Reducing Complexity Increasing Scalability on X-Ray stress system

Hussein Wehbe
University of Windsor

Follow this and additional works at: https://scholar.uwindsor.ca/etd

Recommended Citation
https://scholar.uwindsor.ca/etd/5956

This online database contains the full-text of PhD dissertations and Masters' theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email (scholarship@uwindsor.ca) or by telephone at 519-253-3000ext. 3208.
Reducing Complexity, Increasing Scalability on X-Ray Stress System
by
Hussein Wehbe

APPROVED BY:

S. Rehse
Department of Physics

E. Abdel-Raheem
Department of Electrical and Computer Engineering

K. Tepe, Advisor
Department of Electrical and Computer Engineering

August 22, 2016
DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone’s copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.
Programmable Logic Controllers (PLCs), traditionally used for process control and automation, have certain limitations. Firstly, they operate by running the ladder logic in a scan cycle that has three steps: inputs canning, program execution, outputs updating. This sequential execution affects the response time of the PLC for critical events, as they will keep waiting until they are read in an inputs scanning step. Secondly, PLCs are expensive, available only in standard configurations and need to be mounted on special panels. Finally, using PLC adds to wiring complexity of the system and reduces scalability.

As a solution to these limitations, this thesis proposes using an MCU based embedded control system with event driven software architecture. For the implementation, an ATMEGA-2560 MCU is selected and used in X-Ray measurement system automation board. The controller is designed and implemented for Proto Manufacturing iXRD stress measurement system to replace the existing PLC-based system. All iXRD machine inputs are configured to generate interrupts at Atmega microcontroller to ensure faster response times based on the priority of the system events. The hardware and the firmware designs implemented in the thesis allow quick and easy expansion of the system by adding new inputs or outputs with a minimal wiring and smaller number of new components. The new design minimizes the system number of components, integration time and complexity, thereby reducing the overall system setup and maintenance costs.
DEDICATION TO

My Mother: A strong and gentle soul who taught me to trust in God, believe in hard work and that so much could be done with so little.

My Father: For earning an honest living for us and for supporting and encouraging me to believe in myself.

My Brother Youssef: For supporting and being always by my side throughout my entire life.

My Wife: For supporting and understanding when I had to work late nights, always by my side throughout this year.

My Daughter Salwa and Nephew Hamoudy: Who because of them, I will continue to push myself to be the best version of myself so that when they grow up they will have a role model that both can look up to.
I would like to sincerely thank my supervisor, Dr Tepe, for his guidance and support throughout this study, and especially for his confidence in me. I would also like to thank Dr. Rehse and Dr. Abdel-Raheem for serving as a member on my thesis committee. Their comments and questions were very beneficial in my completion of the manuscript. I learned from their insight a lot.
# CONTENTS

<table>
<thead>
<tr>
<th>Declaration of Originality</th>
<th>iii</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>iv</td>
</tr>
<tr>
<td>Dedication To</td>
<td>v</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>vi</td>
</tr>
<tr>
<td>Contents</td>
<td>vii</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xi</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xiii</td>
</tr>
<tr>
<td>List of Abbreviations</td>
<td>xiv</td>
</tr>
</tbody>
</table>

## 1. Introduction

1.1 Problem Statement                                | 2       |
1.2 Thesis Contribution                              | 2       |
1.3 Design and Implementation Requirements           | 3       |
1.4 Thesis Organization                              | 4       |

## 2. Related Work and Background                    | 5       |

2.1 Introduction                                     | 5       |
2.2 PLC in Process Automation .............................................................................. 5
  2.2.1 PLC Sequential Execution and Response Time ........................................... 6
  2.2.2 PLC Manufacturing Cost ........................................................................... 8
  2.2.3 System Complexity and Scalability with PLC ............................................ 8
  2.3 MCU based Embedded Controllers in Automation ........................................... 9
  2.4 Current System Description ......................................................................... 10
  2.5 RS-485 Communication Protocol .................................................................... 13
  2.6 ATMEGA ATMEGA-2560 Microcontroller ..................................................... 15
  2.7 ATMEGA-2560 Programming Interface ........................................................ 16
  2.8 Summary ..................................................................................................... 17

3. Design and Implementation ........................................................................... 18
  3.1 Introduction ................................................................................................ 18
  3.2 The Design and Implementation Methodology ............................................ 18
    3.2.1 Requirements Analysis ......................................................................... 18
    3.2.2 System Design ..................................................................................... 19
    3.2.3 XSA Board Design ............................................................................. 20
      3.2.3.1 MCU Selection ............................................................................. 21
      3.2.3.2 Sensors Interface ......................................................................... 21
        Temperature sensor interface .................................................................. 21
        Proximity Sensor (E2EC-CR5C1 2M) ...................................................... 22
Photomicrosensor (EE SX1018) .......................................................... 23
DRV101:U9 – Solenoid driver ............................................................. 24
RC low-pass filter ............................................................................ 26
Voltage Regulator ........................................................................... 26
Temperature comparator ................................................................ 27
RS-485 Transceiver .......................................................................... 28
XSA Board Schematic and PCB Layout ........................................... 29

3.2.4  SW Design and Implementation ............................................. 30
3.2.4.1  SW Architecture ................................................................. 31
Software Component Diagram ......................................................... 31
Software Behavioral Diagram ........................................................... 33

3.2.4.2  Application Layer ............................................................... 36
Main ................................................................................................. 36
Interrupt Service Routines ............................................................... 38
Timer0 compare ISR .......................................................................... 38
INT2 ISR .......................................................................................... 40
INT0 ISR .......................................................................................... 40
USART0_UDRE_vect .......................................................................... 41
USART0_RX_vect ............................................................................... 43
RS-485 Message Processing .............................................................. 45
LED/Switch handling functions ........................................................................................................... 45

3.2.4.3 Middleware Layer .................................................................................................................. 45

Timer Functions .................................................................................................................................. 46

Interrupt Setup functions .................................................................................................................... 46

Serial Communication (USART) functions .......................................................................................... 46

SPI functions ....................................................................................................................................... 46

Miscellaneous Controller Initialization Functions .................................................................................. 47

3.3 Summary ..................................................................................................................................... 47

4. RESULTS and CONCLUSION ....................................................................................................... 48

4.1 Introduction .................................................................................................................................. 48

4.2 Response Time Analysis ............................................................................................................... 48

4.3 System Complexity Analysis ......................................................................................................... 50

4.4 System Scalability Analysis ......................................................................................................... 50

4.5 Cost Savings Analysis .................................................................................................................... 51

4.6 Conclusion and Future work ......................................................................................................... 52

REFERENCES ..................................................................................................................................... 54

APPENDIX .......................................................................................................................................... 57

A. Interrupt Vectors in ATMEGA-2560 ............................................................................................ 57

VITA AUCTORIS ................................................................................................................................. 60
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sample ladder logic program</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>Scanning the ladder program</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>PLC execution cycle</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>Existing iXRD X-Ray stress measurement system</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>LEDEX Rotary Solenoid</td>
<td>11</td>
</tr>
<tr>
<td>6</td>
<td>X-Ray stress measurement system architecture with PLC</td>
<td>13</td>
</tr>
<tr>
<td>7</td>
<td>RS-485 configuration</td>
<td>14</td>
</tr>
<tr>
<td>8</td>
<td>RS-485 data frame</td>
<td>15</td>
</tr>
<tr>
<td>9</td>
<td>ATMEL-2560 pin diagram</td>
<td>16</td>
</tr>
<tr>
<td>10</td>
<td>SPI interface</td>
<td>17</td>
</tr>
<tr>
<td>11</td>
<td>X-Ray stress measurement system architecture with XSA board</td>
<td>20</td>
</tr>
<tr>
<td>12</td>
<td>XSA block diagram</td>
<td>20</td>
</tr>
<tr>
<td>13</td>
<td>Temperature sensor interface circuit</td>
<td>22</td>
</tr>
<tr>
<td>14</td>
<td>Comparator circuit</td>
<td>22</td>
</tr>
<tr>
<td>15</td>
<td>Proximity Sensor</td>
<td>23</td>
</tr>
<tr>
<td>16</td>
<td>Photomicrosensors circuit</td>
<td>24</td>
</tr>
<tr>
<td>17</td>
<td>DRV101</td>
<td>25</td>
</tr>
</tbody>
</table>
Figure 18 - RC-low pass filter ................................................................. 26
Figure 19 - Temperature comparator circuit .......................................... 27
Figure 20 - MAX3491 Transceiver .......................................................... 29
Figure 21 - Hardware schematic and the PCB layout of XSA board ............ 30
Figure 22 - Software component architecture ......................................... 33
Figure 23 - Software behavioral architecture ......................................... 34
Figure 24 - Control flow of main function ............................................. 37
Figure 25 - Control flow of Timer0 interrupt ......................................... 39
Figure 26 - Control flow - RS-485 transmit ISR .................................... 42
Figure 27 - Control flow - RS-485 receive ISR ..................................... 44
Figure 28 - Scalability Vs Complexity .................................................... 51
<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Hardware Components on PCB</td>
<td>30</td>
</tr>
<tr>
<td>Table 2</td>
<td>List of interrupts used in XSA board software</td>
<td>35</td>
</tr>
<tr>
<td>Table 3</td>
<td>Response Times of XSA board and PLC</td>
<td>48</td>
</tr>
<tr>
<td>Table 4</td>
<td>Cost Comparison</td>
<td>52</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Definition</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------</td>
<td></td>
</tr>
<tr>
<td>OEM</td>
<td>Original Equipment Manufacturer</td>
<td></td>
</tr>
<tr>
<td>PLC</td>
<td>Programmable Logic Controller</td>
<td></td>
</tr>
<tr>
<td>GM</td>
<td>General Motors</td>
<td></td>
</tr>
<tr>
<td>LL</td>
<td>Ladder Logic</td>
<td></td>
</tr>
<tr>
<td>SoC</td>
<td>System on Chip</td>
<td></td>
</tr>
<tr>
<td>MCU</td>
<td>Micro Controller Unit</td>
<td></td>
</tr>
<tr>
<td>I/O</td>
<td>Input / Output</td>
<td></td>
</tr>
<tr>
<td>PCB</td>
<td>Printed Circuit Board</td>
<td></td>
</tr>
<tr>
<td>FPGA</td>
<td>Field Programmable Grid Array</td>
<td></td>
</tr>
<tr>
<td>SFC</td>
<td>Sequential Function Chart</td>
<td></td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computer</td>
<td></td>
</tr>
<tr>
<td>ADC</td>
<td>Analog to Digital Converter</td>
<td></td>
</tr>
<tr>
<td>PWM</td>
<td>Pulse Width Modulation</td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>Transistor-Transistor Logic</td>
<td></td>
</tr>
<tr>
<td>RC</td>
<td>Resistor Capacitor</td>
<td></td>
</tr>
<tr>
<td>LED</td>
<td>Light Emitting Diode</td>
<td></td>
</tr>
<tr>
<td>ISR</td>
<td>Interrupt Service Routine</td>
<td></td>
</tr>
<tr>
<td>SCI</td>
<td>Serial Communication Interface</td>
<td></td>
</tr>
<tr>
<td>DDR</td>
<td>Data Direction Register</td>
<td></td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit</td>
<td></td>
</tr>
<tr>
<td>DAC</td>
<td>Digital to Analog Converter</td>
<td></td>
</tr>
<tr>
<td>RTOS</td>
<td>Real-Time Operating System</td>
<td></td>
</tr>
<tr>
<td>IDE</td>
<td>Integrated Development Environment</td>
<td></td>
</tr>
<tr>
<td>IXRD</td>
<td>X-Ray Powder Diffraction</td>
<td></td>
</tr>
</tbody>
</table>
1. INTRODUCTION

X-Ray, since its invention in 1895 [1], is used in many applications in medical and industrial applications, such as diagnosis of fractured bones, lung diseases, and baggage screening [2]. One of the important use cases of X-Ray diffraction technology is in the stress analysis of composite materials and the structural analysis of crystals [3]. Companies like Proto Manufacturing in Windsor; and a few other companies around the globe, design and manufacture systems with X-Ray for residual stress mapping of composite materials, powder diffraction, and structural analysis of crystals [4]. However, an exposure to X-Rays and other radioactive materials associated with those systems are hazardous to the human health. For a safe and effective operation of these systems, automation to limit human interaction is essential [5]. That is why Original Equipment Manufacturers (OEMs) have been using Programmable Logic Controllers (PLCs) for the automation of the systems, which uses X-Rays.

PLC was first developed and used by General Motors (GM) in the 1970s[6][7] for the automation systems in their factories as a replacement of expensive and complex relay panels. Since then PLCs have evolved rapidly and have been de facto standard for controlling processes and automation. PLCs are popular because they are programmable, expandable, robust and secure. Invention of PLCs has also introduced a new way of programming where Ladder Diagram (LD), Functional Block Diagram (FBD), Structured Text (ST) [8] are used to enable engineers to implement process control and automation algorithms easily without any need to know underlying microcontroller and other hardware in PLCs.

However, PLCs are not suitable for certain automation applications, where the process or the machine to be automated involves small number of inputs and outputs because their high cost, complexity in wiring and slower response time becomes a problem.
In this thesis, a Microcontroller Unit (MCU) based control and automation system for X-Ray based stress measurement system will be investigated to replace a PLC based system to reduce the cost, complexity and increase response time. With this MCU based control and automation systems, the overall complexity and cost of Proto Manufacturing iXRD stress measurement system will be reduced. Then, the MCU based and PLC based system performances will be tested to verify the design objectives.

1.1 Problem Statement

PLCs, traditionally used for process control and automation, operate by continuously scanning sensor and control inputs, and apply control signals to actuators and relays. A PLC program continuously repeat this input scanning and apply control signaling multiple times per second, which is called a scan cycle of PLC. This cycle has three steps: inputs scanning, program execution, outputs updating\[9\]. A convention is used in the PLC programming that it reads inputs from left to right and top to bottom order in LD and changes output states in the same order. Because of this flow pattern (i.e., left to right and top to bottom), inputs at the top of the rungs get higher priority than inputs on the lower rungs in a scan cycle. In successive cycles, reading and updating is periodic and is equal to the duration of the PLC scan cycle. This order is fixed and you cannot override it in the case of any critical event since an input in the lower rung needs to be serviced. That is why duration of a scan cycle of PLCs directly affects its response time even in a critical scenario\[10\].

1.2 Thesis Contribution

In practice, there are events that must be handled immediately in the automation process; sometimes inputs are received out of order, and controls must be applied accordingly in out of order. PLCs with large number of input and output ports, wiring complexities and inherent issues of ladder logic have problems to meet timing and scalability requirements. That is why an alternative control and automation platform with a common MCU is proposed, designed and implemented in the thesis. This new MCU based controller reduces
complexity of wiring and response time to an unanticipated event. An interrupt-driven MCU based automation system can respond to such unanticipated events and reduces problems associated with PLC scan cycle.

For the implementation, an ATMEGA-2560 [11] MCU is selected and used in X-Ray measurement system automation board. The controller is designed and implemented for Proto Manufacturing iXRD stress measurement system to replace the existing PLC-based system. All iXRD machine inputs are configured to generate interrupts at Atmega microcontroller to ensure faster response times based on the priority of the system events. The hardware and the firmware designs implemented in the thesis allow quick and easy expansion of the system by adding new inputs or outputs with a minimal wiring and smaller number of new components. A multi-node communication mechanism based on RS-485 is implemented between different communicating nodes of the existing system to maintain the functionality of the system intact but to have a shorter response time. The new design minimizes the system integration time and complexity, thereby reducing the overall system setup and maintenance costs.

1.3 Design and Implementation Requirements

An electronics control system board with a microcontroller, sensors and other hardware components shall be designed to automate the operation of Proto Manufacturing iXRD stress measurement system, which is currently automated using PLC. The new control system; without compromising the existing functionality, shall achieve the following objectives:

i. Improve the response time of the system by supporting priority based program execution vs. sequential execution of the PLC program.

ii. Reduce the system design cost by eliminating PLC and other non-critical elements in the system.

iii. Reduce the system integration complexity by simplifying system wiring.

iv. Support scalability of the system to add more sensors when needed.
1.4 Thesis Organization

The thesis is organized into 4 chapters. Chapter 1 (this chapter), explains the problem and contribution of the thesis. Chapter 2 discusses the related work in the field of process control and automation and explains challenges and limitations of PLC-based automation systems. The proposed MCU-based control system’s design is explained in detail in Chapter 3. The performance analysis of the new control system against the existing PLC-based control system is provided in Chapter 4. The conclusion and future work are provided in Chapter 4.
2. RELATED WORK AND BACKGROUND

2.1 Introduction

This chapter will establish the necessary background needed for understanding the design and implementation of the MCU based board proposed and implemented in this thesis for the automation of X-Ray stress measurement system. With the developments in electronics and computer technology, microprocessors and microcontrollers have taken charge of process control and automation in all aspects of human life. Micro Controller Units (MCUs) also referred to as System on Chips (SoCs), are widely used in application specific control systems. An MCU is a single integrated circuit containing a processor core, memory and input/output peripherals and is suitable for implementing control applications with predefined set of tasks. In the field of process automation, MCU-based solutions are preferred by OEMs over other automation solutions such as PLCs. MCU based boards offer better performance, flexibility, customization and cost effectiveness over PLCs.

2.2 PLC in Process Automation

As the science and technology evolved, industrial control has seen various control techniques such as is relay logic, which was introduced with factory electrification and received wide acceptability from 1900 through 1920’s [12]. In this system, limit switches and sensors were configured with banks of control relays based on given specifications to provide control functionalities. One of the disadvantages of this system was that when these specifications changed, the machines needed reconfiguration. To solve this issue, factories started using mini-computers for process automation, and reprogrammed only the
computer when specifications changed. The programs were written in high-level languages such as FORTRAN or low-level languages such as assembly, but the factory floor workers or technicians knew only ladder logic and it was not possible for them to learn new programming languages to reprogram computers. PLCs that use ladder logic programming were developed to overcome this limitation \[13\]. PLCs designed to provide ladder logic found wide acceptance in automation of large factory machines and processes. Over a period, PLCs also got into systems and machines where their usage was not really needed. For example, the X-Ray stress measurement system, referenced in this thesis does not need frequent changes in configurations and hence does not benefit from a PLC. In addition, because of their design, PLCs have an inherent limitation in the response time because of ladder logic and sequential execution discussed below. Other challenges faced by OEMs in using PLCs for automation are discussed in the following sections.

2.2.1 PLC Sequential Execution and Response Time

A typical ladder logic that PLC executes and its equivalent electrical circuit are as shown in Figure 1 \[14\]. The circuit is drawn in a horizontal line between two power lines L1 and L2, which resemble vertical sides of a ladder and the circuit itself looks like a rung in the ladder. Hence the name ladder logic coined for such systems. When a PLC is in a run mode, as shown in Figure 2, it scans the ladder logic from left to right and from top to bottom until it reaches an END rung and then resumes the execution at the top rung. This procedure of going through all the rungs of the program is called a PLC scan cycle. In each scan cycle, the PLC will firstly acquire all input values, secondly execute the program and then finally update outputs as shown in Figure 3.

![Sample ladder logic program](image)
The disadvantage in this type of execution is that the PLC can only identify input changes that occurred before the start of its input scanning stage. That is why the response time for an input that changed immediately after the PLC completed reading that input is one scan cycle. One scan cycle of PLCs varies from tens of milliseconds to hundreds of milliseconds depending on the number of inputs and outputs, the complexity of the ladder program and the execution speed of PLC.
2.2.2 **PLC Manufacturing Cost**

PLCs available in the commercial market must be bought in available standard configuration and offer no flexibility when the number of Input / Output (I/Os) needed is much less than the minimum available 32 I/Os. The iXRD X-Ray stress measurement system considered in this thesis has fewer than 10 I/Os and buying a commercial PLC for each machine is expensive. An MCU-based embedded controller can be implemented to satisfy the specific number of I/O requirement and reduce the cost of the entire system.

2.2.3 **System Complexity and Scalability with PLC**

Since PLCs have a standard configuration, connecting sensors and actuators need more wires than in a custom designed embedded controller system. This could often increase wiring complexity during system integration and increase integration time, effort and cost. With the custom design, the wiring needs can be taken care at the time of Printed Circuit Board (PCB) design to reduce the integration time due to wiring errors. When the system with PLC needs to be scaled up by adding new I/Os to the system, similar wiring complexity issues will be encountered. A custom design can take care of future expansion needs by making necessary provisions in the design.
2.3 MCU based Embedded Controllers in Automation

Though PLCs themselves are embedded controllers, they are not suitable for all type of automation needs as described in the previous sections. When a large number of machine units need to be automated and systems have less I/O requirement, using an MCU-based embedded control system is preferred. The relevance of an embedded controller in place of the traditional PLC is discussed in [15]. It states that an embedded PLC can offer improved compatibility with new equipment and higher performance in complex control algorithms while retaining the traditional ladder logic programming and execution framework. An ATxmega256A3U-AU MCU based embedded system is proposed as mini-PLC in [9]. A Field Programmable Gate Array (FPGA) based micro-PLC design is proposed in [16] and suggests that using a configurable hardware design offers more flexibility and accuracy while reducing the cost. All the solutions proposed above try to retain the traditional ladder logic programming and sequential execution property of PLCs for easier adaptability to the automation engineers who are familiar with PLCs. The reference [10] even proposes an application program to interpret logic diagrams such as ladder diagrams and Sequential Function Charts (SFCs), and run them on microcontrollers. Though these references propose new flexible designs for PLC hardware, they maintain the same execution principle as existing PLCs and loose possible improvements in the system response time that can be achieved with MCU-based solutions.

The reference [17] states that sequential execution nature of general purpose PLCs cannot meet some of the functions of construction machinery and proposes using a special purpose embedded motion controller instead. A 32-bit MCU-based embedded system with the eCos real-time operating system is proposed in [18] for automation of a batch-dying machine and shows, through experimental results, that higher performance is produced. Study of these references implies that a MCU-based control system for automation of X-Ray stress system can produce same results as desired in this thesis.
2.4 Current System Description

To design a good embedded system controller for the automation of the iXRD stress measurement system, it is necessary to understand different parts and construction of the existing system. In this section, the existing iXRD machine is carefully studied and inputs needed for the new design are discussed.

The iXRD stress measurement system has the following major subsections as shown in Figure 4.

- X-Ray head
- Goniometry system
- Detector
- XRD box
- Control application on Personal Computer (PC)

Figure 4 - Existing iXRD X-Ray stress measurement system
The X-Ray head is the main part of the iXRD machine and contains the X-Ray tube, shutter, solenoid and sensing platform. The X-Ray tube is responsible for producing X-Ray radiation, and the shutter is the socket that controls the flow of radiation from the X-Ray head to the target material. Opening and closing of the shutter is controlled by a 45-degree H-1141-030 LEDEX rotary solenoid [19] that is shown in Figure 5. Various sensors responsible for measuring temperature, detecting focus, checking the status of the shutter and X-Ray tube are located in the head. Each sensor, with the help of three wires, is connected to the PLC, which is mounted inside the XRD box. The distance between the X-Ray head and the XRD box, where PLC sits is about two meters. A lot of wiring is needed between X-Ray head and the XRD box as each sensor needs three wires to connect it to the PLC. The new system must be designed to reduce this wiring complexity, reduce cost, and increase overall reliability.

![Figure 5 - LEDEX Rotary Solenoid](image)

The goniometry system is responsible for the precise positioning of the X-Ray head in the three-dimensional (3D) space. This system is a combination of three M-Drive Plus smart motors [20] each controlling one axis in 3D space. These motors are connected to the PC on a bus network topology and are controlled directly by the application running on the PC. The X-Ray system automation board is connected to this network and it receives commands from PC and sends status updates to the status requests from PC. In the X-Ray system automation board designed and implemented in this thesis, the RS-485 communication with PC need to be designed and implemented such that communication with PC is the same as that of the existing system.
The detector on the machine is responsible for detecting the reflected X-Ray and converting it into an electrical signal. The reflected X-Ray first falls onto a phosphorous thin sheet that converts it to the visible light and then using fiber optic this light is transferred onto the surface of 512 pixels photodiode, which converts visible light to electrical signal. This electrical signal is fed into an Analog to Digital Converter (ADC) board and the digital value is sent to the PC through a dedicated USB port.

The PLC used for the automation is housed inside the XRD box and control to turn on/off the system is provided on the front panel of the XRD box. The XRD box also has current, voltage and status indicators. The XSA board needs to be designed such that it can fit inside the X-Ray head and near to all sensors thus leaving only the high voltage power control board inside the XRD box. In this thesis, the current design of the XRD box is used without any modifications.

The operation of the entire machine is controlled by an application running on a PC. The PC is connected with M-Code Plus smart motors and the PLC through an RS-485 bus as shown in Figure 6. The application running on the PC directly controls the goniometry system by directing the movement of each M-drive motor through M-codes [21]. PC sends commands such as open/close shutter to the PLC board over the same RS-485 bus. It also requests frequent status updates from the PLC board. The XSA board must be designed and implemented to be compatible with this control application. The PC hooks up to the RS-485 bus through a USB to RS-485 converter.
2.5 **RS-485 Communication Protocol**

The X-Ray stress measurement system discussed in this thesis uses RS-485 communication for exchange of messages between different operating modules of the system. The background information necessary to understand this communication protocol is presented in this section.

The RS-485 standard, TIA/EIA-485, specifies balanced data-transmission schemes that offer robust solutions for transmitting data over long distances and noisy environments [22]. RS-485 only specifies electrical characteristics of the generator and the receiver. It does not specify or recommend any communications protocol other than the physical layer. The foreword to the standard recommends The Telecommunications Systems Bulletin 89 (TSB-89), which contains application guidelines; including data signaling rate vs. cable length, stub length, and configurations [23]. Section 4 defines the electrical characteristics of the generator (transmitter or driver), receiver, transceiver, and system. These characteristics include the following items: definition of a unit load, voltage ranges, open circuit voltages, thresholds, and transient tolerance. It also defines three generator interface points (signal lines): "A", "B" and "C". The data is transmitted on "A" and "B". "C" is a ground reference. This section also defines the logic states 1 (off) and 0 (on), by the polarity between A and B terminals. If A is negative with respect to B, the state is binary 1. The
reversed polarity (A +, B −) is binary 0. The standard does not assign any logic function to these two states.

RS-485 supports inexpensive local networks and multi-drop communications links. Since it uses the differential balanced line over twisted pair RS-485 can communicate distances up to 1,200 m (4,000 ft). It offers data transmission speeds of up to 35 megabits per second (Mbps) for distances up to 10 m, and 100 kilobits per second (kbps) for distances up to 1200m [23]. RS-485 drivers use 3-state logic allowing individual transmitters to be deactivated. This allows RS-485 to implement linear bus topologies using only two wires. The 3-state logic allows an output port to assume a high impedance state in addition to the zero (0) and one (1) logic levels, effectively removing the output from the circuit. This allows multiple circuits to share the same output line or lines.

The equipment located along a set of RS-485 wires is interchangeably called nodes, stations or devices. The recommended arrangement of the wires is a connected series of point-to-point (multi-dropped) nodes as shown in Figure 7.

![Figure 7 - RS-485 configuration](image-url)
Since RS-485 does not specify any protocol, the firmware on the node is responsible for choosing the data frame. Typically, a data frame that consists of a start bit, 8 data bits and a stop bit as shown in Figure 8 is chosen.

![Figure 8 - RS-485 data frame](image)

The RS-485 bus can function in full-duplex or half-duplex mode. In half-duplex mode, two lines as shown in Figure 7 are used. In this mode, two devices connected in the bus can communicate with each other but in one direction at a time like a walkie-talkie. In contrast to this, in the full-duplex mode, two devices can communicate with each other simultaneously like a telephone. RS-485 uses four wires for this type of full-duplex communication.

### 2.6 ATMELE ATMEGA-2560 Microcontroller

The Atmel ATMEGA-2560 MCU is used on the XSA board proposed in this thesis. The ATMEGA-2560 microcontroller is an 8-bit microcontroller based on AVR enhanced RISC architecture [24]. It has 32 general purpose 8-bit registers and 135 powerful system instructions in which most of them take single clock cycle to execute. It has 16 Million Instructions Per Second (MIPS) throughput at 16MHz frequency. ATMEGA-2560 is a Thin Quad Flat Package (TQFP) with 100 pins, 25 on each side of the package as shown in Figure 9 below.
Figure 9 - ATMEL-2560 pin diagram

2.7 ATMEGA-2560 Programming Interface

A Serial Peripheral Interface (SPI) is designed to program the ATMEGA-2560 MCU used on the XSA board. The SPI interface has six signals as shown in Figure 10. The SPI protocol [25] works in master–slave configuration. In this thesis, the ATMEGA-2560 MCU is configured as a slave and the PC is configured as a master. The Master In Slave Out (MISO) and the Master Out Slave In (MOSI) signals shown in the figure are connected to the corresponding pins on the MCU. The SCK is the clock signal and is controlled by the bus master, which is the PC. The active low RESET signal, which is also called as slave selection signal, is used to select one among many slaves for communication on SPI bus. In the board designed in this thesis, the slave selection signal is not used, as there is only one slave on the bus and is always selected by default.
2.8 Summary

In this chapter, few limitations of PLCs and challenges faced by OEMs by using programmable logic controllers for automation of their machines are discussed. PLC’s design philosophy, execution format and their poor response time are discussed in detail. Different proposals to overcome these limitations by using PLC alternatives are reviewed. Through the review of these proposals, the idea of designing MCU-based control system for automation of iXRD stress measurement system is supported.
3. DESIGN AND IMPLEMENTATION

3.1 Introduction

The design and implementation of the MCU-based X-Ray System Automation board proposed in this thesis are carried out in the following phases:

- Requirements analysis
- System design
- Hardware design
- Software design and implementation

A detailed discussion of these phases is presented in this chapter.

3.2 The Design and Implementation Methodology

3.2.1 Requirements Analysis

The MCU-based embedded control system proposed in this thesis for the automation of the Proto Manufacturing X-Ray stress measurement system shall

- Have faster response time as compared to PLC-based automation
- Reduce system complexity and improve scalability
- Reduce system integration and maintenance cost

In order to have faster response time, firstly, an MCU that can respond quickly to the changes in input signal must be selected. Secondly, the hardware circuits that interface different sensors to the MCU must be carefully designed to minimize the signal propagation delay. Finally, the MCU software must be designed to quickly respond to changes on the input pins.
The existing system has many wires running from the X-Ray head to the XRD box where the PLC is located. RS-485 communication lines that connect PLC, PC and M-drive motors are also running along the same path. This has increased the wiring complexity in the system. To eliminate this wiring complexity the XSA board designed in this thesis must be small enough to fit inside the X-Ray head. By placing the XSA board inside the X-Ray head, all wires running from head to XRD box can be eliminated to get clean and simple wiring. With this change, scalability of the system is improved as new sensors can be easily added to the system for more features when desired.

Eliminating the PLC from the system results in a significant cost reduction as PLC is the highest cost factor in all OEM automation solutions. To achieve further cost savings, components used in the design of the XSA board must be carefully chosen. Achieving reduced wiring complexity is also necessary to reduce integration and maintenance cost.

Following sections of this chapter will explain in detail how these requirements are addressed while designing the XSA board.

3.2.2 System Design

The architecture of the X-Ray stress measurement system with the XSA board designed and implemented in this thesis is as shown in Figure 11. It is evident that the overall architecture of the system is the same as the existing system while PLC is replaced with the XSA board and the wiring is reduced. The architecture of the RS-485 bus and communication between PC and M-drive motors are not altered and hence are not discussed in detail further in this thesis. The hardware design, the software design and implementation of the XSA board designed in the thesis are discussed in detail in the following sections.
3.2.3 XSA Board Design

The block diagram of the XSA board designed in thesis is as shown in the Figure 12. The ATMEGA-2560 MCU is interfaced to auto focus, shutter present, shutter status, temperature and high voltage sensors through dedicated signal conditioning circuits for each sensor. Different hardware modules used in designing this hardware board are discussed below.
3.2.3.1 MCU Selection

Choosing the right MCU for the design of the XSA board was the one of the important parts of this thesis. The selected MCU was supposed to have small package as the size of the XSA board itself had a limitation not to exceed the size of the head. In addition, firstly, it should have at least 8MHz operating frequency and support multiple sensor interfaces. Secondly, it should be a low cost MCU and easily available in the market. Finally, it should be easy to program.

After significant study, Atmel’s ATMEGA-2560 MCU was chosen because, it has the maximum operating frequency of 16MHz, has 100 pins to support different types of sensor interfaces. It is also a low cost MCU with high availability. It is available in Thin Quad Flat Package (TQFP), which is small and takes less space on the PCB. ATMEGA-2560 can be easily programmed in C programming language using Atmel’s Atmel Studio Integrated Development Environment (IDE) tool.

3.2.3.2 Sensors Interface

For the ATMEL-2560 microcontroller to be able to read various system parameters such as temperature, sensors must be connected to the microcontroller through proper hardware interfaces. These interfaces provide the power, grounding and other signal inputs necessary for the sensor to convert the physical quantity such as temperature to electrical signals that are fed to the microcontroller. The design of various hardware interfaces used in the proposed XSA board is discussed in the sections that follow.

Temperature sensor interface

The AD590 [26] is the temperature transducer used in the design of the XSA board and it produces current proportional to the absolute temperature. It has a wide operating temperature range from -55°Centigrade to +150°Centigrade. The temperature signal from the sensor that is residing inside the X-Ray head is fed to the temperature sensor circuit through pin 5 of the DB15 connector as shown in Figure 13. The output of this circuit is connected to a comparator stage as shown in Figure 14 before getting connected to the external interrupt pin INT0 [24] of the microcontroller. The comparator produces a high (1) or low (0) output based on the reference voltage (Vref) set at voltage corresponding to 70°Centigrade.
Proximity Sensor (E2EC-CR5C1 2M)

To measure the residual stress in the material under test, the X-Ray head must be positioned to a predefined focus position. I implemented this functionality in the proposed system by using the E2EC-CR5C1 2M proximity sensor [27] in the X-Ray head. The output of this sensor is connected to the digital input pin of the microcontroller. When the autofocus is requested, the controlling application in the PC requests XSA board for the
status from the proximity sensor. The controlling application then moves the X-Ray head until it is at a known distance from the object being scanned. The head will then move up to the predefined focus position to complete the autofocus process. The schematic of the proximity sensor is shown in Figure 15.

![Figure 15 - Proximity Sensor](image)

**Photomicrosensor (EE SX1018)**

Before starting a measurement, the X-Ray stress measurement system needs to check the presence of X-Ray tube and shutter. To implement this requirement, I designed simple circuits as shown in Figure 16 using EE SX1018 photomicrosensors [28] and [29]. The sensor circuits produce a high level (1) when the presence of the tube or shutter is detected and low level (0) when the absence is detected. The outputs of these sensors are connected to digital input pins of the MCU. Software will read these pins when it gets request from XRD software and updates the status back to the software.
The control application running on the PC needs to know the status (Open/Close) of the shutter before it could start a measurement. I implemented this requirement also using the same EE SX1018 photomicrosensor. The output of this circuit is connected to an input pin of the MCU and read by the software whenever the control application requests for shutter status.

![Photomicrosensors circuit](image)

**Figure 16 – Photomicrosensors circuit**

**DRV101:U9 – Solenoid driver**

I designed a circuit to drive the LEDEX rotary solenoid used in controlling the opening and closing of the X-Ray tube shutter using the solenoid driver DRV101 [30]. The DRV101 is a low-side power switch employing a Pulse-Width Modulated (PWM) output and its schematic is shown in Figure 17.
The input pin (1) of DRV101 is connected to a digital output pin of microcontroller and is controlled by the software. When the “Shutter Open” command is received from the control application running on the PC, the XSA software will set high (1) level on this pin to open the shutter. When it receives the “Shutter Close” command, it will set low (0) on this pin to close the shutter.

It is required to have high pull-in current to drive the solenoids because they have a much higher pull-in current requirement than hold requirement. I achieved this by connecting a capacitor between pin (2) and ground. The capacitance of this delay adjustment capacitor is calculated by using Equation-1.

\[
\text{Delay Time} = C \times 10^6 \text{ s} \quad \text{(Time in second, C in Farad)}
\]  

In the proposed XSA board, for a preset delay of 0.22 milliseconds a 0.22 nf (nano farad) capacitor is connected in this circuit.

The duty cycle is adjusted by connecting pin (3) to the input of a comparator and a 19kΩ resistor to ground. It is driven by a 200\(\mu\)A current source from \(V_S\). The voltage at this node linearly sets the duty cycle. The duty cycle can be programmed with a resistor, analog voltage, or output of a Digital to Analog Converter (DAC). The active voltage range is from 0.75V to 3.7V to facilitate the use of single-supply control electronics. At 0.75V (or RPWM = 3.5kΩ), duty cycle is near 90%. In the proposed system, a resistor is used to
program the duty cycles and the required resistor value is calculated as 25kΩ using Equation-2.

\[
RPWM = \left[ a + b \; (DC) + c \; (DC)^2 + d \; (DC)^3 + e \; (DC)^4 \right]^{-1}
\]

Where: 
\[
a = 2.4711 \times 10^{-6}, \quad b = -5.2095 \times 10^{-7}, \quad c = 4.4576 \times 10^{-8}
\]
\[
d = -7.6427 \times 10^{-10}, \quad e = 6.8039 \times 10^{-12}, \quad DC = \text{Duty Cycle}
\]

At a temperature T = +25°C and supply voltage = +24Vc

**RC low-pass filter**

While designing the XSA board, there was a problem of high frequency noise from the main DC input signal, sensors and the through holes in PCB board. This high frequency noise affected signal levels on the system and the voltage regulator and solenoid driver. To eliminate this high frequency noise, I designed a Resistor-Capacitor (RC) low-pass filter [31], as shown in Figure 18. This filter passes signals with a frequency lower than a certain cutoff frequency and attenuates signals with frequencies higher than the cutoff frequency. In the proposed system, I designed an RC-low pass filter with the cutoff frequency of 50Hz using the equation \( RC = \frac{1}{2\pi f} \). With \( f=50\text{Hz} \), RC product is 0.00318FOhm which is approximately achieved using a 30kΩ resistor and a 100nf capacitor.

![Figure 18 - RC-low pass filter](image)

**Voltage Regulator**

While designing the board, I found that the main DC power supply is giving 12V output but the ATMEGA-2560 MCU operates at 5V DC. To solve this problem, I needed to convert 12V DC to 5V DC. Therefore, I designed a voltage regulator circuit using...
KA78M05TUFS [32] from Fairchild Semiconductor which gives a steady 5V DC output for inputs ranging from 12V to 18V.

**Temperature comparator**

The X-Ray stress measurement systems should not operate if the temperature inside the X-Ray head is more than 70°C. This is a requirement taken from the existing system. This could have been implemented in either software or hardware. Implementing this requirement in software would mean using lot of processing time of MCU which could affect the systems response time. Therefore, I decided to design a temperature comparator circuit using the LM339 [33], that could interrupt the MCU when the temperature inside the X-Ray head is greater than 70°C.

In the proposed system, this device is used to compare the output voltage (TubeTemp) of temperature sensor AD590 with the reference voltage (Vref) corresponding to a pre-set temperature value which is 70°C. The temperature comparator circuit used in the proposed system is as shown in Figure 19. The circuit will output a logic low or high impedance (logic high with pull-up) based on the inputs difference. The output of this circuit is connected to an interrupt pin of the ATMEGA-2560 MCU. This interrupt is triggered when the temperature is greater than 70°C and Interrupt Service Routine (ISR) is invoked. The ISR will close the X-Ray head shutter to stop the measurement.

![Temperature comparator circuit](image)

Figure 19 - Temperature comparator circuit
RS-485 Transceiver

The XSA board is required to communicate with the monitor and control application running on the PC through the existing RS-485 communication bus. The Universal Synchronous Asynchronous Receiver Transmitter (USART) module of the ATMEGA-2560 micro controller can be used for serial communications such as RS-232, RS-422 and RS-452. The USART module has RX (Receiver) and TX (Transmitter) pins provided on the microcontroller for this purpose. However, these pins cannot be directly connected to any serial bus because these pins are at different voltage levels and encoding than that of bus signals. Therefore, I decided to use an external RS-485 bus transceiver. In the proposed system, the MAX3491 RS-485 transceiver[34] is used to connect the XSA board to the RS-485 bus. A transceiver such as MAX3491 converts the voltages levels to the desired bus voltages and desired encoding. The pin diagram of the MAX3491 is as shown in Figure 20. In the proposed system, the RX and TX pins of the microcontroller are connected to receive out (RO) and data in (DI) pins of the transceiver respectively. In software, the USART module is configured to communicate at 9600 baud rate using one start bit, eight data bits and one stop bit. In order to achieve the flow control during communication, two digital output signals of the microcontroller are used and are connected to transmit enable (DE) and receive enable (RE) pins of MAX3491. The digital output controlling the receiving is pulled low in hardware so that receiving is always enabled. The digital output controlling the transmission is managed in software and is pulled high only when the microcontroller needs to transmit otherwise it is held low.
I designed the schematic and PCB layout of the XSA board with provisions to connect all sensors and RS-485 communication lines. The hardware schematic and the PCB layout of the XSA board designed with this 100 pins ATMEL-2560 is as shown in Figure 21. The PCB of XSA board has three layers with many through holes running from one side to the other. Numbers 1 through 7 on the PCB show locations of hardware components discussed in this chapter. Each number represents one hardware component as listed in the Table 1.
3.2.4 SW Design and Implementation

The software for the ATMEGA-2560 microcontroller that handles the operation of the X-Ray system automation board is designed and developed by following standard industry software development procedures. Firstly, a high-level software component model is laid out to identify different software modules/components needed and to classify them into different layers of software. Secondly, a software behavioral model is then developed to determine the high-level execution sequence of the software. Finally, low-level software modules are identified, designed and coded to complete the software development. The
popular C programming language; which is widely used for coding embedded controller software, is used in writing software code of the X-Ray system automation board. In the development phase, module level (unit testing) and integration testing are performed. A detailed system testing is then carried out to analyze the in-system performance of the XSA board hardware and software in real time. Detailed information about software layers, software modules, and their execution pattern is given in this section.

3.2.4.1 SW Architecture

Structurally, the software of the X-Ray automation board is divided into two layers: application layer and middleware layer. The software modules that contain the source code for handling the core machine functions such as closing and opening the X-Ray head shutter are classified as the application layer. The modules that link the application layer with the hardware by providing access to the microcontroller resources such as timers, digital inputs etc. are middleware layer modules.

Different behavioral software architectures as discussed in [35] are in use in designing software for embedded control systems. For the design and development of the software for the X-Ray system automation board proposed in thesis, “round robin with interrupts” architecture is the most applicable because the system has very minimal set of functionalities that can be implemented on the ATMEGA-2560 microcontroller without any real-time operating system. The “round robin with interrupts” architecture is preferred in this thesis over the basic “round robin” architecture because it suffers from the same sequential execution problem of the PLC that was discussed in earlier chapters in this thesis. Using the sequential execution for low priority tasks and interrupts for high priority tasks improves the system response time for critical events.

Software Component Diagram

The arrangement of the X-Ray system automation board’s software modules into different architectural layers is as shown Figure 22. All the software needed for control and execution of X-Ray stress measurement system’s functions is in application layer modules. The main loop, which is the entry point for the entire control system software and the backbone of the round-robin execution structure, is part of the application layer. All ISRs that add the parallel execution feature to the XSA board software is another important
module in the application layer. The software modules that process the RS-485 communication messages and handle Light Emitting Diode (LED) and switch are also placed in the application layer.

The application layer software components use middleware components mainly to interact with the microcontroller and in turn with the hardware and sensors of the X-Ray stress measurement system. The middleware also provides the utility functions and data structures such as serial communication buffers, message queues etc. to the application layer. In the XSA board software, the middle layer consists of three major components namely, timer initialization, interrupt setup and serial USART communication. There is also an SPI communication module and a miscellaneous initialization module available in middleware for the use of application layer functions.
The round robin with interrupts behavioral architecture of the XSA board software is shown in Figure 23. The execution of the XSA board software starts from the main function which initializes all system peripherals and resources needed for other modules to run. It then enters into an infinite loop, which runs an idle task that just keeps looping without any particular control activity. When the main function has finished execution, all interrupts are initialized and these interrupts can occur periodically or in case of a predefined event.
Table 2. When any interrupt occurs, the execution will jump from the idle task to interrupt service routine of that interrupt. This switching will happen based on the entries in the interrupt vector table initialized in ATMEGA-2560 microcontroller by the software. The ATMEGA-2560 microcontroller has 57 interrupt vectors each mapped to an interrupt source [24]. These 57 interrupt vectors, corresponding addresses and sources are listed in APPENDIX A. In software, for all used interrupts, each location of the vector table is updated with the address of the corresponding ISR function. Interrupt at the top of the interrupt vector table has the highest priority and the priority decreases as the interrupt vector number increases.
<table>
<thead>
<tr>
<th>Interrupt Name</th>
<th>Description</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Over temperature</td>
<td>Occurs when the temperature inside the X-Ray head is more than 70°Centigrade</td>
<td>0</td>
</tr>
<tr>
<td>interrupt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shutter not OK</td>
<td>Occurs if the shutter has been removed from its place on the head</td>
<td>1</td>
</tr>
<tr>
<td>interrupt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Timer0 interrupt</td>
<td>A periodic interrupt that occurs every 25 milliseconds</td>
<td>3</td>
</tr>
<tr>
<td>RS-485 receiver</td>
<td>When a character is received by the ATMEGA-2560 microcontroller that was</td>
<td>4</td>
</tr>
<tr>
<td>interrupt</td>
<td>transmitted on RS-485 bus by PC this interrupt is triggered</td>
<td></td>
</tr>
<tr>
<td>RS-485 transmit</td>
<td>When the XSA board software wants to send an RS-485 message, it instructs</td>
<td>5</td>
</tr>
<tr>
<td>interrupt</td>
<td>the USART module of the microcontroller by copying the byte to be transmitted</td>
<td></td>
</tr>
<tr>
<td></td>
<td>on to transmit data register. On completing transmission of this byte, this</td>
<td></td>
</tr>
<tr>
<td></td>
<td>interrupt is triggered. The ISR of this interrupt will copy the next byte</td>
<td></td>
</tr>
<tr>
<td></td>
<td>to be transmitted into transmit data register. This process will continue</td>
<td></td>
</tr>
<tr>
<td></td>
<td>until all bytes are transmitted</td>
<td></td>
</tr>
</tbody>
</table>
completely and USART transmit interrupt is disabled

3.2.4.2 Application Layer

The application layer of the XSA board software has following software modules:

i. Main
ii. Interrupt service routines
iii. RS-485 message processing
iv. LED/Switch handlers

Each of these modules contain functions written in C programming language. Algorithm of each function, inputs taken and outputs produced are discussed in the following sections.

Main

The main function is the entry point to the XSA board application software. The program execution starts from this function on power up after some basic initializations are performed by the software in the boot up sequence. A detailed control flow of the main function is shown in Figure 24. This function initializes the PORTB as an output port and PORTD as input port by calling initPorts function to enable the rest of the software to handle LEDs connected to PORTB and switches connected to PORTD. It also initializes the Serial Communication Interface (SCI) and Timer0 of the microcontroller by calling SCIInitialize and ISR_InitTimer0 routines respectively. Interrupts are disabled while these initialization steps are executed to prevent occurring of undesirable events. This function then enters into an infinite loop, which is implemented using a “for loop” and just rolling without performing any control or processing activity. When an interrupt occurs, the execution will switch to the interrupt service routine and control is transferred back to this function on completion of ISR. Because of the infinite for loop, the ‘return’ statement in this function will never be executed.
Figure 24 - Control flow of main function
Interrupt Service Routines

For each interrupt listed in Table 2, an interrupt service routine is designed in this thesis. The algorithms of these interrupt service routines are explained in the below sections.

**Timer0 compare ISR**

The Timer0 compares ISR executions every 25 milliseconds and controls the execution of the heartbeat task and the RunMenu task. The heartbeat task is set to run two times in a second and it toggles LEDs on the machine to indicate that software is running and the machine is working normally. The RunMenu task which handles all the RS-485 communication is set to run every 100 milliseconds. The Timer0 ISR achieves this scheduling by incrementing an internal counter uiMedThreadCount and comparing it against the preset scheduling times. The detailed control flow of this function is shown in Figure 25.
Figure 25 - Control flow of Timer0 interrupt
**INT2 ISR**

This is the ISR for the shutter not OK interrupt that occurs if the shutter is not detected on the head. The ISR just calls the close function to update the digital output status and send a message to the PC to indicate the failure condition.

```
void close()
{
    SCWriteString_P(PSTR("some error happened, sorry, the shutter is closed\n\n"));
    END
}
```

**INT0 ISR**

This is the ISR for the over temperature interrupt that occurs when the temperature inside the X-Ray head is more than 70 °C. This ISR, like INT2 ISR, just calls the close function to update the digital output status and sends a message to the PC to indicate the failure condition.

```
void close()
{
    SCWriteString_P(PSTR("some error happened, sorry, the shutter is closed\n\n"));
    END
}
```
USART0_UDRE_vect

The USART0_UDRE_vect is the interrupt service routine for transmitting the specified number of bytes over the RS-485 bus. This interrupt occurs when the transmit data register UDR0 of ATMEGA-2560 is empty. Inside the ISR, the next bit to be transmitted is copied into the transmit data register and the interrupt is enabled so that new transmit register empty interrupt is received after the transmission of this byte. The data to be transmitted is available in the circular buffer zOutputChars that is managed using read and write pointers ptrOutputCharHead and ptrOutputCharTail respectively. When they are equal, the transmission is stopped as the buffer is empty.
Figure 26 - Control flow - RS-485 transmit ISR
USART0_RX_vect

This interrupt service routine is triggered when any data is received on the USART0, which is receiving the RS-485 messages. This interrupt occurs for every character received. The received character is available in the UDR0 register and copied to the received circular buffer zInputChars. Reading and writing of this circular buffer is managed by two pointers ptrInputCharHead and ptrInputCharTail respectively. These characters are consumed by the RunMenu function to process the received RS-485 message. If the consumption of the characters is slower than the rate at which characters are received, an overflow condition is detected and received characters can get lost. This condition is prevented by executing the RunMenu task at 100 milliseconds rate. The complete control flow of the USART0_RX_vect is shown in Figure 27.
Figure 27 - Control flow - RS-485 receive ISR
**RS-485 Message Processing**

The RunMenu task which runs every 100 milliseconds is responsible for processing all RS-485 messages. “Open Shutter”, “Close Shutter” and “Status” are RS-485 messages that the XSA board receives from the PC. For each received message, the board sends a response message that has success, failure or status information. The RunMenu function starts by reading all the characters available in the received circular buffer followed by checking the received characters for “AA” which is the RS-485 node address of the XSA board. If the address is matched, it continues to analyze the received characters to identify the received message. If the received message is “Open Shutter”, the Sensors function is called to check the status of X-Ray tube and shutter. When the Sensors function returns all OK status, the shutter is opened and a “Success” message is sent to the PC. Similar steps are followed in processing the “Close Shutter” message. When PC requests “Status” message, the Status function is called to check the status of X-Ray tube and shutter. On completing the status check, the function will send “Tube is good” or “Tube is open”, and “Shutter is open” or “Shutter is closed” messages to the PC.

**LED/Switch handling functions**

The initPorts function is implemented to initialize PORTB of the ATMEGA-2560 microcontroller as the output port and PORTD as the input port. This is achieved by writing the corresponding port Data Direction Register (DDR) [24]. Setting the DDR register to 0x00 configures a port as an input port and the value 0x11 configures the port as an output port.

### 3.2.4.3 Middleware Layer

The middleware layer of the XSA board software has following software modules:

i. Timer functions

ii. Interrupt setup functions

iii. Serial communication (USART) functions

iv. SPI communication function

v. Miscellaneous controller initialization functions

The implementation details of these functions are discussed in the below sections.
Timer Functions

The ISR_InitTimer0 function is implemented to initialize the Timer0 module of the microcontroller to produce a periodic interrupt of 25 milliseconds. A value of 197 is loaded to the Output Compare Register (OCR) [24] of Timer0 to achieve the 25 milliseconds duration with the 8 MHz Central Processing Unit (CPU) clock used in the design of the XSA board.

Interrupt Setup functions

A simple ClearInterrupt function is implemented to clear the external interrupt flags in the microcontroller as per needs of the application. Clearing of interrupts is needed inside the interrupt service routines to indicate to the controller that the interrupts are serviced and they can be re-enabled.

Serial Communication (USART) functions

Three functions are implemented in this thesis for the operation of the serial communication interface to send and receive RS-485 messages. The SCIInitialize function initializes the USART0 of the ATMEGA-2560 microcontroller with 9600 baud rate, 8 data bits and odd parity configuration. It also enables the receive interrupt and initializes receive and transmit circular buffers.

The SCIWriteString function takes a null-terminated string as input and copies it to transmit circular buffer zOutputChars. It then enables the USART0 transmit interrupt to start the transmission of input string one character at a time until all characters are transmitted.

The SCIReadChar function checks if the receive circular buffer zInputChars is empty and if it is not, it returns the character at the top of the buffer to the calling function. It then moves the buffer read pointer to the next character.

SPI functions

An SPIInit function is implemented to initialize the SPI communication needed for programming of the ATMEGA-2560 microcontroller. It configures the microcontroller as the SPI slave device to receive commands from the PC application, AVR Studio, during
programming. It also sets clock phase, polarity and clock rate parameters by writing the desired values to the SPI Control Register (SPCR) [24].

**Miscellaneous Controller Initialization Functions**

In addition to the functions discussed in different sections above, simple utility functions to select the functionality of each pin, initialization of the clock are implemented in the XSA board software.

### 3.3 Summary

In this chapter, the design and implementation of the proposed XSA board is discussed in detail. The discussion starts by analyzing the existing X-Ray stress measurement system with PLC automation and identifying the design requirements for new MCU-based automation board. The hardware design of the MCU-Based XSA automation board is then discussed. Selection of different hardware components, design of the analog circuits is discussed in details. The architecture of the software that runs on the XSA board along with detailed explanation of important software modules is given in this chapter.
4. RESULTS AND CONCLUSION

4.1 Introduction

In this section, the response time of the PLC for changes in the input are compared with the response time of the XSA board for same inputs. A test application was developed to measure the response time of the MCU-based XSA board. The complexity and scalability of the system is analyzed with reference to the PLC-based system. The conclusion of this thesis and future work are discussed at the end of the chapter.

4.2 Response Time Analysis

The response times of the PLC and the MCU-based XSA board for different sensor inputs are listed in the Table 3.

<table>
<thead>
<tr>
<th>Sensors</th>
<th>Response Time in milliseconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>XSA Board</td>
<td>PLC Automation</td>
</tr>
<tr>
<td>Temperature Sensor</td>
<td>0.087</td>
</tr>
<tr>
<td>Temperature Sensor and Shutter OK Sensor</td>
<td>0.172</td>
</tr>
<tr>
<td>Temperature Sensor, Shutter OK and Tube OK Sensor</td>
<td>0.264</td>
</tr>
</tbody>
</table>

From Table 3, it can be noted that the response times of MCU-based XSA board are far smaller than that of PLC. When the temperature sensor detects that the temperature inside the X-Ray head is more than 70°C, the XSA board responds to it with in 0.087 milliseconds.
Meanwhile, the PLC would take 9.67 milliseconds to respond to the same event. It should also be noted that, the PLC would scan the temperature input during every input scan process. However, the MCU would work on the temperature sensor input only when temperature is greater than 70°C.

When any event occurs independently, there is no competition for the MCU time with other events and the MCU will respond to it in 0.087 milliseconds. When two or more events occur at the same time or an event occurs when another event is being processed, the MCU will handle events based on their priority. Multiple sequences of events are possible with multiple inputs. In the following section two sequences are discussed to understand the priority based event handling.

**Sequence 1: Shutter not OK event occurs, while the over temperature event being processed**

When the events occur in this sequence, the processing of over temperature event will continue without any affect, because, the shutter not OK even is assigned with lower priority than the over temperature event in the MCU software. When the MCU finishes processing the over temperature event, it will start processing the shutter not OK event and it would take 0.0172 milliseconds to start processing it.

**Sequence 2: Over temperature event occurs while the shutter not OK event being processed**

In this sequence, since I have assigned the highest priority to the temperature sensor input the MCU will give immediate attention to it and start handling over temperature event by pushing the processing of shutter not OK event to the background. Once it completes handling the over temperature event, it will resume the processing of the shutter not OK event.

This sequence of handling events is applicable when more than two interrupts are involved in a scenario. In each sequence, the least priority interrupt will get processed the last and will have the slowest response time.
4.3 System Complexity Analysis

Currently, the head contains five sensors. Each sensor connection needs a dedicated data line, a power supply line and a ground line. A common power supply line and a ground line are used for all sensors. In addition, there are four RS-485 communication lines connecting PLC, M-Drive motors and PC. Thus, many wires are running from the X-ray head to the XRD box in a complex way. This wiring complexity is reduced significantly in the proposed design. The small form factor of the XSA board allows it to be placed inside the X-Ray head itself. Hence, all wires that were running from the X-Ray head up to the XRD box to connect different sensors to the PLC are now not needed. The proposed design has only 12V power line, a ground line running from X-Ray head to the XRD box. There are also four communication lines that run from XRD box to the M-drive motors and PC.

4.4 System Scalability Analysis

Adding new sensor to the existing system will increase the complexity of the system as a new wire is added to connect the sensor to the PLC. On the other hand, adding a new sensor to the system with the proposed design, does not add any new wiring between the X-Ray head and the XRD box. All wiring needs are within the X-ray head and only short lengths of wires are enough to fulfill this requirement. From the outside view, the wiring complexity in the proposed design does not increase by adding new sensors to the system. This complexity variation with system scalability is illustrated in the Figure 28. In this graphical plot, it is assumed that two wires add one to the complexity scale.
4.5 Cost Savings Analysis

A comparison of important components and their approximate costs is given in the Table 4. In PLC based automation solutions, the PLC cost is the major cost factor. In the automation of the X-Ray stress measurement system discussed in this thesis, a Trilogic PLC was used. The PLC alone costs $350. Thus, elimination of the PLC gave an upfront cost saving of $350. When PLC is eliminated, some of the associated components like PLC rack are also eliminated to add further cost savings. On the other hand, the main processing unit of the XSA board, the ATMEGA-2560 MCU cost only $8. All the major components of the XSA board including ATMEGA-2560 cost only $88 in total. Thus a significant cost saving is achieved by using MCU-based XSA board for the automation of the X-Ray stress measurement system.
<table>
<thead>
<tr>
<th>Components</th>
<th>Price (in $)</th>
<th>Components</th>
<th>Price (in $)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trilogic PLC</td>
<td>350</td>
<td>ATMEGA-2560</td>
<td>8</td>
</tr>
<tr>
<td>Communication Components</td>
<td>100</td>
<td>Other Components</td>
<td>10</td>
</tr>
<tr>
<td>Rack</td>
<td>100</td>
<td>PCB</td>
<td>30</td>
</tr>
<tr>
<td>Wiring Cost (5 sensors; 15 wires – 30 meter)</td>
<td>50</td>
<td>Soldering</td>
<td>40</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>600</strong></td>
<td><strong>88</strong></td>
<td></td>
</tr>
</tbody>
</table>

### 4.6 Conclusion and Future work

The X-Ray system automation board proposed and designed in this thesis is an effective replacement to the PLC-based automation framework and meets all the system design requirements. Through computation and analysis, the proposed design has improved the system response time by employing an interrupt driven software architecture to cater to the higher priority inputs. The wiring complexity of the system is significantly reduced as the XSA board can be now placed inside the X-Ray head because of the small form factor of the MCU-based board. With the reduced wiring complexity, the scalability factor of the system is improved. New sensors can be added to the proposed system without any apparent wiring changes. Elimination of high-cost PLC and reduced wiring requirement has drastically reduced the system cost. The total cost of the new system is less as it is designed from scratch using basic components needed to fulfill the design requirements.

The performance of the XSA board can be further improved by using a basic 32-bit micro controller that has higher operating frequency and a real-time operating system with multi-tasking feature. A boot loader can be developed to add in-system software upgrade.
and diagnostics feature to easily maintain the system. The design philosophy of the XSA board can be used as a reference to develop embedded process automation systems by any OEM wishing to design custom automation solutions. Designing customized and specific automation solutions help OEMs to control the cost, performance of their system and eliminate dependency on third party solutions. The proposed XSA boards have a capability to network, which can allow system to be remotely monitored and controlled. This can be done through the internet, which can open new business opportunities for the OEMs.
REFERENCES:


http://www.chm.bris.ac.uk/webprojects2001/cdavies/xrays.htm

http://www.protoxrd.com/products.html


    http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-
    ATmega640-1280-1281-2560-2561_datasheet.pdf


    https://www.ia.omron.com/product/item/1017/

    https://www.omron.com/ecb/products/photo/34/ee_sx1018.html


    Voltage Regulator.


    tml


[36] Fairchild Semiconductor Corporation. (2015, July) 3 Terminal 0.5A Positive
    Voltage Regulator.
A. **Interrupt Vectors in ATMEGA-2560**

<table>
<thead>
<tr>
<th>Vector No.</th>
<th>Program Address</th>
<th>Source</th>
<th>Interrupt Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$0000</td>
<td>RESET</td>
<td>External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset</td>
</tr>
<tr>
<td>2</td>
<td>$0002</td>
<td>INT0</td>
<td>External Interrupt Request 0</td>
</tr>
<tr>
<td>3</td>
<td>$0004</td>
<td>INT1</td>
<td>External Interrupt Request 1</td>
</tr>
<tr>
<td>4</td>
<td>$0006</td>
<td>INT2</td>
<td>External Interrupt Request 2</td>
</tr>
<tr>
<td>5</td>
<td>$0008</td>
<td>INT3</td>
<td>External Interrupt Request 3</td>
</tr>
<tr>
<td>6</td>
<td>$000A</td>
<td>INT4</td>
<td>External Interrupt Request 4</td>
</tr>
<tr>
<td>7</td>
<td>$000C</td>
<td>INT5</td>
<td>External Interrupt Request 5</td>
</tr>
<tr>
<td>8</td>
<td>$000E</td>
<td>INT6</td>
<td>External Interrupt Request 6</td>
</tr>
<tr>
<td>9</td>
<td>$0010</td>
<td>INT7</td>
<td>External Interrupt Request 7</td>
</tr>
<tr>
<td>10</td>
<td>$0012</td>
<td>PCINT0</td>
<td>Pin Change Interrupt Request 0</td>
</tr>
<tr>
<td>11</td>
<td>$0014</td>
<td>PCINT1</td>
<td>Pin Change Interrupt Request 1</td>
</tr>
<tr>
<td>12</td>
<td>$0016</td>
<td>PCINT2</td>
<td>Pin Change Interrupt Request 2</td>
</tr>
<tr>
<td>13</td>
<td>$0018</td>
<td>WDT</td>
<td>Watchdog Time-out Interrupt</td>
</tr>
<tr>
<td>14</td>
<td>$001A</td>
<td>TIMER2 COMP</td>
<td>Timer/Counter2 Compare Match A</td>
</tr>
<tr>
<td>15</td>
<td>$001C</td>
<td>TIMER2 COMP</td>
<td>Timer/Counter2 Compare Match B</td>
</tr>
<tr>
<td>16</td>
<td>$001E</td>
<td>TIMER2 OVF</td>
<td>Timer/Counter2 Overflow</td>
</tr>
<tr>
<td>17</td>
<td>$0020</td>
<td>TIMER1 CAPT</td>
<td>Timer/Counter1 Capture Event</td>
</tr>
<tr>
<td>18</td>
<td>$0022</td>
<td>TIMER1 COMP</td>
<td>Timer/Counter1 Compare Match A</td>
</tr>
<tr>
<td>19</td>
<td>$0024</td>
<td>TIMER1 COMP</td>
<td>Timer/Counter1 Compare Match B</td>
</tr>
<tr>
<td>20</td>
<td>$0026</td>
<td>TIMER1 OVF</td>
<td>Timer/Counter1 Overflow</td>
</tr>
<tr>
<td>21</td>
<td>$0028</td>
<td>TIMER0 COMP</td>
<td>Timer/Counter0 Compare Match A</td>
</tr>
<tr>
<td>22</td>
<td>$002A</td>
<td>TIMER0 COMP</td>
<td>Timer/Counter0 Compare Match A</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>23</td>
<td>$002C</td>
<td>TIMER0</td>
<td>COMPB</td>
</tr>
<tr>
<td>24</td>
<td>$002E</td>
<td>TIMER0</td>
<td>OVF</td>
</tr>
<tr>
<td>25</td>
<td>$0030</td>
<td>SPI, STC</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>$0032</td>
<td>USART0 RX</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>$0034</td>
<td>USART0 UDRE</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>$0036</td>
<td>USART0 TX</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>$0038</td>
<td>ANALOG</td>
<td>COMP</td>
</tr>
<tr>
<td>30</td>
<td>$003A</td>
<td>ADC</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>$003C</td>
<td>EE READY</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>$003E</td>
<td>TIMER3 CAPT</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>$0040</td>
<td>TIMER3 COMPA</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>$0042</td>
<td>TIMER3 COMPB</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>$0044</td>
<td>TIMER3 COMPC</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>$0046</td>
<td>TIMER3</td>
<td>OVF</td>
</tr>
<tr>
<td>37</td>
<td>$0048</td>
<td>USART1 RX</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>$004A</td>
<td>USART1 UDRE</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>$004C</td>
<td>USART1 TX</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>$004E</td>
<td>TWI</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>$0050</td>
<td>SPM READY</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>$0052</td>
<td>TIMER4 CAPT</td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>$0054</td>
<td>TIMER4 COMPA</td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>$0056</td>
<td>TIMER4 COMPB</td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>$0058</td>
<td>TIMER4 COMPC</td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>$005A</td>
<td>TIMER4 OVF</td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>$005C</td>
<td>TIMER5 CAPT</td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>$005E</td>
<td>TIMER5 COMPA</td>
<td></td>
</tr>
<tr>
<td>Address</td>
<td>Value</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
<td>----------------------</td>
<td></td>
</tr>
<tr>
<td>0060</td>
<td>COMPB</td>
<td>Timer/Counter5 Compare Match B</td>
<td></td>
</tr>
<tr>
<td>0062</td>
<td>COMPC</td>
<td>Timer/Counter5 Compare Match C</td>
<td></td>
</tr>
<tr>
<td>0064</td>
<td>OVF</td>
<td>Timer/Counter5 Overflow</td>
<td></td>
</tr>
<tr>
<td>0066</td>
<td>RX</td>
<td>USART2 Rx Complete</td>
<td></td>
</tr>
<tr>
<td>0068</td>
<td>UDRE</td>
<td>USART2 Data Register Empty</td>
<td></td>
</tr>
<tr>
<td>006A</td>
<td>TX</td>
<td>USART2 Tx Complete</td>
<td></td>
</tr>
<tr>
<td>006C</td>
<td>RX</td>
<td>USART3 Rx Complete</td>
<td></td>
</tr>
<tr>
<td>006E</td>
<td>UDRE</td>
<td>USART3 Data Register Empty</td>
<td></td>
</tr>
<tr>
<td>0070</td>
<td>TX</td>
<td>USART3 Tx Complete</td>
<td></td>
</tr>
</tbody>
</table>
NAME: Hussein M. Wehbe

PLACE OF BIRTH: Baalbeck, Lebanon

DATE OF BIRTH: 1st June 1987

EDUCATION: Al-Najah High School, Baalbeck, Lebanon

University Of Windsor, BS, Physics, Windsor, ON, 2012

University Of Windsor, MASC, Electrical Engineering, Windsor, ON, 2016