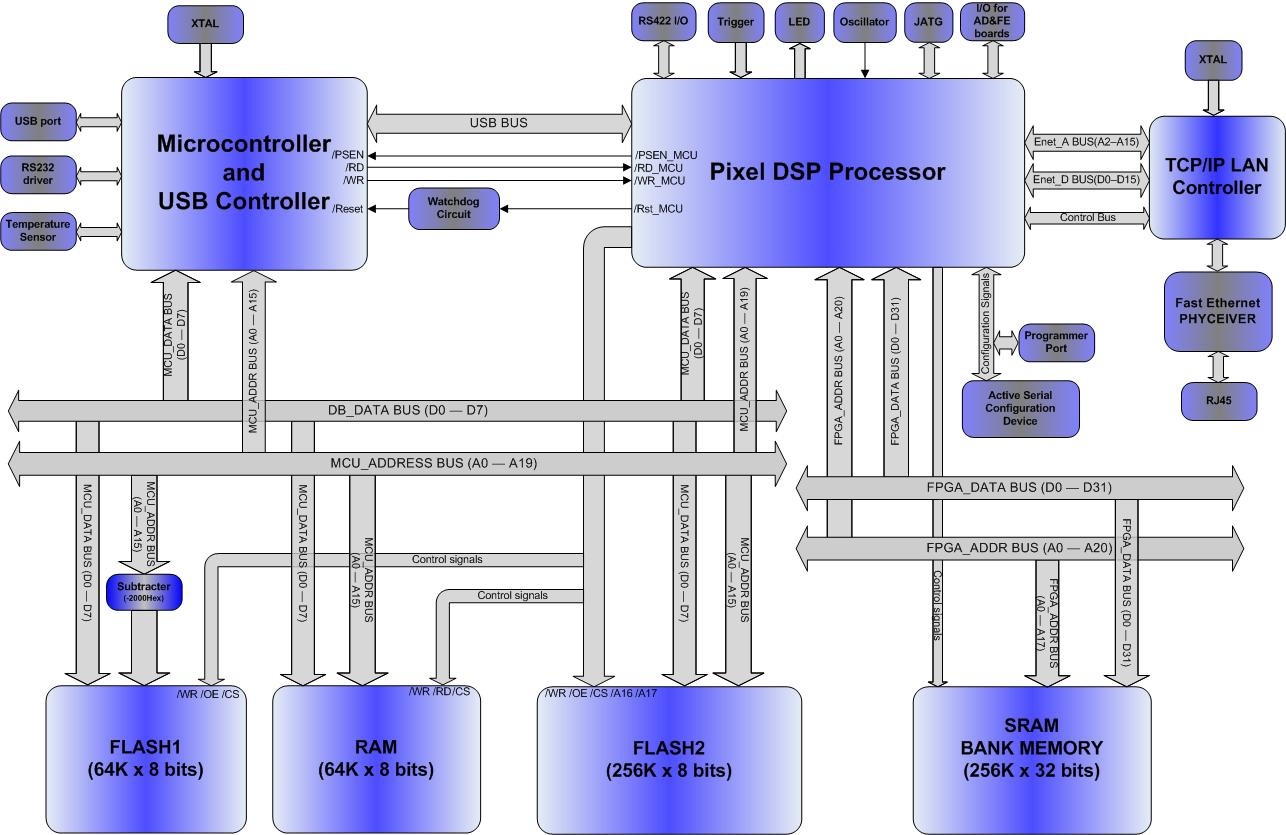


Figure 16 The top level block diagram of control unit board

### 3.3.1 Pixel DSP Processor

Based on the discussion in last section, in our system, FPGA is the best solution used to realize the pixel DSP processor.

To reduce the cost, the Altera Cyclone family FPGA is selected, because the Cyclone family is the low-cost devices with high-volume, application-focused features such as embedded memory, external memory interfaces, and clock management circuitry. Cyclone family FPGAs are also the optimal solution for high-volume, price-sensitive applications that previously required the use of fixed solutions such as gate arrays and standard cells.

According to the functionality requirement, a high-volume FPGA (EP1C12F324C7) is selected for the pixel DSP processor. The features of selected Cyclone device are shown in Table 2.

Table 2 Cyclone FPGA features

|  |  |
| --- | --- |
| **Feature** | **EP1C12F324** |
| Logic Elements | 12060 |
| M4K RAM blocks (128 x 36 bits) | 52 |
| Total RAM bits | 239616 |
| PLLs | 2 |
| Maximum user I/O pins | 249 |
| Package | 324-Pin Fine Line BGA |

In our system, the PLL is used to generate synchronized the clocks for the two or three clock domain design, because it is necessary to use different clock domains to control FE or AD board to meet the timing requirement for detector board and AD sampling. The PLL is employed to generate the FE\_CLK (40MHz) or AD\_CLK based on the 100MHz system clock. It makes the easy and stable synchronization between two different clock domains.

The internal M4K RAMs in the FPGA is very important in the design of the processor, the raw data buffer and the output buffer are built in the FPGA using M4K blocks, and these buffers are implemented as dual-port RAM to handle the pipelined buffer control.

The pixel DSP processor design is divided into several functional blocks at top-level; they are decoder, FE controller, output controller, SRAM interface and pixel processor. The block diagram of pixel DSP processor can be found in Figure 17 below.

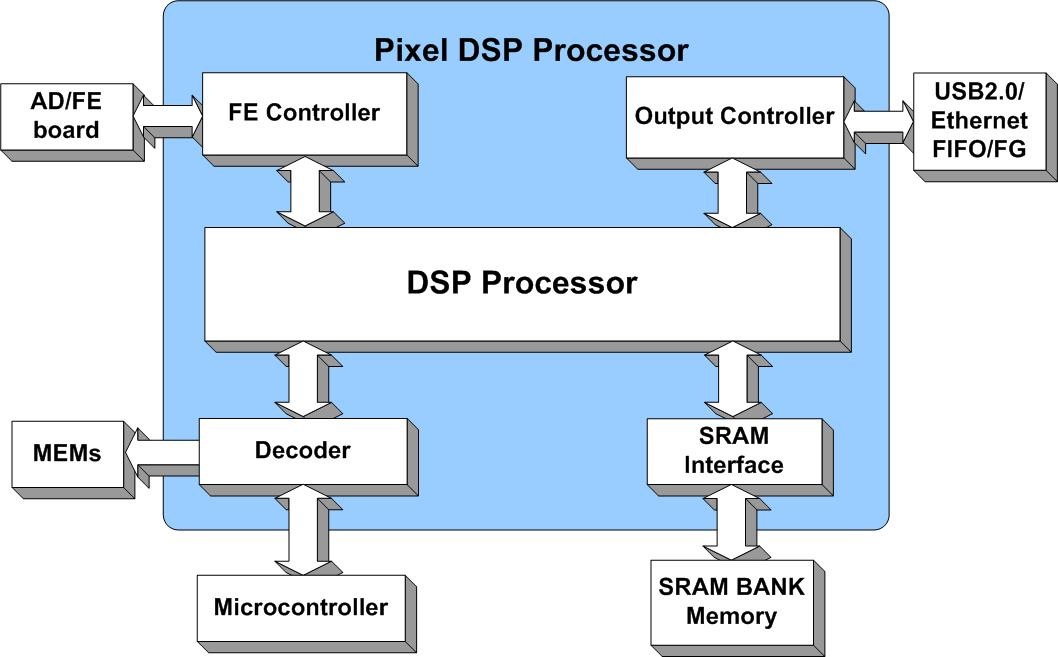


Figure 17 The block diagram of pixel DSP processor

In the design of the pixel DSP processor, the VHDL (VHSIC Hardware Description Language) is used for programming.

Since the architecture of the control unit board is based on the co-processors of FPGA and 8051 microcontroller, the decoder block provides the interface to communicate with 8051 microcontroller. In this component, the register file of pixel DSP processor is defined, and the handshakes between the pixel DSP processor and microcontroller are also decoded here.

On one hand, the FE Controller is responsible to generate all control timing to AD or FE board and collect the raw data from AD or FE board. On the other hand, it also saves the raw data to raw data buffer that is a dual-port RAM embedded in the FPGA chip using M4K RAMs according to the physical position of photodiodes in order. The control timing of this component is given by DSP processor that is the main control component in pixel DSP processor.

The output controller also has another embedded dual-port RAM that is so-called output buffer in the design of pixel DSP processor.

Generally, at least two lines data could be saved to the output buffer before it is sent to FIFO of USB controller or Ethernet chip buffer. In common sense, the more buffer, the better reliability, but how many RAMs can be used in the design is limited by the total size of embedded memory in FPGA chip.

We will introduce all the blocks as following.

#### 3.3.1.1 FE controller

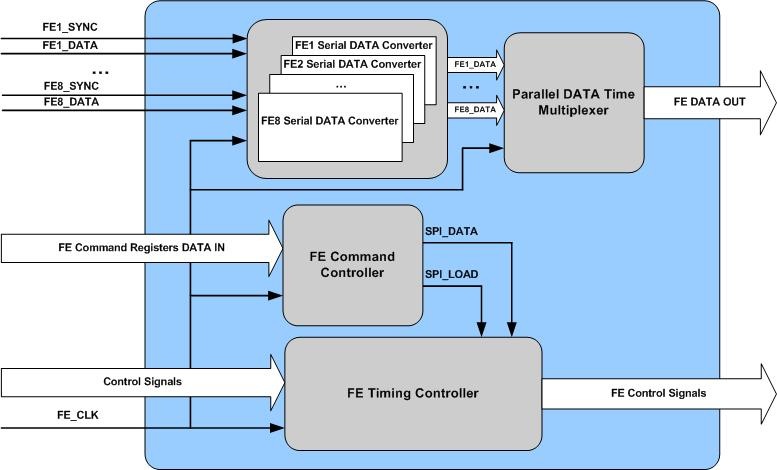


Figure 18 The block diagram of FE controller

The block diagram of FE controller is shown in Figure 18, it consists of serial data converter, parallel data time multiplexer, FE command controller and FE timing controller.

In our system, the raw data are sent from DM board to the control unit board using a serial link for each FE board, thus a data format converter from serial to parallel is needed in the FPGA. The serial data converter block is responsible for receiving the serial raw data from maximal 8 FE boards, and then converting the format to parallel, and then output the 8 parallel data to the parallel data time multiplexer block, in this block, 8 parallel data is multiplexed to one parallel data bus according to time division period.

In addition, FE controller is also responsible for sending control commands to FE boards so as to configure the mode, offset and gain of CCD processor (AD converter) according to the address of the FE board, and the sensitivity of detector cards also can be configured by the pixel DSP processor. The commands encoding is done by the FE command controller block, and the serial SPI link is used to send the commands.

The FE timing controller block is the control timing interface to FE boards. The system trigger, serial clock, data enable signals and SPI command links are sent from this block to FE boards.

#### 3.3.1.2 Decoder

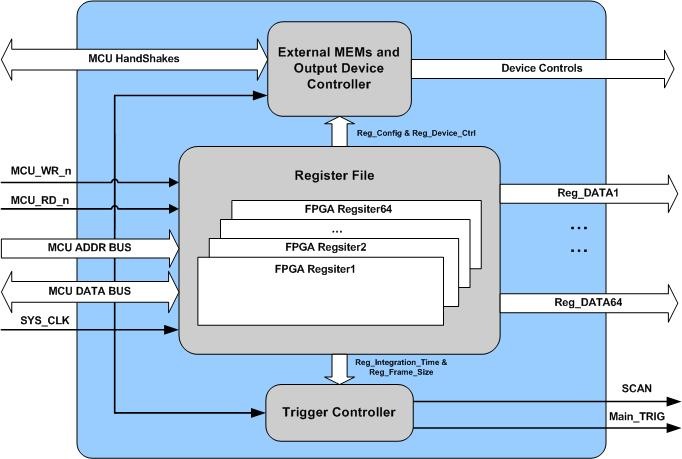


Figure 19 The block diagram of decoder

The main task of the pixel DSP processor is to perform the real-time DSP algorithms in the system, and the co-processor (8051 microcontroller) not only performs all of offload computations, and also is responsible for communication with host PC, but after the microcontroller receives the commands from host PC, it also needs to send the commands to the pixel DSP processor. The internal register file of FPGA is used for communication between FPGA and microcontroller.

The address, data bus and write and read signals of microcontroller are connected to the pixel DSP processor. The microcontroller interface is implemented in the decoder. With the interface, the microcontroller could access the internal registers in FPGA chip. At the same time, eight normal I/Os of microcontroller are also connected to FPGA for handshake signals between microcontroller and FPGA. Using the handshakes, microcontroller could not only select the external memories, and is also able to send or receive some special commands to or from FPGA.

As the block diagram of decoder is shown in Figure 19, the decoder could divide into three sub-blocks, they are trigger controller, register file and external memories and output device controller. The trigger controller block generates the main trigger and scan enable signals for the whole system, and its interval defines the integration time of the detector board. In the design of the pixel DSP processor, the main trigger is used to reset sub-blocks and synchronize start positions among different boards. There are total 64 registers reserved for communication between microcontroller and FPGA. With handshakes and the command decoded from a register, the external memories and different output interfaces could be enabled by FPGA using the external memories and output device controller.

#### 3.3.1.3 SRAM interface

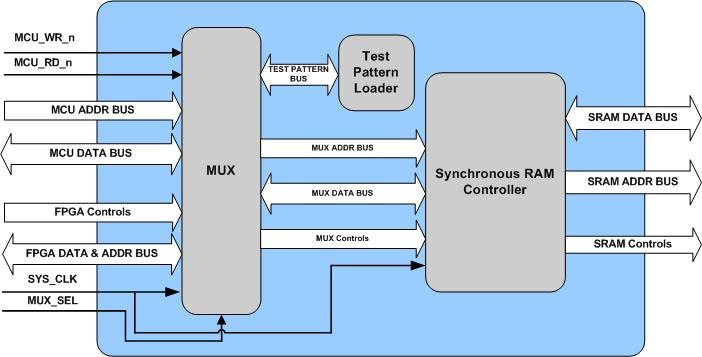


Figure 20 The block diagram of SRAM interface

In our system, the 133MHz synchronous SRAM is used to save the processed data and histogram of corrections. The synchronous SRAM is divided into thirty three 8K blocks that is called “banks”.

All of the raw data, processed data and correction values for DSP algorithms are stored in different banks. The correction values are calculated partly by microcontroller, after the system is power up. During the operation, the histogram of all data will be applied to the DSP processing, so the external SRAM interface has to be built for the microcontroller to access the correction values (offset and gain corrections). Since the bus of 8051 microcontroller is not compatible with the SRAM bus, a wrapper called SRAM controller is built to facilitate the communication between the 8051 microcontroller and SRAM banks.

Figure 20 shows the block diagram of SRAM interface. It consists of 3 sub-blocks; they are test pattern loader, MUX and SRAM controller. The SRAM controller is the key component that is used to interface with external synchronous SRAM. The test pattern is the gray scale values used to simulate the raw data for testing, and it could be loaded into the test pattern bank in the test pattern loader sub-block. The SRAM should be able to be accessed by both microcontroller and FPGA, thus the MUX sub-block is used to select which could access SRAM at a time.

#### 3.3.1.4 Output controller

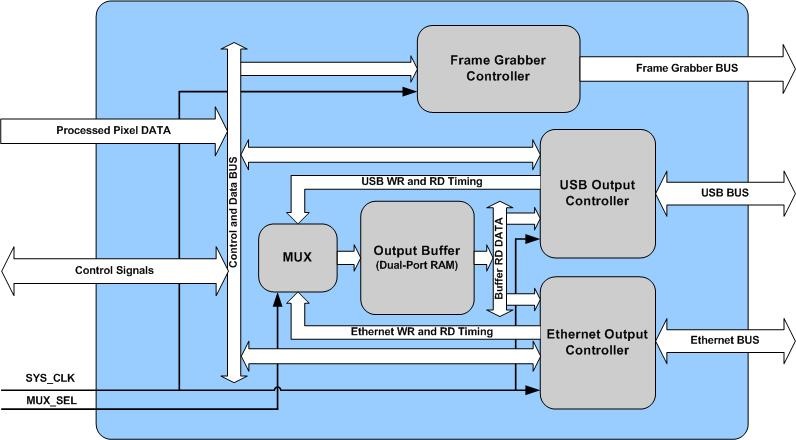


Figure 21 The block diagram of output controller

There are three image output interfaces could be selected by user in our system: the frame grabber, USB2.0 and the Ethernet interface. Only one of these output interfaces can be activated at one time, which can be defined by the user. The output controller block consists of these three output interfaces. In addition, a 4096 x 16 bits dual port RAM is built inside the block to act as an 1st level output buffer when data is transferred via the USB or Ethernet interfaces.

The block diagram of output controller is shown in Figure 21. It consists of frame grabber controller, USB output controller, Ethernet output controller, output buffer and a MUX. The frame grabber card is a high-performance PCI image acquisition board that interfaces with virtually all commercially available parallel output digital line and area scan cameras. It is quite powerful board to ensure high speed and carry out some DSP functions, so in this case, the buffer control has done in the frame grabber, the output data doesn’t need to make the buffer control, which can be sent to frame grabber directly.

The USB and Ethernet buffer control are implemented in this block. The MUX is used to select which interface controller could control the output buffer at a time. The output buffer is built by embedded M4K blocks, and it is a 4096 x 16 bits RAM with one write port and one read port.

The USB and Ethernet controller are not only responsible for generating all control timing to external USB or Ethernet chips, and also the timing to write and read the output buffer. USB and Ethernet controllers are able to control how to and when to load the data to external USB and Ethernet chips according to the full or empty flags of USB FIFO or Ethernet RAM.

How to ensure the data transfer without data lost is implemented in the block, we will describe the function in detail in next chapter.

#### 3.3.1.5 DSP processor

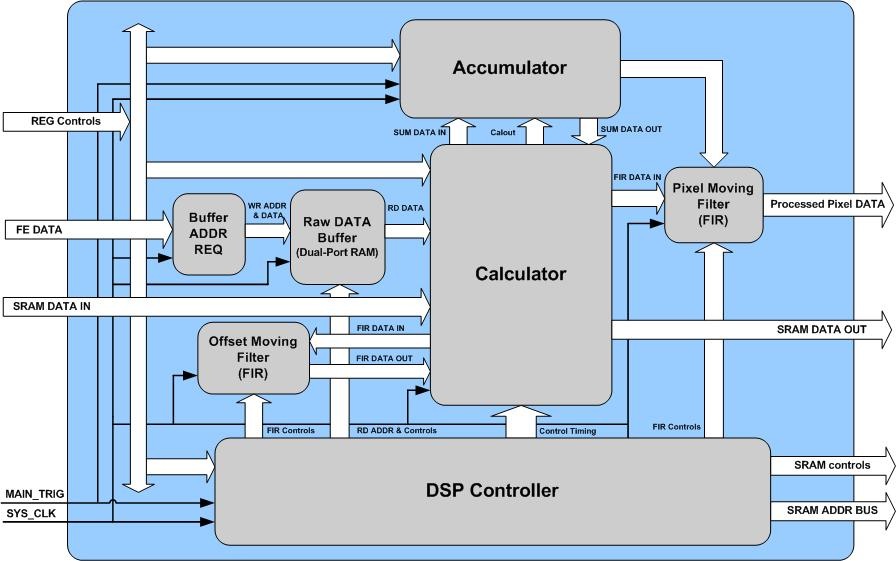


Figure 22 The block diagram of DSP processor

The DSP processor block is the most critical part of the pixel DSP processor design. All the DSP algorithms have been done in this block. The block diagram of DSP processor is presented in Figure 22. The block consists of DSP controller, accumulator, calculator, pixel moving filter Buffer ADDR REQ and raw data buffer.

First of all, the parallel multiplexed FE data from FE controller would be saved to the raw data buffer, the Buffer ADDR REQ sub-block is used to generate the re-sequenced address as the write address of the raw data buffer, so the read data from the buffer would be in order according to physical position of the photodiode array. Moreover, the raw data buffer is able to save two lines of data, the pipelined read and write timing are also controlled by the buffer ADDR REQ sub-block. Thus, when one line of raw data is written to the raw data buffer, simultaneously, the last saved line of re-sequenced data is read from the raw data buffer.

The DSP controller sub-block is responsible for generating all the timing signals to control the digital signal processing (DSP) process in calculator and accumulator, and finally the processed data is sent to output buffer in the output controller block.

The calculator is used to make real-time calculation to remove the FPN, four 32 x 32 multipliers and four 64 bits adders are used in the calculator. The accumulator is applied when the summing or averaging functions are enabled, and it also used to calculate the offset and gain values during on-board calibration. Before the calculated pixel data is sent out, the pixel moving filter needs to be applied to the data.

In our architecture, the external SRAM (bank memory) is used to save the histogram of the offset and gain corrections, so in the real-time processing the correction values would be loaded to the calculator, after the calculation, the result is saved back to one bank, and output to other blocks. The offset correction values need to be loaded first for offset moving average filtering to reduce the offset noise, after the calculation, the processed pixel data also needs to be applied to the filter.

The DSP controller also generates the timing to control external bank memory for real-time processing. The detailed process how to implement the moving filter and how to eliminate FPN in the FPGA design will be introduced in next chapter.

The DSP process includes:

1. Offset correction
2. Gain correction
3. Reference correction
4. Runt pulse detection and removal
5. Pixel averaging
6. Pixel summing

### 3.3.2 Microcontroller and USB Controller

In the design of control unit board, the microcontroller is employed as a co-processor for configuration of the pixel DSP processor, system initialization, some offload calculations and communication with host PC.

USB2.0, Ethernet and RS232 interfaces are used to communicate between X-ray line-scan camera and host PC. Since the control commands do not need to be real- time processed, it is better that these functions can be implemented into microcontroller, since the software-based design of the microcontroller is much easier than the hardware FPGA design.

In our X-ray line-scan camera, the Cypress EZ-USB FX2 (CY7C68013) is selected for USB controller, the FX2 (enhanced 8051) microcontroller is also embedded in the same chip, so the USB management is implemented in the microcontroller, which can easily access the internal USB registers and FIFOs. In addition, the extra microcontroller is not needed in the board, so it could save the space of CU PCB board.

The slave FIFO mode is used to transfer the image data using the pixel DSP processor. In this mode, the pixel DSP processor acts as a master to control the data transfer between EZ-USB FX2 and FPGA, so it could guarantee the high data rate.

#### 3.3.2.1 Introduction to EZ-USB FX2

Cypress’s EZ-USB FX2™ is the world’s first USB 2.0 integrated microcontroller. By integrating the USB 2.0 transceiver, SIE, enhanced 8051 microcontroller, and a programmable peripheral interface in a single chip, it has created a very cost effective solution that provides superior time-to-market advantages. The ingenious architecture of FX2 results in data transfer rates of 56 Mbytes per second, the maximum allowable USB 2.0 bandwidth. Because it incorporates the USB 2.0 transceiver, the FX2 is more economical, providing a smaller footprint solution than USB 2.0 SIE or external transceiver implementations. With EZ-USB FX2, the Cypress Smart SIE handles most of the USB 1.1 and 2.0 protocol in hardware, freeing the embedded microcontroller for application-specific functions and decreasing development time to ensure USB compatibility. The General Programmable Interface (GPIF) and Master/Slave Endpoint FIFO (8- or 16-bit data bus) provides an easy and glueless interface to popular interfaces such as ATA, UTOPIA, EPP, PCMCIA, and most DSP/processors [[[19]](#endnote-22)].

The block diagram of cypress chip is shown in Figure 23.

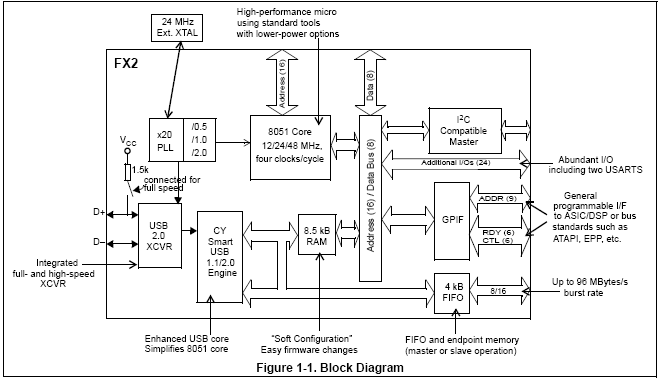


Figure 23 Block diagram of FX2 USB controller

### 3.3.3 Fast Ethernet Chips

In our X-ray line-scan camera, the 10/100M Ethernet interface also can be used to transfer the control and image data. As it is known, TCP protocol could provide reliable, stream-oriented connections between the host and client, and the reliability of X-ray line-scan camera is a very critical issue, so TCP protocol is quite suitable for this application. In order to design the system with an easy, fast and low-cost solution, the LSI of hardware protocol stack and fast Ethernet phyceiver chips are employed in the CU board.

#### 3.3.3.1 Introduction to LSI of hardware protocol stack

The i2Chip W3100A is an LSI of hardware protocol stack that provides an easy, low-cost solution for high-speed Internet connectivity for digital devices by allowing simple installation of TCP/IP stack in the hardware. The W3100A offers system designers a quick, easy way to add Ethernet networking functionality to any product. Implementing this LSI into a system can completely offload Internet connectivity and processing standard protocols from the system, thereby significantly reducing the software development cost. The W3100A contains TCP/IP Protocol Stacks such as TCP, UDP, IP, ARP and ICMP protocols, as well as Ethernet protocols such as Data Link Control and MAC protocol. The W3100A offers a socket API (Application Programming Interface) that is similar to the windows socket API. The chip offers Intel and Motorola MCU (8051, i386, 6811 tested) bus interface and I2C for upper-layer and supports standard MII interface for under-layer Ethernet.

The W3100A can be applied to handheld devices including Internet phones, VoIP SOC chips, Internet MP3 players, handheld medical devices, LAN cards for Web servers, cellular phones and many other nonportable electronic devices such as large consumer electronic products [].  
[[20]](#endnote-23)

The block diagram of i2Chip W3100A is shown in Figure 24.

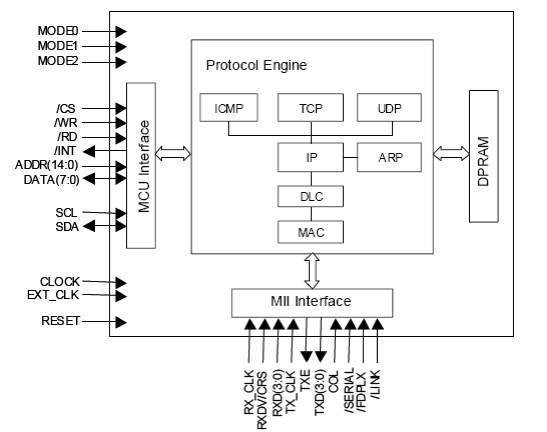


Figure 24 The block diagram of i2Chip W3100A

#### 3.3.3.2 Introduction to Fast Ethernet Phyceiver

The RTL8201BL is a single-port Phyceiver with an MII (Media Independent Interface)/SNI (Serial Network Interface). It implements all 10/100M Ethernet Physical-layer functions including the Physical Coding Sublayer (PCS), Physical Medium Attachment (PMA), Twisted Pair Physical Medium Dependent Sublayer (TP-PMD), 10Base-Tx Encoder/Decoder and Twisted Pair Media Access Unit (TPMAU). A PECL interface is supported to connect with an external 100Base-FX fiber optical transceiver. The chip is fabricated with an advanced CMOS process to meet low voltage and low power requirements.

The RTL8201BL can be used as a Network Interface Adapter, MAU, CNR, ACR, Ethernet Hub, Ethernet switch. Additionally, it can be used in any embedded system with an Ethernet MAC that needs a twisted pair physical connection or fiber PECL interface to external 100Base-FX optical transceiver module [].  
[[21]](#endnote-24)

The block diagram of RTL8201BL is shown in Figure 25.

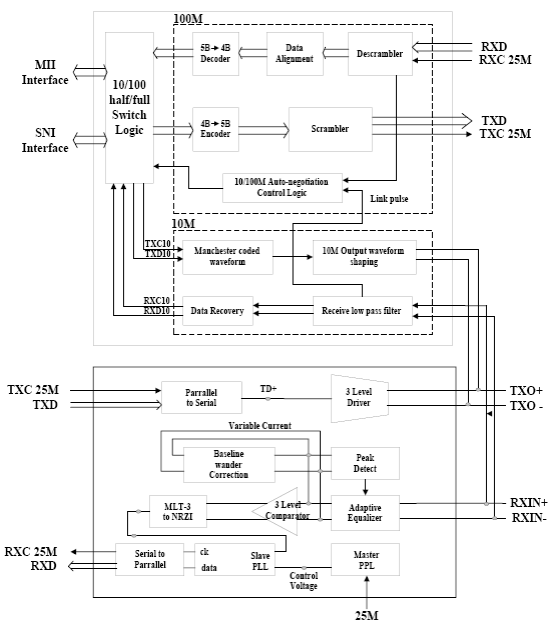


Figure 25 The block diagram of RTL8201BL

### 3.3.4 Memories and Other Components

#### 3.3.4.1 The synchronous SRAM

As we described in last sections, the synchronous SRAM acts as the “bank” memory in the system, and all the real-time correction values, calculation results and sum values are stored into different banks of the SRAM, so the SRAM is a critical memory in the system. The speed requirement of the system can be met that is not only depends on the arithmetic of DSP processor, and also needs the external SRAM has to be high speed memory, so the IDT71V65603 is employed as bank memory in the CU board.

The IDT71V65603 is 3.3V 133Mhz high-speed 9,437,184-bit (9 Megabit) synchronous SRAMS. It is designed to eliminate dead bus cycles when turning the bus around between reads and writes, or writes and reads. Thus, they have been given the name ZBTTM, or Zero Bus Turnaround. Address and control signals are applied to the SRAM during one clock cycle, and two cycles later the associated data cycle occurs, be it read or write [].  
[[22]](#endnote-25)

The IDT71V65603/5803 contains data I/O, address and control signal registers. Output enable is the only asynchronous signal and can be used to disable the outputs at any given timing [24].

#### 3.3.4.2 Other memories

A 256 x 8 bits flash memory is used to save the system parameters and the normalization gain and offset corrections, the flash is used by microcontroller, however the address bus of microcontroller is 16 bits, so the page of the flash is controlled by FPGA.

A 64 x 8 bits flash memory acts as a code ROM for microcontroller in the CU board, and a 64 x 8 bits RAM is used to be the data RAM for both microcontroller and the pixel DSP processor.

Moreover, there are three external memories can be accessed by microcontroller, the write, read and chip enable signals are controlled by FPGA. At a time, only one external memory can be activated for microcontroller.

#### 3.3.4.3 RS232 driver

A one channel RS232 driver is connected to the microcontroller. As one option of control interface, X-ray line-scan camera could communicate with host PC using RS232 link.

#### 3.3.4.5 Digital temperature sensor

A digital temperature sensor is connected to microcontroller using I2C links, so the temperature could be monitored in the camera by host PC software.

## 3.4 Conclusion

Again, low cost, high flexibility, high performance and excellent reliability are the design targets for the X-ray line-scan camera, as the key component of whole system, the control unit board needs to implement all of requirements.

Based on the architecture discussion and functional descriptions of control unit board design in last section, we could conclude the design of control unit board.

In the control unit board, finally, the co-processors architecture is employed: FPGA is selected as the pixel DSP processor to perform real-time digital signal processing, and the 8051 microcontroller is used to implement offload calculations, configuration of the pixel DSP processor and communication with host PC.

How to meet target requirements could be clarified below:

### 3.4.1 Low Cost

Based on the co-processor architecture, the family of low cost and high density Cyclone FPGA is employed in the design, and moreover, the embedded M4K blocks are used as the raw data buffer and 1st output buffer. It is not only easy to manipulate by FPGA design, and also could save cost for external dual-port memories and PCB area.

The commercial frame grabber board is very expensive, the Ethernet and USB2.0 interfaces are supported to let customer to have the possibility to develop their own software without frame grabber card.

### 3.4.2 High Flexibility

The co-processors architecture makes the system much easier to be designed and modified; the system design then could be divided into software-based microcontroller and hardware-based FPGA design. Only real-time DSP process and control timing need to be designed in FPGA, and the register file of the pixel DSP processor is the simple and easy interface between FPGA and microcontroller.

In system programming (ISP) function is supported for the design, so user could reprogram FPGA and microcontroller by using host PC software.

The pixel DSP processor design is target as good scalability and adaptability. Depending on the different user requirements, the CU board could connect with eight FE boards at most. Moreover, the 16-bit parallel data and 8-bit control reserved connectors could be used to connect extension board to connect more FE boards.

The three interface options could meet the requirements from most customers. Customers can select the suitable interface according to the required data rates, cable lengths and costs from their specific applications.

### 3.4.3 High Performance

High data rate could be reached on the FPGA-based design together with 100MHz oscillator and 133/150 MHz synchronous SRAM, including DSP processing, the output data rate of the pixel DSP processor could reach to 20 Mbytes/second, and it meets the specification.

The DSP algorithms are applied in the pixel DSP processor that could eliminate the FPN, reduce the noise and enhance the signal level. Finally, the S/N could be improved by using the pixel DSP processor.

#### 3.4.4 Excellent Reliability

TCP/IP protocol is employed with Ethernet interface to guarantee reliable, stream-oriented connections between X-ray line-scan camera and host PC.

The 1st level output buffer and 2nd level bank memory are built in the pixel DSP processor, and they are used together to improve the reliability for image transfer.

Error control has done in the software, for example, the software could report the error message when data lost occurs.

# 4. IMPLEMENTATION OF THE PIXEL DSP PROCESSOR

## 4.1 Overview

The output signal from detector card is still an analog signal, so it cannot be recognized by the computer. Therefore it cannot be seen by human eyes or processed by computer software. This requirement leads to the conclusion that the sampled signals should be quantized to get digital signals, and be processed to improve the performance, finally be sent to user computer with USB2.0 or Ethernet or frame grabber card interfaces.

Digital signal processor can be programmed to perform a variety of signal processing operations, such as filtering, spectrum estimation, and other DSP algorithms. Depending on the speed and computational requirements of the application, the digital signal processor may be realized by a general-purposed processor, FPGA, custom ASIC or special purpose DSP device. In last chapter, we get the conclusion that FPGA should be the most suitable for the pixel DSP processor in X-ray line-scan camera.

The simplified signal processing diagram for an X-ray line-scan camera is shown in Figure 26.



Figure 26 A simplified signal flow in X-ray line-scan camera

The quantized signals will be processed for three purposes:

1. Correct the fixed pattern noise (non-uniformity) from detector cards.

2. Apply the FIR filter to offset, reference and pixel output values to improve the signal to noise ratio and dynamic range;

3. The timing signals for the whole system are designed to organize the data flow and image grabbing.

In our system, FPGA together with memories, buffers and line drivers build the digital signal processing system. So in this chapter, to implement the pixel DSP processor, digital signal processing algorithms and practice for the X-ray line-scan camera are discussed, and the output control for USB2.0 and Ethernet interfaces is also studied. All the analyses and algorithms are realized by hardware and measured results are presented in this chapter.

## 4.2 Noise Reduction and Signal Enhancement

In the X-ray line-scan camera system, generally, there are two main categories of sources, which generate noise in the system [[[23]](#endnote-26)].

1. Fixed pattern noise

It consists of the dark noise and read out noise (e.g. due to internal switching). The fixed pattern noise is constant when the read out conditions are fixed and the integration time is not changed. So it is possible to eliminate it from the measured signal.

2. Random noise

It is traceable to erroneous fluctuations of voltage, current or electrical charge, which are caused in the signal output process. If the fixed pattern noise is subtracted, the random noise determines the lower limit of light detection or the lower limit of the system dynamic range.

In order to get accurate data with least possible noise and distortion, fast digital signal processing logic has been profoundly implemented in the X-ray line-scan camera in this thesis.

The fixed pattern noise in the X-ray line-scan camera can be and must be removed, then the object in scanned image can be identified by computer software or human eyes. Since the real-time requirement of the line-scan camera, the FPN correction has to be done in pixel DSP processor instead of in computer software. The gain correction and offset correction to eliminate the FPN problem are also implemented in the pixel DSP processor. The algorithm to correct the FPN is presented in section 5.2.1.

The random noise can be reduced not only by analog circuit improvement, and also by the DSP functions, for example, digital filtering. The digital signal processing theory and hardware realizations will be discussed in next sections, which can be verified with field measurements. The sample pictures will show that the digital filter can improve the noise.

The algorithm to reduce the random noise and enhance the signal level will be presented in section 5.2.2.

### 4.2.1 Fixed Pattern Noise Correction

Fixed pattern noise (FPN) refers to a non-temporal spatial noise and is due to device mismatches in the pixels, variations in amplifiers, and mismatches between multiple PGAs and ADCs. FPN can be either coherent or non-coherent. Dark current FPN due to mismatches in pixel photodiode leakage currents tends to dominate the non-coherent component of FPN, especially with long integration time (exposure time). Again, as with pixel dark current shot noise, low leakage photodiodes are preferable to reduce this FPN component. Dark frame subtraction is an option, but this tends to increase the read-out time of the sensor. The most problematic FPN in image sensors is associated with easily detectable (or coherent) row-wise artifacts due to mismatches in multiple signal paths, and un-correlated, row wise operations in the image detector. Coherent FPN offset components can generally be eliminated by reference frame subtraction. Gain mismatches are more difficult to remove since they require time or hardware intensive gain correction (e.g. multiplication) [[[24]](#endnote-27)].

In our system, the FPN correction is done by using the hardware adders and multipliers in the pixel DSP processor design, and the external 133MHz synchronous SRAM is also used to save and load the histogram of offset and gain correction values for real-time processing. The DSP data flow diagram of pixel DSP processor is presented in Figure 27.

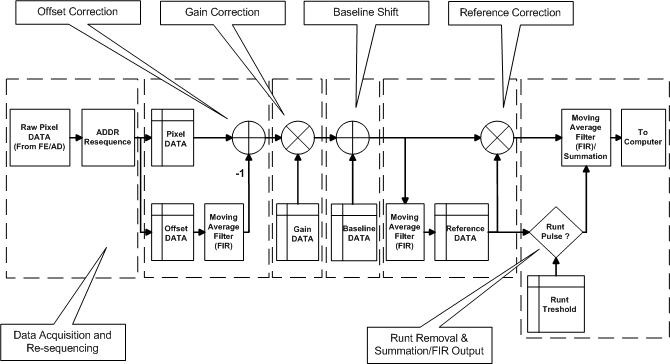


Figure 27 The DSP data flow diagram of the pixel DSP processor

The DSP data flow could be divided into six parts: they are data acquisition and re-sequencing, offset correction, gain correction, baseline shift, reference correction and runt removal and summation/FIR output.

The following symbols are used.

* Pxx[n,p] Pixel signal data, line number n, pixel value p
* Pxx[n] Pixel signal data , line number n (all pixel values)
* Oxx[n] Offset data, line number n

#### 4.2.1.1 Data acquisition and re-sequencing

This block also generates the main timing signals to organize the whole digital signal process.

The signal data, as output from the detector modules is not in the correct order for an image line and therefore it has to be reorganized before further processing, so the pixel DSP processor first acquires the data from the detector modules and makes the data re-sequencing, then save the data to raw data buffer (dual-port RAM). The data acquisition is controlled by the pixel DSP processor. There are two line data could be stored in the raw data buffer, so one line data is being saved to the buffer according to physical order of photodiode array, while another line data that is in order is being loading to processing block for further DSP processing. The pipelined data acquisition and loading makes data saving and loading in parallel.

In this system, the offset values are processed when the pixels are being read. So it is a parallel system.

The detailed procedure for re-sequencing the data is presented in last chapter.

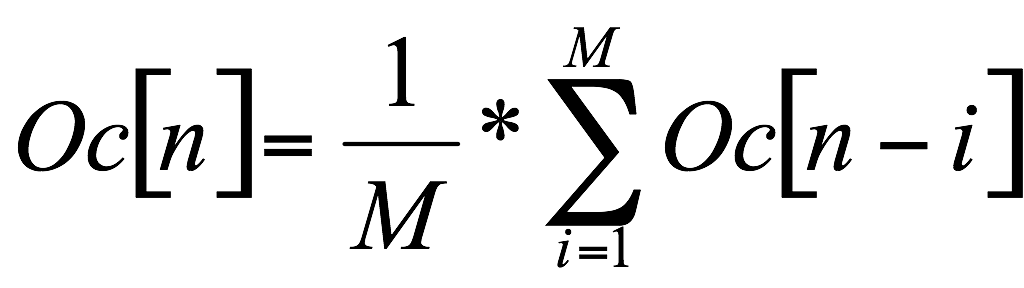
#### 4.2.1.2. Offset correction

The photodiodes, front-end amplifiers and ADCs have a certain degree of offset differences, which are accumulated to the actual output signals. The offset values must be subtracted from the actual signals. The moving average of the offset value of each channel is used for the offset correction of that individual channel (Oc1[n]).

The FIR filter is used for offset correction value calculation. We will present details of why the FIR filter is presented and how it is implemented in hardware in next section.

The offset values are first updated to the latest bank (The synchronous SRAM (bank memory) is divided into 4k units, and it is so called the units as “bank”). and then the offset value will be processed (user can specify the TAP values of the FIR filter) and the result will be saved to the specified bank.

The maximum size (m) of filter taps is limited to 11 due to memory size limitations. The offset corrected value is calculated as follows:



Where, M is size of the moving average filter. The offset corrected pixel value will therefore be:

PC1[n]= Praw- Oc1[n]

Where, the Praw is the raw data that is loaded from raw data buffer.

#### 4.2.1.3. Gain correction

With X-ray, pixel-to-pixel signal variations are mainly caused by scintillator crystal material variations, crystal-to-photodiode coupling variations and gain variations between the channels. These variations should be corrected by calibration. Calibration is performed by taking an offset scan (radiation off), an air scan (radiation on, no object) and then calculating the individual pixel variations when compared to the average of the complete image line. Correction factors for the each pixel are calculated and correction values are stored in specific banks. The gain-corrected line of pixel data is then

PC2[n,p] = PC1[n,p]\*G[p]

Where , p = 0… 1023 (for 1024 channels).G[p]is the correction factor.

#### 4.2.1.4. Baseline shift

The calibrated pixel data without X-ray would drop to around zero, in the digital system, after the processing block, the zero data would be outputted when the calibrated pixel data is less than zero, so the baseline shift is needed when calibrated pixel data without X-ray needs to be analyzed in software. The baseline value could be specified by user using the special register in FPGA. Then the shifted line of pixel data is:

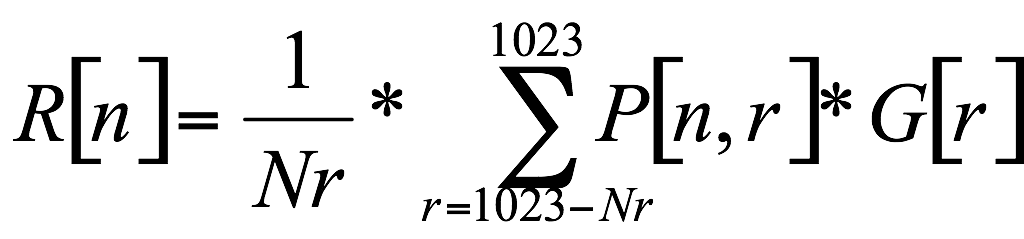
PC3[n,p] = PC2[n,p] + B

Where, the B is a constant for baseline shift, but it can be changed by user.

#### 4.2.1.5. Reference correction

LINAC is short for the term linear accelerator. Linear accelerator machines produce radiation that is referred to as high energy X-ray.

The signal amplitude from LINAC will vary from pulse to pulse. The signal variations caused by LINAC are corrected by measuring the LINAC pulses by reference channels, which are always measuring direct LINAC radiation output without target object between the source and the detector. The reference correction factor is calculated by taking an average of the reference channels and dividing the value with the nominal reference signal value set by the host PC. (Actually this can be done in gain correction phase by replacing the gain correction factor of the reference channels with the inverse of the nominal reference value). The nominal reference value is the average signal of all the reference channels taken in an air scan (radiation on, no object) during the calibration.



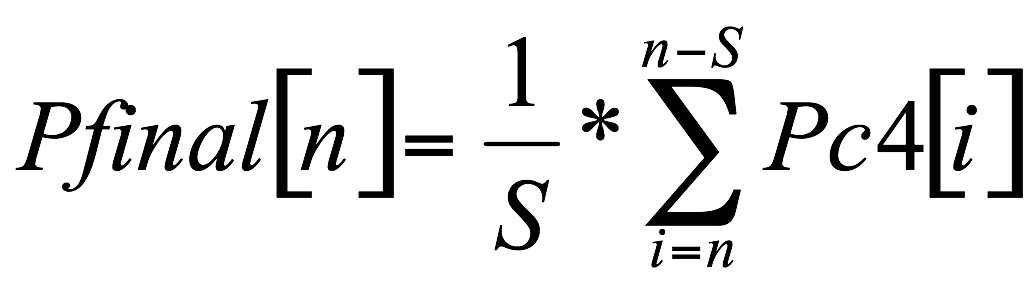
Where Nr is the number of reference channels.

Reference corrected pixel values are then calculated as follows:

PC4[n,p] = PC3[n,p]\*(1/R[n]) p=0….(1023-Nr),

#### 4.2.1.6. Summation and runt pulse detection

In high-resolution mode of the system, the image line shown is calculated as an average of a programmable number of successive LINAC pulses. The final pixel values (image pixel data to be sent to the frame grabber) are calculated as follows:

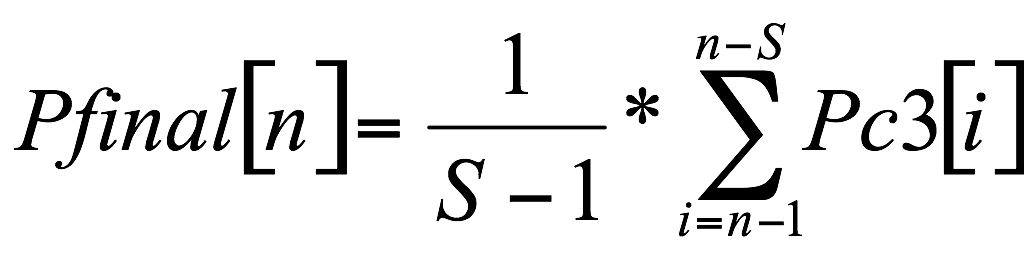
 , where S is the number of lines to be averaged.

In case of a runt LINAC pulse (misfire in LINAC) the corresponding line should not be taken into account when calculating Pfinal. A runt pulse is detected by comparing the reference channel values to a runt pulse threshold value given by the host PC.

If R[n]> Rth , Rth = runt pulse threshold value.

If the average reference is smaller than the threshold, then this pixel will be ignored. Otherwise it will be sent to the correction process.

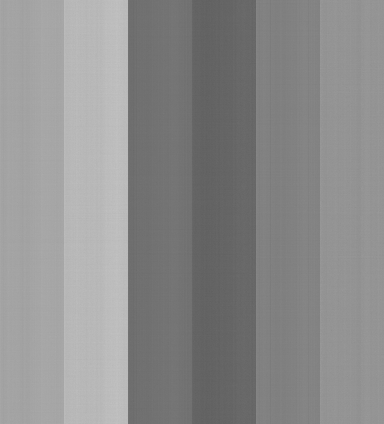
Final pixel values should then be calculated as follows:



Finally, the final output data will be filtered again (the TAP values of the FIR filter is also can be specified by user). After that, the data will be sent to output controller.

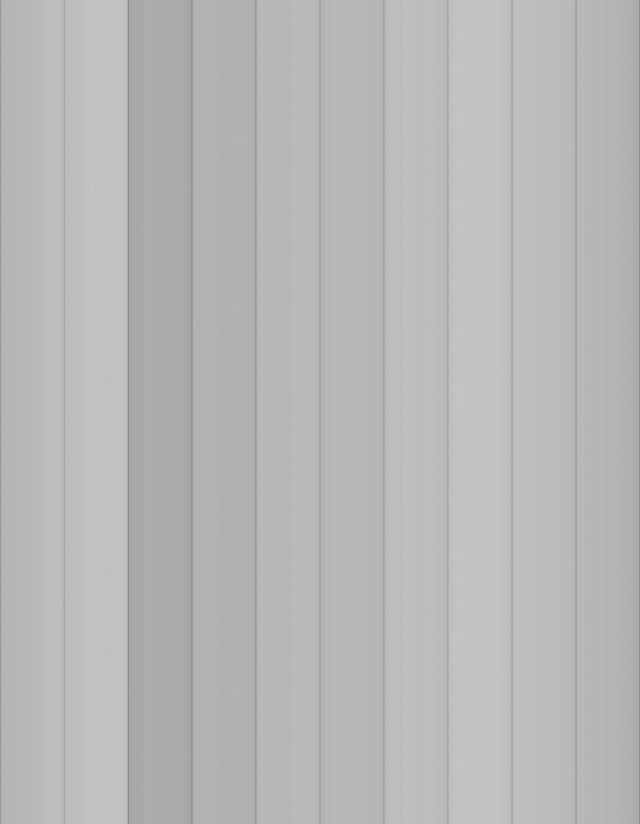
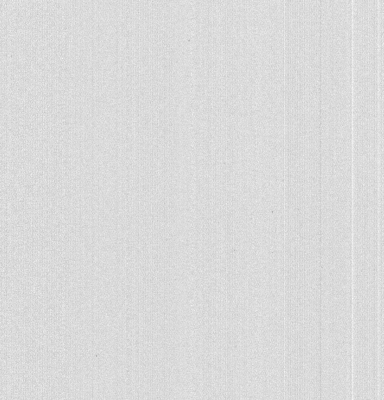
Figure 28 shows the result of offset correction. Figure 29 shows the result of gain correction. From the pictures it can be seen that the stripes of the uncorrected background are completely removed by the digital processor. All the non-uniformities caused by the sensors now are corrected and invisible to human eyes.

Figure 30 is a sample image made with the X-ray line-scan camera. Figure 30 (a) is a telephone’s X-ray image made without calibrations. Figure 30 (b) is the image made with calibration. From the comparison it can be seen that the fixed pattern noise of the detectors is totally removed by the digital correction.

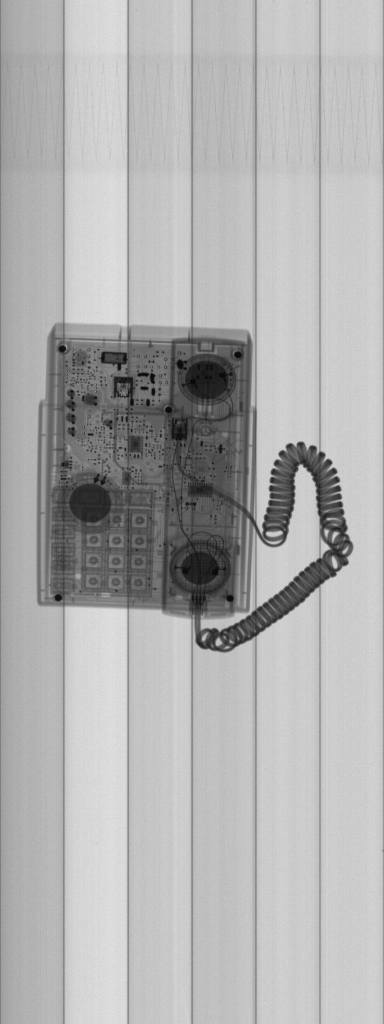
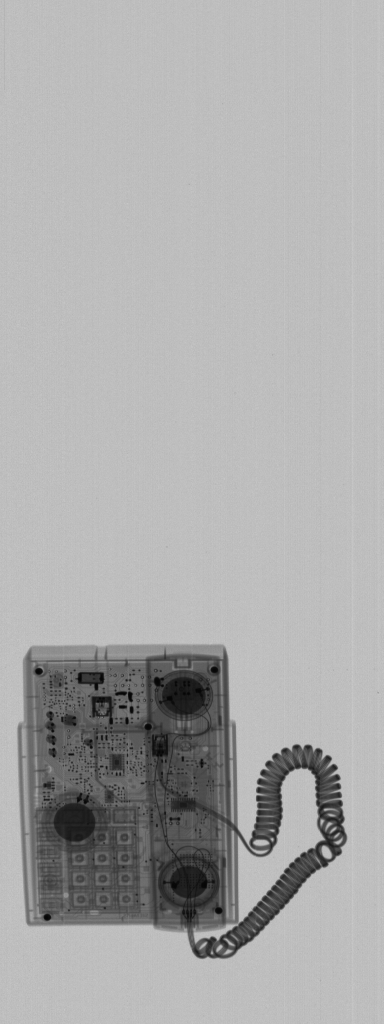
1. (b)

Figure 28 Offset calibration (a) a blank image without offset calibration (b) a blank image with offset calibration.

1. (b)

Figure 29 Gain calibration (a) a blank image without gain calibration (b) a blank image with gain calibration.

(a) (b)

Figure 30 X-ray scanning images (a) without calibration (b) with calibration.

### 4.2.2 Noise Reduction Ratio

One of the most common problems in signal processing is to extract a desired signal, say *s(n)*, from a noisy measured signal

*x(n) = s(n) + v(n)* (5-1)

where *v(n)* is the undesired noise component of the signal.

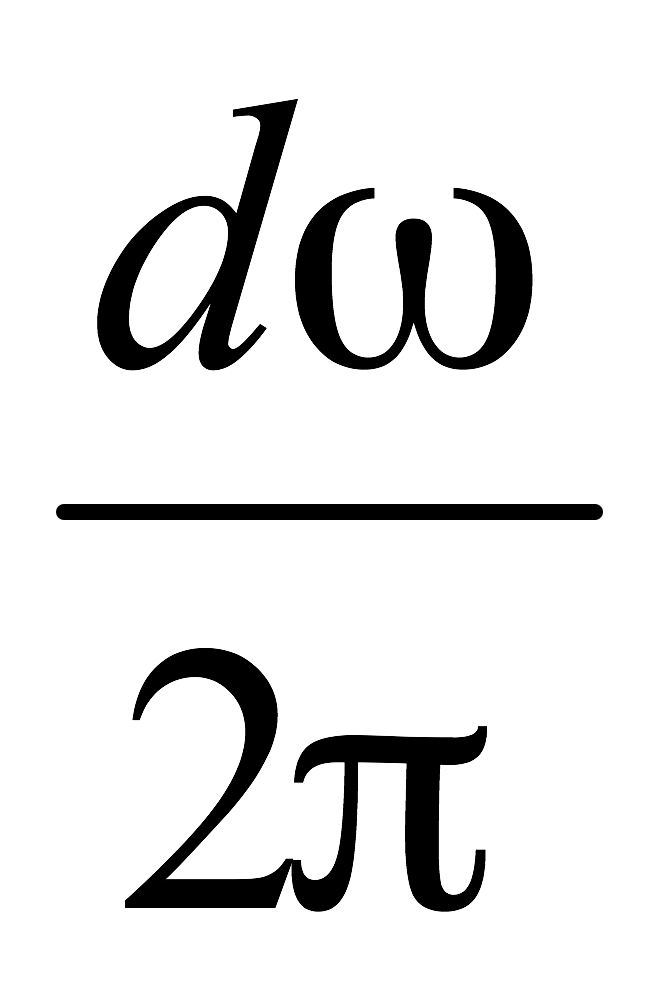
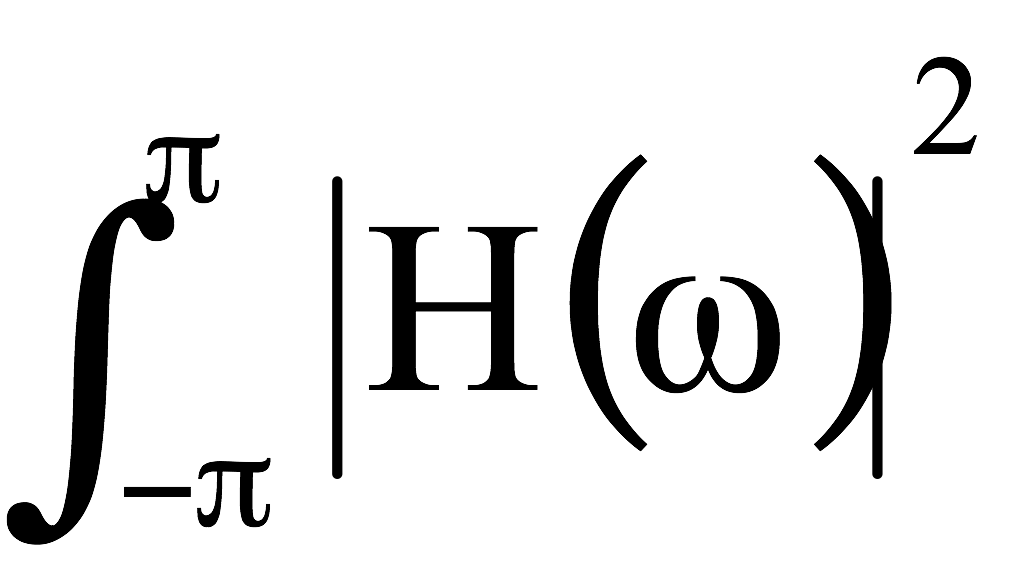
The standard method of extracting *s(n)* from *x(n)* is to design an appropriate filter *H(z)* which removes the noise component *v(n)* and at the same time lets the desired signal *s(n)* go through unchanged. This process can be expressed as

*y(n) = ys(n) + yv(n)* (5-2)

where *ys(n)* is the output due to *s(n)* and *yv(n)* is the output due to *v(n)*.

In the smart sensor micromodule, the significant 1/*f* noise is reduced to much lower level by the CDS technique. So the main noise to be reduced is white noise. A properly designed digital low-pass filter can greatly reduce the noise for the smart sensor micromodule.

Suppose the transfer function of the digital low-pass filter is H(ω), the coefficient of the transfer function in time domain is hn, then the amount of noise reduction achieved by the filter can be calculated using the noise reduction ratio (NRR), it is defined as below

*NRR* = *2yv/2v = E[yv(n)2]/E[v(n)2] = = (hn)2* (5-3)

where *2yv = E[yv(n)]2* is the output mean-square noise and *2v = E[v(n)2]* is the input mean square noise, hn is the coefficient of the filter transfer function.

The signal-to-noise ratio at the input and output of the filter is defined in terms of the mean-square values as

*SNRin = E[s(n)2]/E[v(n)2], SNRout = E[ys(n)2]/E[yv(n)2]*  (5-4)

Therefore, the relative improvement in the SNR introduced by the filter will be

*SNRin/ SNRout =* *E[ys(n)2]/E[yv(n)2]\*E[v(n)2]/E[s(n)2]* = *1/NRR \* E[ys(n)2]/E[s(n)2]* (5-5)

If the desired signal is not changed by the filter, ys(n) = s(n), then

*SNRout/SNRin =* *1/NRR* (5-6)

From above equations it can be seen that a smaller NRR will result in a larger S/N ratio.

### 4.2.3. FIR Moving Average Filter to Reduce White Noise

There are two ways to realize a digital filter, they are: FIR (finite impulse response) filter and IIR (infinite impulse response) filter [10]. Each method has its own advantage:

1. Complexity: IIR smoother is simpler computationally than FIR, requires only 2 multiplier accumulators (MACs). However, an N-order FIR type filter requires N MACs. So FIR filter will take more hardware resources.

2. Performance: The FIR smoother performs better than IIR in terms of both the NRR and the transient response. In the following section we will see that it can achieve unlimited noise reduction capacity if we use unlimited number of N. Of course, with the increase of N, the signal bandwidth will also be limited. FIR filter also has better transient response.

3. A further advantage of the FIR smoother is that it is a linear phase filter.

So FIR smooth filter is chosen as the digital processor. The transfer function to reduce the broadband white noise of the detectors is deducted as follows:

We may first consider, for example, a seven-order low-pass filter, so

*H(z) = h0 + h1 z-1 + h2 z-2 + h3 z-3 + h4 z-4 + h5 z-5 + h6 z-6 + h7 z-7* (5-7)

where all parameters also are determined by requiring unity gain at DC:

*H(1) = h0 + h1+ h2 + h3 + h4 + h5+ h6 + h7* = 1 (5-8)

*NRR =* *2yv/2v =(hn)2 = (h0)2+(h1)2+(h2)2+(h3)2+(h4)2+(h5)2+(h6)2+(h7)2* (5-9)

Finding the minimum value of (5-5) under the condition of (5-4), we have

*h0 = h1= h2= h3 = h4 = h5= h6= h7* = 1/8 (5-10)

So the optimal smoothing filter becomes

*y(n) = [x(n)+x(n-1)+x(n-2)+x(n-3)+x(n-4)+x(n-5)+x(n-6)+x(n-7)]/8* (5-11)

Extending to general condition, the optimum length-N FIR filter with unity DC gain and minimum NRR is the filter with equal weights

*hn = 1/N, n = 0, 1, …, N-1* (5-12)

and I/O equation is

*y(n) = [x(n) + x(n-1) + x(n-2) +…+ x(n-N+1)]/N* (5-13)

Its NRR is

*NRR = (h0)2 + (h1)2 +…+ (hN-1)2 =1/N*  (5-14)

Thus, an N-order FIR type smoother can reduce broadband white noise to a very low level if we use large N value. Of course, a large value of N has larger time constant, which means longer transient time and narrower signal bandwidth. So it can be seen that the white noise voltage reduction ratio is proportional to the square root of N in a statistical sense. It is the suitable digital filter to remove the white noise from the signal of the detectors.

### 4.2.4. Hardware Realization of the Digital Filter

There are many digital signal processing algorithms; however, all of them are limited by the resources. One of the resource limitations is hardware limitation. With the development of microelectronic technology, very large-scale digital circuits can be implemented in a small BGA package. The size and cost of memory chips are also affordable. So the hardware doesn’t bring too much limitation for our system. The other limitation is processing speed. In order to achieve a fast operating speed, the number of clock cycles for the processing of one pixel is also limited. So an efficient processor must be designed to meet the speed requirement.

A generic FIR filter’s hardware architecture is shown in Figure 31. The tapped delay line registers wi are sequential RAM located on board. The filter weights hi, together with the instructions of the filtering algorithms, are all stored in the DSP chip.

*h0*

*h1*

*h2*

*h3*

*h4*

*h5*

*h6*

*h7*

*w0*

*w1*

*w2*

*w3*

*w4*

*w5*

*w6*

*w7*

ROM or RAM

RAM

y

x

y

*wi*

MAC

*hi*

*hiwi*

BUS

IN

OUT

Figure 31 Architecture of a FIR filter.

The workhorse of the DSP chip is an on-board multiplier accumulator [10], which implements the dot product multiplications/accumulations, that is, the operations

*y <= y + hiwi* (5-15)

As discussed in section 5.2.2, the FIR-type filter needs memory buffers to hold the internal states. A straightforward way to implement the memory is to use linear delay line buffer. At each time instant, the data in the delay line are shifted one memory location ahead. The other way is the circular delay-line buffer. Instead of shifting the data forward while holding the buffer addresses fixed, the data are kept fixed and the addresses are shifted backwards in the circular buffer. The relative movement of data versus addresses remains the same. To understand this, consider first the conventional linear delay-line buffer case, but wrap it around in a circle, as shown in Figure 32 for the case M = 8.

Going from time n to n+1 involves shifting the content of each register counterclockwise into the next register. The addresses of the eight registers {*w0, w1, w2, w3, w4, w5, w6, w7*} remain the same, but now they hold the shifted data values, and the first register w0 receives the next input sample *xn+1*. By contrast, in the circular buffer arrangement shown in Figure 32, instead of shifting the data counterclockwise, the buffer addresses are decremented, or shifted clockwise once, so that *w3* becomes the new beginning of the circular buffer and will hold the next input *xn+1*.

In both the linear and circular implementations, the starting address of the state vector is the current input sample, and from there, the addresses pointing to the rest of the state vector are incremented counterclockwise. But, whereas in the linear case the starting address is always fixed and pointing to *w0*, as shown in Figure 33, the starting address in the circular case is back shifted from on time instant to the next. To keep track of this changing address, we introduce a pointer variable **p** that always points to the current input, as shown in Figure 33. By moving the pointer **p**, every time only one register or memory cell needs to be updated. This action can be finished only in one clock cycle. So the circular delay line buffer has the benefit of high efficiency and high speed, which is suitable for the X-ray line-scan camera digital signal processing.

*xn+1*

*xn-6*

*xn-5*

*xn*

*xn-4*

*xn-1*

*xn-2*

*xn-3*

**p**

*w1*

*w2*

*w3*

*w4*

*w5*

*w6*

*w7*

*w0*

shifted data

shifted data

*xn*

*xn-7*

*xn-6*

*xn-1*

*xn-5*

*xn-2*

*xn-3*

*xn-4*

**p**

*w1*

*w0*

*w2*

*w3*

*w4*

*w7*

*w5*

*w6*

Next cycle

Figure 32 Wrapped linear delay-line buffer.

*xn*

*xn-7*

*xn-6*

*xn-1*

*xn-5*

*xn-2*

*xn-3*

*xn-4*

p

*w1*

*w0*

*w2*

*w3*

*w4*

*w7*

*w5*

*w6*

*xn*

*xn+1*

*xn-6*

xn-1

*xn-5*

xn-2

*xn-3*

*xn-4*

**p**

*w0*

*w1*

w2

*w3*

*w4*

*w5*

*w6*

*w7*

Shifted pointer

Next cycle

Figure 33 Circular delay-line buffer.

Because the filter is making the averaging of the data, and the data is all the ways being updated during the operation, so the filter is also called a moving average filter. The offset and gain values are sent by microcontroller, and some of the controlling parameters, such as the order of the filter, are also sent by the microprocessor in the control unit. All the internal states (the data in wi in Figure 32 and Figure 33), are saved in bank memory in the control unit board. All the necessary timings and operating instructions are saved in FPGA chip, which will be introduced in the last chpater.

The algorithms of the low-pass filtering of the offset, pixel and reference values are similar, so offset filtering is taken as an example here.

The measurements are done with a 256 detector module and the data is processed with 16 bit AD converter. The data is processed by MathCAD software.

The noise of the offset value before filtering is shown in Figure 34(a). The average RMS noise value is 811 ADC counts. When a 4-order low-pass FIR filter is applied, the noise of the offset value is measured at 407 ADC counts, as shown in Figure 34(b). when an 8-order moving average filter is used, So the filtered noise is reduced to 311 ADC counts, as shown in Figure 34(c). The theoretical value and the measured results are normalized and listed in Table 3. It can be seen that the result agrees well with what the theory predicts.

Table 3 Comparison between the theoretical values and the measured results

|  |  |  |
| --- | --- | --- |
| Filter order N | Theoretical RMS noise | Measured noise results |
| N=0 (No filtering) | 1 | 1 |
| N=4 | 0.500 | 0.502 |
| N=8 | 0.354 | 0.383 |

The noise of the pixel and reference value can also be filtered to the same extent. So the increased white noise by the correlated double sampling process is greatly reduced by the digital noise filter.

It can be seen that with an 8-order moving average filter, the noise can be improved at least 8 dB. So the system’s dynamic range reaches about 73 dB, which is

12 bits

.

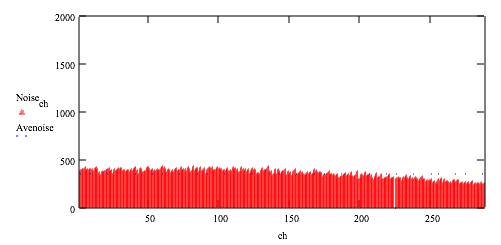


**ADC counts**

**Channel number**

**(a)**

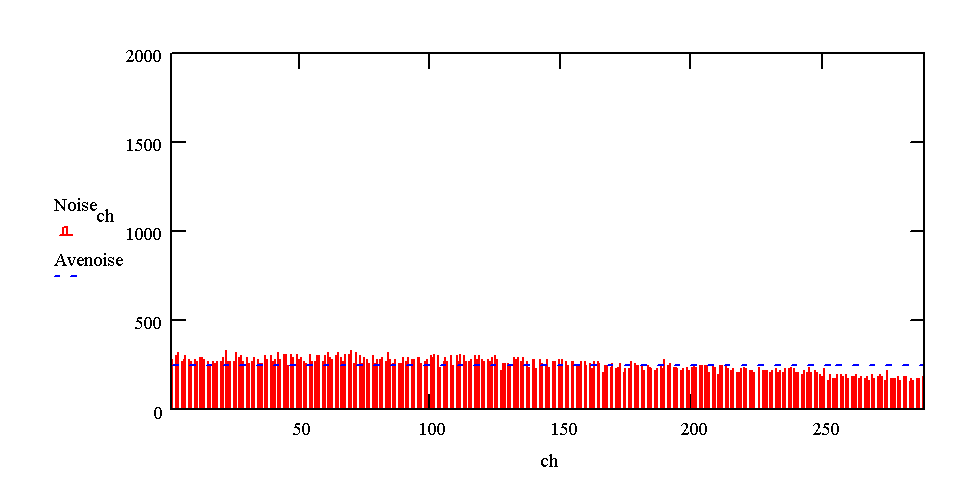
**ADC counts**



**Channel number**

**(b)**

**ADC counts**



**Channel number**

**(c)**

Figure 34 The noise of the offset values (a) before filtering (b) A four order moving average filter is used; (c) An eight order moving average filter is used.

## 4.3 Output Buffer Control

As we mentioned before, in our X-ray line-scan camera, there are three interface options can be chosen by user. They are the USB2.0, the Ethernet and the frame grabber card interfaces. The control would be much easier when the frame grabber card interface is chosen, because the buffer control and trigger control are done in the frame grabber card. In this case, the frame grabber is just a RAM-like device; the frame grabber controller block only needs to be designed according to the timing requirement of frame grabber card. However a big disadvantage of frame grabber card is its price.

In this sense, the USB2.0 and the Ethernet interfaces are better alternatives than frame grabber card for saving cost, but the output buffer control needs to be done in FPGA for these two interfaces.

X-ray line-scan camera is a real-time system, and the transfer reliability is required demandingly. As we know, the transfer speed and reliability are not only up to the performance of X-ray line-scan camera, and also depends on the performance of the host PC. The X-ray line-scan camera must have the ability to control the transfer process, even when the computer can not receive the data temporarily, for example, the computer is busy temporary when running CPU consuming software. In this case, a large size buffer that can temporarily save at least several lines data is necessary for output buffer control.

In our system, the Altera Cyclone FPGA (EP1C12F324C7) is used as the pixel DSP processor, and as we introduced in chapter three, there are total 52 M4K blocks are embedded in the FPGA. Using these M4K blocks, there are two dual-port buffers are built in the processor design, they are raw data buffer and 1st output buffer.

Because of limitation of the M4K blocks number, there are only one or two lines dual-port RAM can be used as 1st output buffer, so this requirement leads us to find out other on-board memories used to be the part of output buffer too. The external synchronous SRAM is used as bank memory in the data processing, and it has at least five banks that can store five lines of data, so it can be used for output buffer.

Finally, the architecture of output buffer is designed based on 2-level buffers, on one hand, the first level buffer uses the embedded dual-port M4K RAM that is easy to be controlled, on the other hand, the second level buffer implements in the bank memory that could save enough large size data.

The block diagram of output buffer is presented in Figure 35.

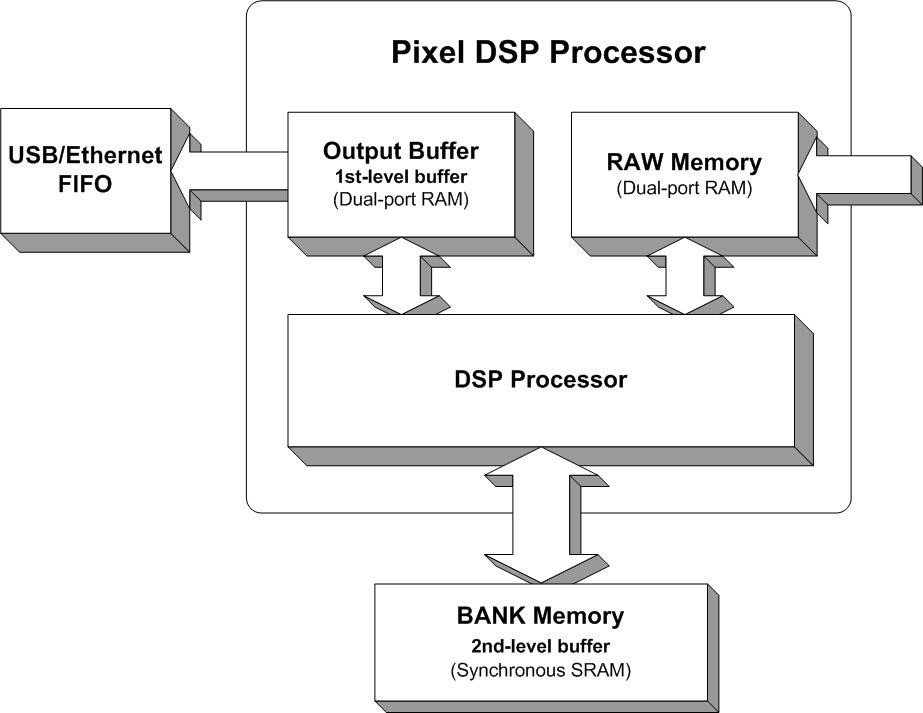
****

Figure 35 Block diagram of buffer control

In the block diagram of buffer control, we could see both the internal output buffer (1st level buffer) and external bank memory (2nd level buffer) are connected to DSP processor. Actually, the sub-block output controller in FPGA is used to control the process. The block diagram and the functional description can be referred to the chapter 3.3.1.4.

The control process for output buffers can be divided into two processes: they are the output buffer (1st level buffer) control process and the bank memory (2nd level buffer) control process.

### 4.3.1 The Output Buffer (1st level buffer) Control Process

Before the X-ray line-scan camera starts to scan the image, the output buffer should be empty, so the write address (WR\_Addr\_Tag) and read address (WR\_Addr\_Tag) of the output buffer are in initial state.

After the system starts to scan the images, the continuous processed data feeds into the output controller block from the DSP processor block, then the output controller would generate control timing signals to save the incoming processed data to the output buffer, the write address (WR\_Addr\_Tag) would increase with the incoming processed data. The write speed must match the speed of incoming data. The write process is responsible for generating the write timing signals to save the data into the buffer. Simultaneously, the output controller would also generate the read timing signals to read the saved data in the output buffer and send to the FIFO of USB device or buffer of Ethernet device when the buffer is not full. Under the control of read process, the read address (RD\_Addr\_Tag) can move along with the write address (WR\_Addr\_Tag). The read process is not only responsible for generating the read timing signals to load the data from output buffer, and also responsible for generating the timing signals to feed the data into the FIFO of USB or the buffer of Ethernet chip.

In a word, the read address (RD\_Addr\_Tag) is controlled to keep up with the write address (WR\_Addr\_Tag) of output buffer by the output controller.

In our system, the length of the output buffer is equal to one or two lines data depending on the pixel number (active length) of the camera.

The state of 1st level buffer is presented in Figure 36. The write and read flags in initial state and middle state are shown in the figure.

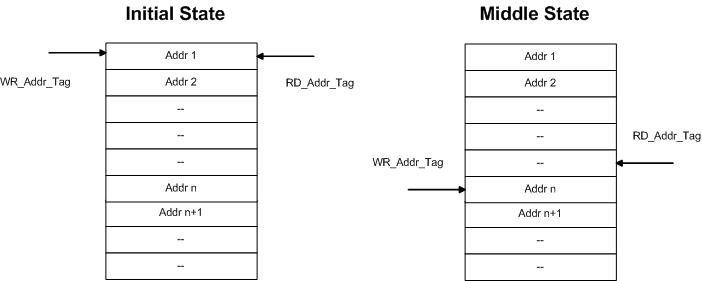


Figure 36 The output buffer (1st-level buffer) state diagram

The write process and the read process are shown in Figure 37. In the write process, we could see that the write process could report the full signal of 1st-level buffer to the DSP processor block when the read address could not catch up with write address, because without control the old data will be overwritten before it is read out. In this case, the data will then save into the bank memory (2nd-level buffer), in 2nd-level buffer, at least five lines data could be saved temporarily. The DSP processor also could adjust the output speed of the processed data, because it needs to be loaded from the bank memory to output buffer (1st-level buffer) when the old data has been sent to output devices. The incoming processed data is 16-bit, the write process of 1st-level buffer could save the data according to the requirement of output devices, for example, the bus of Ethernet chip is 8-bit, so the data width of the output buffer is 8-bit, and 16-bit processed data needs to be saved into the output buffer byte by byte.

In the read process, the read timing is different according to USB or Ethernet devices.

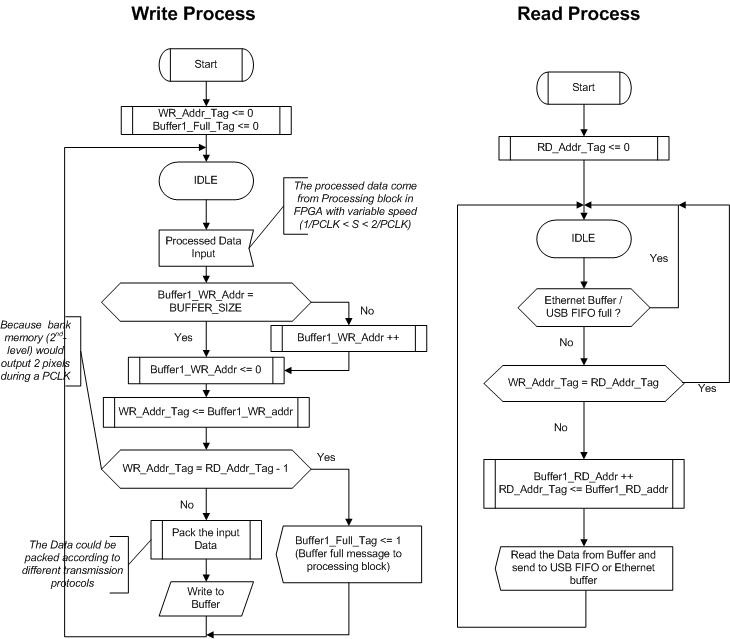


Figure 37 The control process of the output buffer

### 4.3.2 The Bank Memory (2nd-level buffer) Control Process

As we described in previous sections, the bank memory is used to save the histogram of correction value, test pattern data, and processed data, so the bank memory is used in almost every single clock cycle for DSP processing purpose. The 2nd-level buffer of output controller can only be used when it is not used for the DSP processing. So the basic timing of DSP processing using bank memory needs to be introduced before we describe how to control the 2nd-level buffer.

The simplified timing of DSP processing using bank memory is presented in Figure 38. The data is processed in the DSP processor block pixel by pixel, and we could see that one processing cycle is 100ns, the processing clock is so called PCLK that is 10MHz. One PCLK consists of ten SYS\_CLKs that is 100MHz system clock. In every SYS\_CLK cycle, the DSP processor block can perform one time read or write from or to the bank memory. As it is shown in Figure 38, the processing cycle consists of reading raw data, reading correction values, writing back processed data and reading the saved processed data etc. There are two SYS\_CLKs used to load processed data from banks. The second read clock cycle does not need to be used when the 1st buffer is enough large for the data buffering. The second clock cycle has to be used when the 1st buffer is full, because using two clock cycles, the read speed could reach up to be double of write speed, when the buffer in output devices has the enough empty room, it makes the read process is possible to catch up with write process.

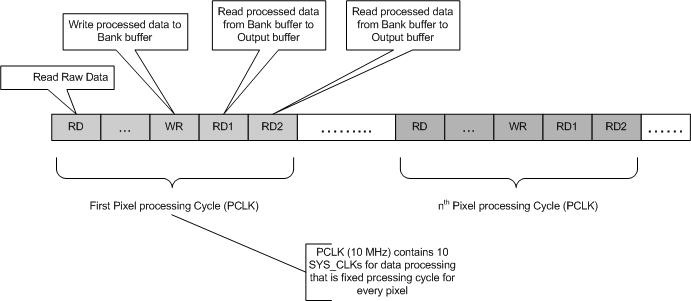


Figure 38 Simplified timing of DSP processing using bank memory

The state of 2nd-level buffer is presented in Figure 39. There are total five lines of data can be saved in the bank memory, each line of data can be saved to one bank, so totally, five banks could be used for 2nd level output buffer.

In the example of Figure 39, bank5, 6, 7, 8, and 9 are used for 2nd level buffer. These five banks compose of a two dimensions buffer, so the WR\_BankNum\_Tag and RD\_BankNum\_Tag are the flags to indicate which bank is being written or read. WR\_Addr\_Tag and RD\_Addr\_Tag are used to show that which pixel is being written or read in the bank. After WR\_BankNum\_Tag and WR\_Addr\_Tag are known, the position of current writing data could be located, and so is the read process.

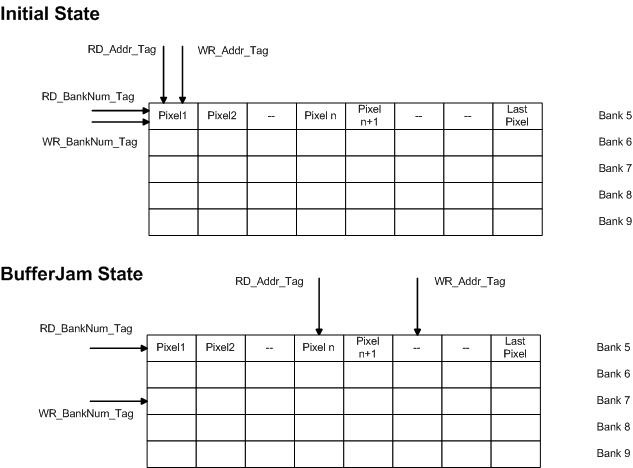


Figure 39 The state of 2nd level buffer

The write process is prsented in Figure 40, the write process could response the buffer full signal of 1st level buffer, and then start save the pixel data to the banks of 2nd level buffer. The write process also could manage how to move the two-dimension pointers (WR\_BankNum\_Tag and WR\_Addr\_Tag). When these five banks are full, the 2nd-level level buffer full signal will be triggered, and an error message would send to microcontroller to report the whole buffer full of the pixel DSP processor.

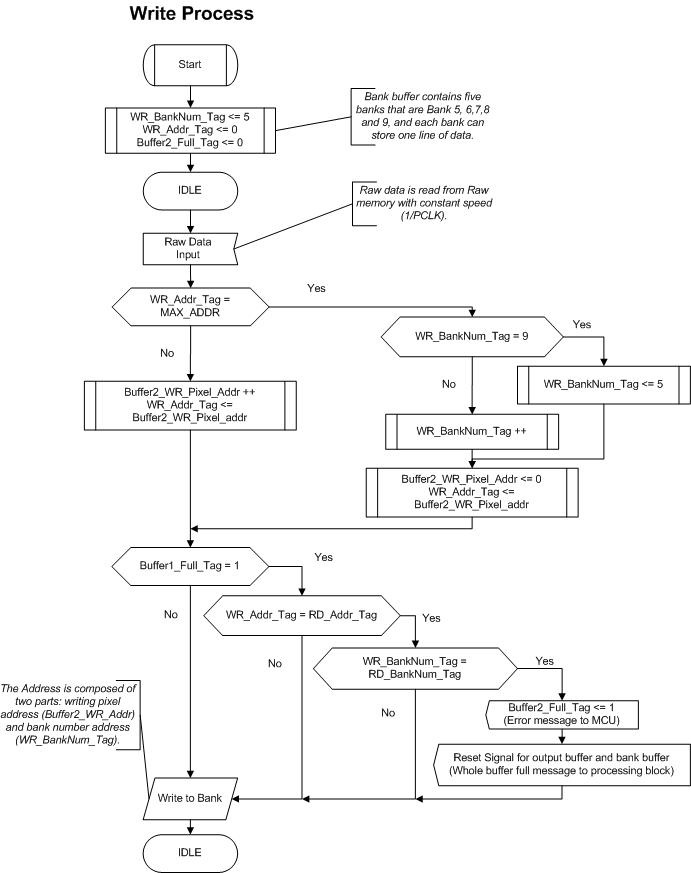
****

Figure 40 The write control of the bank buffer

The read process is presented in Figure 41. The read process could control to read the saved data in the banks of 2nd-level buffer, the second read clock cycle could be used to make read process faster than the write process, it makes the read process possible to catch up with the write process.

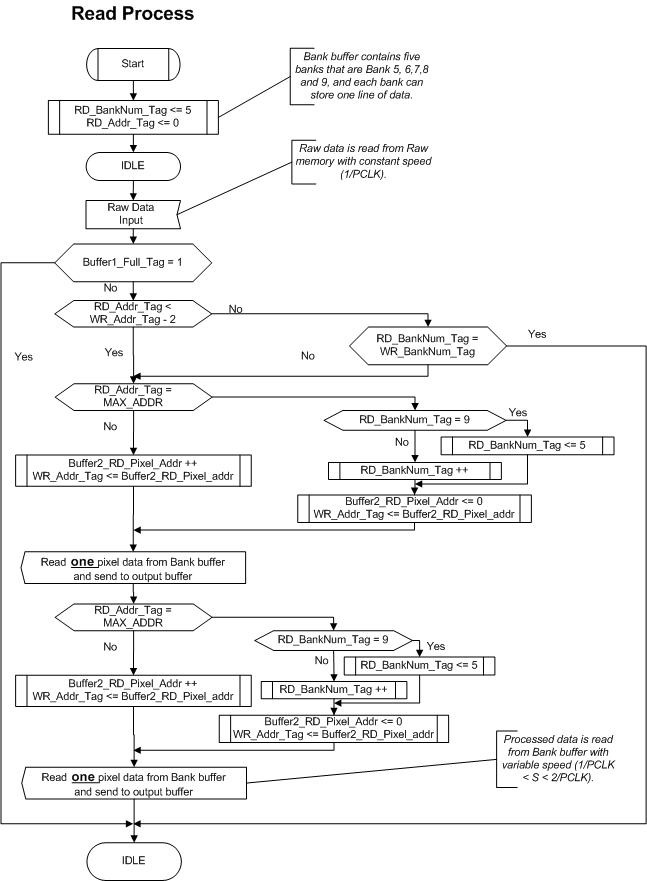


Figure 41 The read control of bank buffer

The comparison of with different lines of data of buffer is shown in Table 4. The throughput (data rate) of the X-ray line-scan camera is set to 22M bits/second that is a typical case for most of customers.

The testing computer configuration:

PC type: Toshiba Laptop

Processor type: Intel Pentium M

Processor frequency: 1.8 GHz

RAM Memory Size: 512 MB

Table 4 Comparison of data lost for different buffer size

|  |  |
| --- | --- |
| **Buffer Size (The data of line number)** | **Average line lost per minute** |
| 1 | 138.7 |
| 2 | 36.3 |
| 7 | 0 |

When the buffer size is seven lines of data, the 1st level and 2nd level buffers are used together. As we see in Table 4, The more lines buffer are used, the more reliable the system could be, so the buffer control could make the X-ray line-scan camera more reliable with the Ethernet and the USB interfaces.

# 5. MEASURED RESULTS AND DISCUSSIONS

## 5.1. Overview

The measured results including digital corrections for fixed pattern noise and digital FIR filter have been presented in last chapter.

In this chapter, the sample pictures with X-ray source are made and studied.

## 5.2. Main Applications and Sample Pictures

### 5.2.1. Main Specifications

The main specifications that most of the users care are: spatial resolution, operating speed and dynamic range.

#### 5.2.1.1. Spatial Resolution

Spatial Resolution is defined as a measure of the smallest angular or linear separation between two objects usually expressed in radians or meters.

The X-ray line-scan camera is measured in this experiment using photodiode array with 0.4 mm  0.4 mm pixel size. The spatial resolution is measured with the line pairs. Metal line pairs with different width and spacing (called phantom) are measured with the X-ray line-scan camera. The finest line pair that can be discerned by the X-ray line-scan camera is defined as the spatial resolution.

The X-ray image of the phantom is shown in Figure 42. It can be seen that line-pairs of 400 m width and spacing can be discerned.

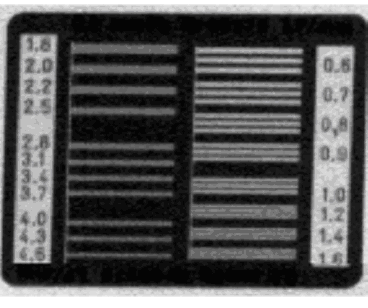
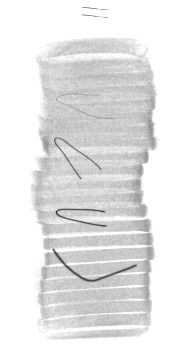


Figure 42 400m line pairs can be discerned.

Figure 43 shows a package of bread with metal wires inside. The width of the finest wire is 250 m and could be clearly detected by the X-ray line-scan camera studied in this thesis.



###### Metal wire of 1.0 mm width

###### Metal wire of 750 m width

###### Metal wire of 500 m width

###### Metal wire of 250 m width

Figure 43 Bread package with metal wires on the bottom.

#### 5.2.1.2. Speed

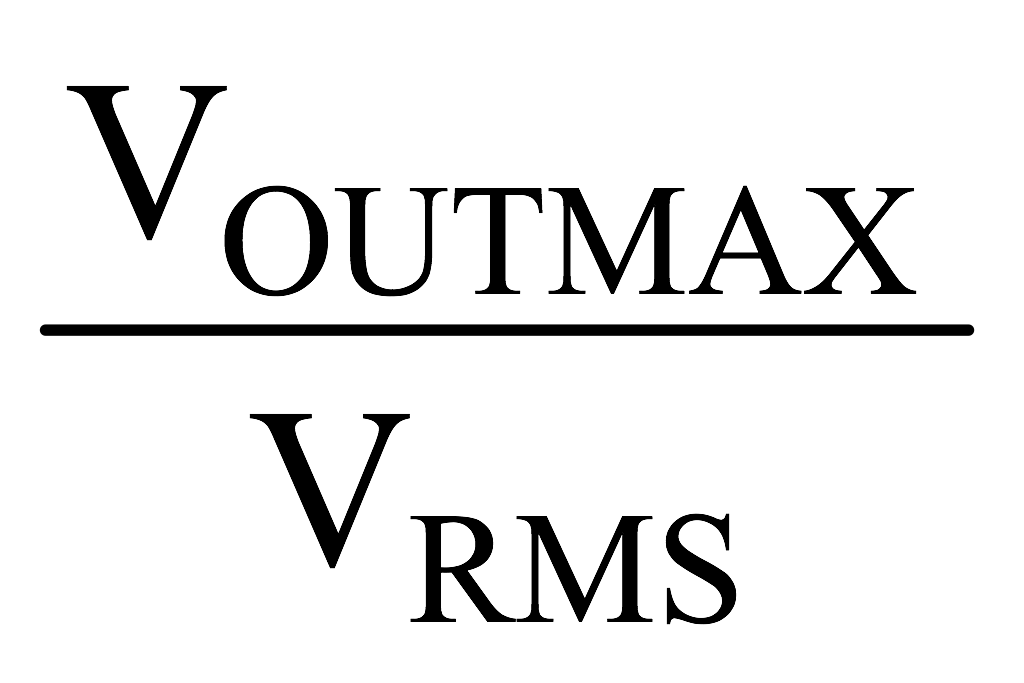
The maximum operating speed of the X-ray line-scan camera could reach 2.00 m/s without any geometric distortion. It is the fastest speed for the similar products in industry.

#### 5.2.1.3. Dynamic Range

The dynamics is that range in which the detector is capable of an accurately measurement of the signal and is defined as the maximum detectable signal divided by the minimum signal level. The maximum detectable signal is limited by the detector saturation, the minimum signal by the noise of the system.

The dynamics depends on the integration time due to the thermal generation of noise.

The dynamic range is define as follows

DR =  (6-1)

The dynamic range of detector is about 65 dB. In chapter 4, the measured results show that the pixel DSP processor can improve 8 dB dynamic range. So the dynamic range of this system could achieve 73 dB, which is 12 bits.

### 5.2.2. Main Applications and Sample Images

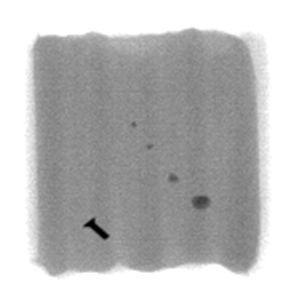
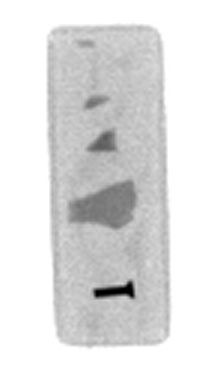
Food quality inspection is one of the main applications of the X-ray line-scan camera. In Figure 44, various food quality inspections are performed. In Figure 44(a) and (b), screws and glass pieces are mixed into the food, and can be detected easily. Figure 44(c) is a juice box with metal wire inside. In some applications, not only foreign materials need to be detected, but also the unwanted part of the food itself. For example, in some food processing factories, bones should be completely removed from meat. The residual bones should be detected. Figure 44(d) shows such an application.

Another typical application is the custom security inspection. The weapons hidden in suitcase by terrorists could easily be detected by the X-ray line-scan camera. Figure 45 shows a knife hidden in suitcase. The knife can be distinctively separated with other materials inside the suitcase.

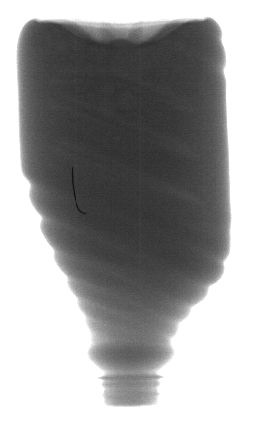
In other industries’ quality inspection, the X-ray line-scan camera can also be used widely. Figure 46(a) shows an alien metal ball inside an air-fresher box. Figure 46(b) shows a packaged bulb. The fine structures of the translucent bulb can be revealed clearly under the X-ray camera.

Figure 47 shows some applications in electronic assembly industry. Figure 47(a) shows a multi-meter. The components, such as integrated circuits, battery, metal wires, connectors, and various components, can be seen very clearly. Figure 47(b) shows a normal telephone. The components, button contacts, and other fine structures, even the solder points, can be shown clearly in the image.

A further improvement can be made by applying pseudo color to the image. The foreign material is usually quite different than the scanned objects. So a threshold voltage can be set for the gray scale. When the gray scale is larger than certain value, then the computer will show a certain color to the corresponding pixels. Figure 48 shows the pseudo color processing method. The images in Figure 48 are the same images as Figure 48(a) and (b). When the pseudo color is set, the foreign material could be found much more easily. When different pseudo colors are set, then foreign materials with different densities and thickness can be separated. Thus, detection efficiency will be enhanced.



1. (b)



(c) (d)

Figure 44 Sample pictures on industrial food quality inspection: (a) Chocolate bar with alien materials (b) Sausages package with alien material (c) Juice box with metal wire (d) Chicken with bones.

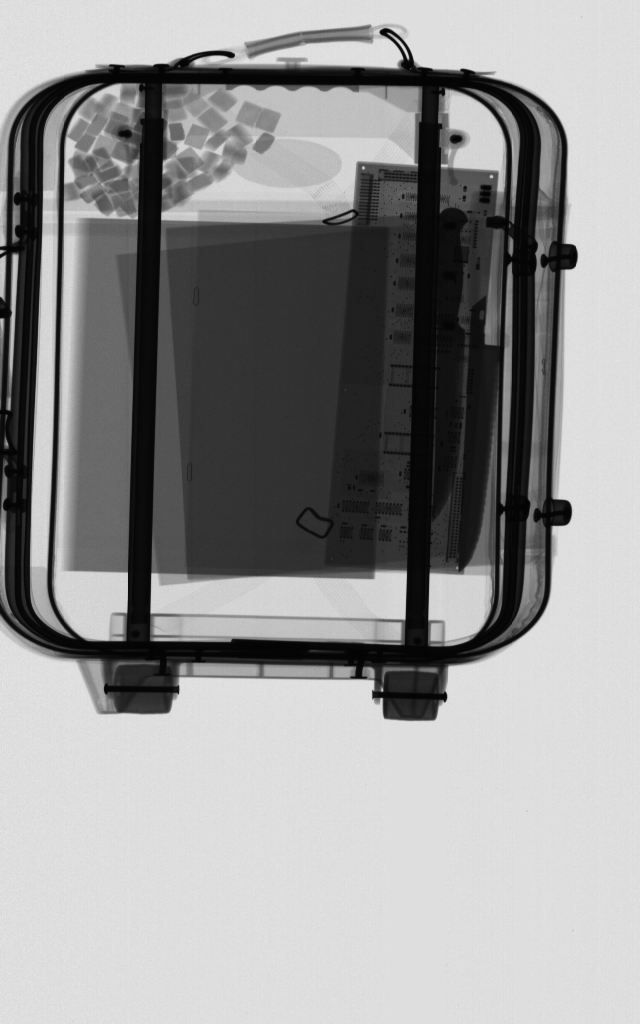
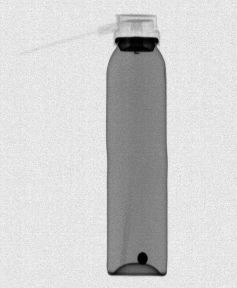
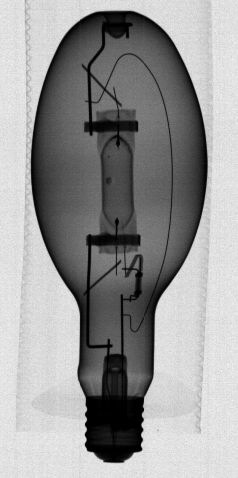
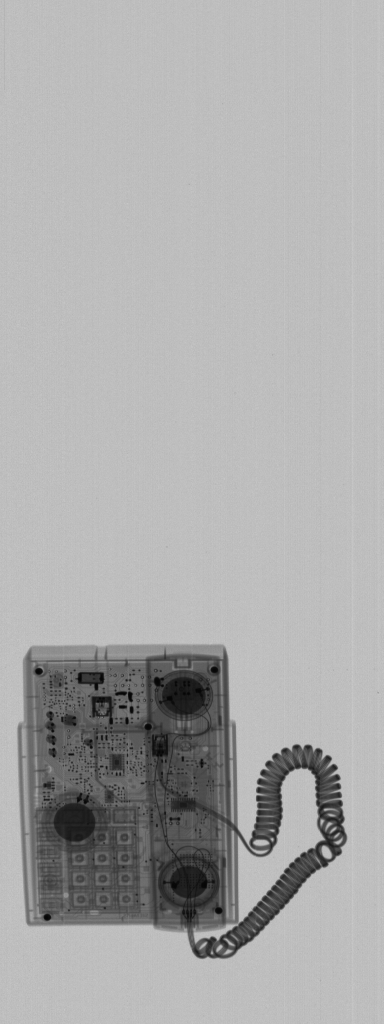


Figure 45 Custom security inspection.

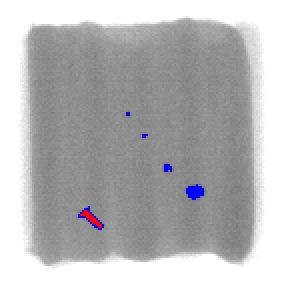
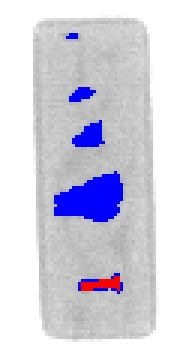
1. (b)

Figure 46 Industrial inspection (a) Air fresher bottle with a metal ball inside (b) a bulb inside paper package.

1. (b)

Figure 47 Electronic assembly industry quality inspection (a) A multi-meter (b) A telephone.

****

1. (b)

Figure 48 Pseudo-colored picture.

# 6. Conclusions

The X-ray line-scan camera studied in this thesis is a complicated smart sensor system. The whole system is introduced in this thesis, and the design of control unit board especially the pixel DSP processor in the board is presented. To meet the target requirements of low-cost, high-flexibility, high-performance and excellent-reliability, the architecture of control unit board is discussed and described in detail.

A traditional method attempts to improve the property of the sensor by exploring the physical property of the devices, which would be more and more difficult when it approaches the physical limitations. The Digital signal processing techniques are implemented to improve the performance of the sensor system. The measured results show that these various DSP algorithms can help to improve the performance of the X-ray line-scan camera.

A digital signal processor that is called pixel DSP processor in this study is used to reduce the signal noise. The fixed pattern noise caused by offset and gain variations of the detectors is totally eliminated by the digital corrections. The white noise is also reduced by the moving average filter presented in this thesis. The dynamic range of the system reaches 12bit, which meets the design goal. Parallel and pipeline processing algorithm are used to improve the system speed.

In the digital design, the output buffer control is studied in the thesis for USB2.0 and Ethernet interfaces, with the efficient buffer control, the system could become more reliable.

To make the system more flexible and cheaper, the Ethernet and USB2.0 interfaces are implemented in the X-ray line-scan camera. It makes the camera possible to work without the expensive commercial frame trigger card.

Measured results show that the operating speed of the system can reach 2 m/s, which is among the fastest level for similar products reported in industry. The spatial resolution can reach 400 m.

# References

1. [] Koskinen J., “LDA technology today and possibilities in the future,” in *Proceedings of the ASNT Digital Imaging IV conference*, August 2001. [↑](#endnote-ref-2)
2. [] Holland S., “Fabrication of detectors and transistors on high-resistivity silicon,” *Nuclear Instruments and Methods in Physics Research*, A275: 537-541, 1989 [↑](#endnote-ref-3)
3. [] Kemmer J., “Improvement of detector fabrication by the planar process,” *Nuclear Instruments and Methods in Physics Research*, A226: 89-93, 1984. [↑](#endnote-ref-4)
4. [] Betta G., Pignatel G. U., Verzellesi G., and Boscardin, M., “Si-PIN X-Ray detector technology,” *Nuclear Instruments and Methods in Physics Research*, A395: 344-348, 1997. [↑](#endnote-ref-5)
5. [] Knoll G., *Radiation detection and measurement*. John Wiley & Sons Inc., New York: 1989. [↑](#endnote-ref-6)
6. [] Hoe D. and Ribner D. “An auto-ranging photodiode preamplifier with 114dB dynamic range,” *IEEE Journal of Solid-State Circuits*, vol.31, pp. 187-194, February 1996. [↑](#endnote-ref-7)
7. [] Heijne E., “A 1006 element hybrid silicon pixel detector with strobed binary output,” *European organization for nuclear research* CERN/ECP91-26, 13th, pp. 1-10, December 1991. [↑](#endnote-ref-8)
8. [] Matsumoto H. and Watanabe K., “Spike-free switched-capacitor circuits,” *Electronics letters*, vol. 23, pp. 428-429, April, 1987. [↑](#endnote-ref-9)
9. [] Thelen D. and Chu D. “A low noise readout detector circuit for nanoampere sensor applications,” *IEEE Journal of Solid-State Circuits*, vol. 32, pp. 337-348, March 1997. [↑](#endnote-ref-10)
10. [] Ott H. *Noise reduction techniques in electronic systems*. 2nd ed. AT&T Bell laboratories, New York: 1989. [↑](#endnote-ref-12)
11. [] Detection Technology Ltd., *X-Scan linear-array detectors: X-Scan f2 series*, User's Manual, 2003. [↑](#endnote-ref-13)
12. [] Brown S. and Rose J., “FPGA and CPLD Architectures: A Tutorial”, *IEEE Design and Test of Computers*, vol. 13, pp. 42-57, 1996. [↑](#endnote-ref-15)
13. [] Berkeley Design Technology, Inc., *Choosing a DSP processor*, Technical Report, 2000. [↑](#endnote-ref-16)
14. [] Integrate Circuit Engineering Corporation, “Introduction To ASIC Outlook,” 1998 [↑](#endnote-ref-17)
15. [] Chris Dick, “FPGAs The High-end Alternative For DSP Applications,” Spring 2000 [↑](#endnote-ref-18)
16. [] Altera Corporation, *FPGA co-processing solutions for high-performance signal processing applications*, White Paper, 2005. [↑](#endnote-ref-19)
17. [] http://www.andraka.com/dsp.htm [↑](#endnote-ref-20)
18. [] Jack Shandle, “DSP Meets FPGA: Is Massive Parallelism Enough? Custom-wired algorithms make FPGAs a serious signal-processing contender” Techinical Report, 1999 [↑](#endnote-ref-21)
19. [] Cypress Semiconductor Corporation, *CY7C68013 EZ-USB FX2™ USB microcontroller high-speed USB peripheral controller*, Rev. \*C Revised, Datasheet, December 19, 2002. [↑](#endnote-ref-22)
20. [] WIZnet, *i2Chip W3100A*, v1.3, Technical Datasheet. [↑](#endnote-ref-23)
21. [] Realtek Semiconductor Corporation, *Realtek single chip single port 10/100M fast Ethernet phyceiver RTL8201BL*, Rev.1.2, Datasheet, March, 2002. [↑](#endnote-ref-24)
22. [] Integrated Device Technology, Inc., *256K x 36, 512K x 18 3.3V Synchronous ZBT™ SRAMs ZBT™ feature 3.3V I/O, burst counter pipelined outputs*, Datasheet, December 2002. [↑](#endnote-ref-25)
23. [] Hewlett-Packard Company, Hewlett Packard Components Group Imaging Products Operations, *Noise sources in CMOS image sensors*, Technical Report, April 1998. [↑](#endnote-ref-26)
24. [] Choubey B. and Collins S., “Fixed Pattern Noise in Pixels with Combined Linear and Logarithmic Response,” in *Proceedings of International Congress of Imaging Science*, 2006, in press.

    # Appendix1 MCU Control Register File

    Reg No. Register Name MCU Address Allowed Operation Description

    0 MODE #CFFF R/W FPGA mode settings

    1 FE\_CONFIG\_MODE #CFFE R/W The command type and board address of FE control command.

    2 FE\_CONFIG\_DATA\_MSB #CFFD R/W The MSB of FE command content

    3 FE\_CONFIG\_DATA\_LSB #CFFC R/W The LSB of FE command content

    4 IT\_H #CFFB R/W The highest 8 bits of integration time

    5 IT\_M #CFFA R/W The middle 8 bits of integration time

    6 IT\_L #CFF9 R/W The lowest 8 bits of integration time

    7 SCALE\_BITS #CFF8 R/W The bit number of output scaling

    8 OUTPUT\_BANK #CFF7 R/W The output bank number

    9 SUM\_LINE\_NUM #CFF6 R/W Summing line number in calibration and summing function.

    10 AVER\_LINE\_NUM #CFF5 R/W Averaging line number in averaging function.

    11 CORR #CFF4 R/W Enable/Disable Baseline, Offset, Gain, Reference and Dead channel correction.

    12 CONFIG #CFF3 R/W Configuration latch (page selection)

    13 DEAD\_PIXEL\_MODE #CFF2 R/W Dead pixel correction mode

    14 DEAD\_PIXEL\_1\_MSB #CFF1 R/W The MSB of first dead channel

    15 DEAD\_PIXEL\_1\_LSB #CFF0 R/W The LSB of first dead channel

    16 DEAD\_PIXEL\_2\_MSB #CFEF R/W The MSB of second dead channel

    17 DEAD\_PIXEL\_2\_LSB #CFEE R/W The LSB of second dead channel

    18 DEAD\_PIXEL\_3\_MSB #CFED R/W The MSB of third dead channel

    19 DEAD\_PIXEL\_3\_LSB #CFEC R/W The LSB of third dead channel

    20 BASELINE\_MSB #CFEB R/W The MSB of BASELINE value

    21 BASELINE\_LSB #CFEA R/W The LSB of BASELINE value

    22 PIXEL\_TAP #CFE9 R/W Number pixel values used in moving averaging pixel filter

    23 OFFSET\_TAP #CFE8 R/W Number offset values used in moving averaging offset filter

    24 SH\_DELAY #CFE7 R/W Trigger delay time (S/H delay)

    25 DM\_GAIN #CFE6 R/W Programmable gain setting of detector modules.

    26 ENET\_A\_ST\_MSB #CFE5 R/W MSB of start address to access the FIFO of Wiznet TCP/IP chip

    27 ENET\_A\_ST\_LSB #CFE4 R/W LSB of start address to access the FIFO of Wiznet TCP/IP chip

    28 ENET\_A\_LENGTH\_MSB #CFE3 R/W MSB of address length to access the FIFO of Wiznet TCP/IP chip

    29 ENET\_A\_LENGTH\_LSB #CFE2 R/W MSB of address length to access the FIFO of Wiznet TCP/IP chip

    Reg No. Register Name MCU Address Allowed Operation Description

    30 STATUS #CFE1 R Current FPGA status

    31 FRAME\_LINE\_NUM #CFE0 R/W The line number for frame trigger

    32 LINE\_PX\_NUM\_MSB #CFDF R/W The MSB of pixel number of one line

    33 LINE\_PX\_NUM \_LSB #CFDE R/W The LSB of pixel number of one line

    34 TEST\_MODE #CFDD R/W Test mode settings

    35 OUTPUT\_GAIN\_MSB #CFDC R/W The MSB of output programmable gain

    36 OUTPUT\_GAIN\_LSB #CFDB R/W The LSB of output programmable gain

    37 ENET\_IP\_FIRST\_BYTE #CFDA R/W The first byte of Ethernet IP address

    38 ENET\_IP\_SECOND\_BYTE #CFD9 R/W The second byte of Ethernet IP address

    39 ENET\_IP\_THIRD\_BYTE #CFD8 R/W The third byte of Ethernet IP address

    40 ENET\_IP\_FOURTH\_BYTE #CFD7 R/W The fourth byte of Ethernet IP address

    41 ENET\_SM\_FIRST\_BYTE #CFD6 R/W The first byte of Ethernet subnet mask

    42 ENET\_SM\_SECOND\_BYTE #CFD5 R/W The second byte of Ethernet subnet mask

    43 ENET\_SM\_THIRD\_BYTE #CFD4 R/W The third byte of Ethernet subnet mask

    44 ENET\_SM\_FOUTH\_BYTE #CFD3 R/W The fourth byte of Ethernet subnet mask

    45 ENET\_MAC\_ FIRST\_BYTE #CFD2 R/W The first byte of Ethernet MAC address

    46 ENET\_MAC\_SECOND\_BYTE #CFD1 R/W The second byte of Ethernet MAC address

    47 ENET\_MAC\_THIRD\_BYTE #CFD0 R/W The third byte of Ethernet MAC address

    48 ENET\_MAC\_FOURTH\_BYTE #CFCF R/W The fourth byte of Ethernet MAC address

    49 ENET\_MAC\_FIFTH\_BYTE #CFCE R/W The fifth byte of Ethernet MAC address

    50 ENET\_MAC\_SIXTH\_BYTE #CFCD R/W The sixth byte of Ethernet MAC address

    51 ENET\_GA\_FIRST\_BYTE #CFCC R/W The first byte of Ethernet gateway address

    52 ENET\_GA\_SECOND\_BYTE #CFCB R/W The second byte of Ethernet gateway address

    53 ENET\_GA\_THIRD\_BYTE #CFCA R/W The third byte of Ethernet gateway address

    54 ENET\_GA\_FOURTH\_BYTE #CFC9 R/W The fourth byte of Ethernet gateway address

    55 FRAME\_TRIG\_DELAY #CFC8 R/W The delay line number of frame trigger

    56 FRAME\_TRIG\_MODE #CFC7 R/W Set different trigger mode for external frame trigger

    57 Reserved #CFC6 NULL For future usage

    58 Reserved #CFC5 NULL For future usage

    59 Reserved #CFC4 NULL For future usage

    60 Reserved #CFC3 NULL For future usage

    61 DT\_TEST #CFC2 R/W For internal test purpose

    62 FPGA\_REVISION #CFC1 R The current revision of FPGA design

    63 FPGA\_BUILD #CFC0 R The current build number of FPGA design [↑](#endnote-ref-27)