

The Control of a Model Washing Machine

Mbuotidem Ime Archibong

Bachelor's thesis
2019



Tomas Bata University in Zlín
Faculty of Applied Informatics

Tomas Bata University in Zlín
Faculty of Applied Informatics
Academic Year: 2018/2019

BACHELOR'S THESIS ASSIGNMENT

(PROJECT, ARTWORK, ARTISTIC PERFORMANCE)

Degree, First Name and Surname: **Mbuotidem Ime Archibong**
Personal Code: **A16657**
Degree Programme: **B3902 Engineering Informatics**
Degree Course: **Information and Control Technologies**

Thesis Topic: **The Control of a Model Washing Machine**
Thesis Topic in Czech: **Control of a Model of Washing Machine**

Thesis Guidelines:

- 1. Describe the equipment used for the course of microcontroller programming including the washing machine model.**
- 2. Design and implement the software for controlling the model of a washing machine connected to the development board.**
- 3. Prepare tasks for students focused on understanding and expanding the software.**
- 4. Create documentation for the software with description of the source code.**
- 5. Prepare teaching materials with instructions for students and solution for prepared tasks.**

Thesis Extent:

Appendices:

Form of Thesis Elaboration: **tištěná/elektronická**

Bibliography:

1. BARR, Michael a Anthony J MASSA. Programming embedded systems: with C and GNU development tools. 2nd ed. Sebastopol: O'Reilly, 2006, xxi, 301 s. ISBN 978-0-596-00983-0.
2. CATSOULIS, John. Designing embedded hardware. 2nd ed. Sebastopol, CA: O'Reilly, 2005, xvi, 377 p. ISBN 0596007558.
3. KOHOUT, Luděk. cz [online]. [cit. 2018-11-26]. Available from: <http://www.edumat.cz/produkty.php?produkt=edumod>
4. MCCONNELL, Steve. Code complete. 2nd ed. Redmond, Wash.: Microsoft Press, c2004. ISBN 978-0735619678.
5. MORTON, Todd D. Embedded microcontrollers. Upper Saddle River, N.J.: Prentice Hall, c2001. ISBN 0139075771.
6. SMITH, Warwick A. C programming for embedded microcontrollers. 2nd ed. Susteren: Elektor International Media BV, 2008. ISBN 978-090-5705-804.

Thesis Supervisor:

Ing. Jan Dolinay, Ph.D.

Department of Automation and Control Engineering

Date Assigned:

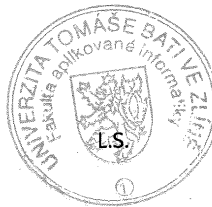
21 December 2018

Thesis Due:

15 May 2019

Zlín, 21 December 2018

doc. Mgr. Milan Adámek, Ph.D.
Dean



prof. Ing. Vladimír Vašek, CSc.
Head of Department

I hereby declare that:

- I understand that by submitting my Diploma thesis, I agree to the publication of my work according to Law No. 111/1998, Coll., On Universities and on changes and amendments to other acts (e.g. the Universities Act), as amended by subsequent legislation, without regard to the results of the defence of the thesis.
- I understand that my Diploma Thesis will be stored electronically in the university information system and be made available for on-site inspection, and that a copy of the Diploma/Thesis will be stored in the Reference Library of the Faculty of Applied Informatics, Tomas Bata University in Zlin, and that a copy shall be deposited with my Supervisor.
- I am aware of the fact that my Diploma Thesis is fully covered by Act No. 121/2000 Coll. On Copyright, and Rights Related to Copyright, as amended by some other laws (e.g. the Copyright Act), as amended by subsequent legislation; and especially, by §35, Para. 3.
- I understand that, according to §60, Para. 1 of the Copyright Act, TBU in Zlin has the right to conclude licensing agreements relating to the use of scholastic work within the full extent of §12, Para. 4, of the Copyright Act.
- I understand that, according to §60, Para. 2, and Para. 3, of the Copyright Act, I may use my work - Diploma Thesis, or grant a license for its use, only if permitted by the licensing agreement concluded between myself and Tomas Bata University in Zlin with a view to the fact that Tomas Bata University in Zlin must be compensated for any reasonable contribution to covering such expenses/costs as invested by them in the creation of the thesis (up until the full actual amount) shall also be a subject of this licensing agreement.
- I understand that, should the elaboration of the Diploma Thesis include the use of software provided by Tomas Bata University in Zlin or other such entities strictly for study and research purposes (i.e. only for non-commercial use), the results of my Diploma Thesis cannot be used for commercial purposes.
- I understand that, if the output of my Diploma Thesis is any software product(s), this/these shall equally be considered as part of the thesis, as well as any source codes, or files from which the project is composed. Not submitting any part of this/these component(s) may be a reason for the non-defence of my thesis.

I herewith declare that:

- I have worked on my thesis alone and duly cited any literature I have used. In the case of the publication of the results of my thesis, I shall be listed as co-author.
- That the submitted version of the thesis and its electronic version uploaded to IS/STAG are both identical.

In Zlin; dated:

.....
Student's Signature

ACKNOWLEDGEMENT

Throughout my study in Tomas Bata University in Zlin and in writing of this bachelor thesis, I have undoubtedly received tremendous support, motivation and guidance. Firstly, I would like to thank my supervisor Ing. Jan. Dolinay, PhD, whose expertise was extremely useful in formulating the topic of this project and the methodology in particular. I am sincerely grateful to him for sharing his truthful and illuminating views on a quite number of issues connected to this project.

Besides, my sincere thanks go to the dean of the Faculty of Applied Informatics, doc. Mgr. Milan Adamek, PhD, Head of Department of Automation and Control Engineering, Prof. Ing. Vladimir Vasek, CSc, thesis committee, including my tutors whom I have not mentioned their names, for their insightful comments and encouragement.

Last but not the least, I must express my profound gratitude to my family: My father Mr. Ime Archibong Akpan and my siblings for providing me with unfailing support and unremitting encouragement for the entire years of my study and throughout the process of researching and writing of this bachelor thesis.

ABSTRACT

The aim of the thesis is to create a program for controlling model of a washing machine using EDU-mod hardware connected to the development setup board.

The KL25 microcontroller has been used in this project to obtain data from the sensor and forward it to the PC via USB 2.0 that has been integrated into the microcontroller. The KL25 microcontroller is selected for this project due to the compatibility with the EDU-mod hardware.

As the thesis focuses more on the designing software than hardware, the development setup board is used which has all the hardware components needed for this project.

The Kinetis Design Studio IDE has been used to build the software for this model and load the program to the KL25 microcontroller. The Kinetis Design Studio supports C language which is flexible and makes debugging and testing to be less complicated.

Lastly, the work carried out on this project shows the example project for controlling the model washing machine and the documentation of this project helps broaden the students understanding regarding Microcomputer Programming course.

Keywords: Microcontrollers, EDU-mod, Process Control, FRDM – KL25Z

CONTENTS

1.	INTRODUCTION.....	7
1.1	Introduction	7
1.2	Project Scope	7
1.3	Project Objectives	7
1.4	Layout of the Thesis	8
2.	LITERATURE REVIEW	10
2.1	General Overview of Microcontroller	10
2.1.1	Main Parts of Microcontroller	11
2.1.2	Input and Output Module	11
2.1.3	Timers	14
2.2	Kinetis Design Studio – IDE Overview	15
2.3	Kinetis Design Studio IDE Key Features	16
2.4	General Debug Solutions	22
3.	BASIC PART.....	24
3.1	KL25 Microcontroller Unit	24
3.2	General Description of KL25	24
3.3	Analog to Digital Converter	25
3.4	Development Board	26
3.5	Edu-Mod Hardware	27
4.	ANALYSIS OF THE PROJECT	30
4.1	Introduction	30
4.2	Teaching Material for The Course Subject	30
4.3	Software Implementation	31
4.4	Practical Tasks for Students	38
4.5	Theoretical Tasks and Solutions for Students	39
5.	CONCLUSIONS	41
	BIBLIOGRAPHY	42
	LIST OF ABBREVIATIONS	43
	LIST OF FIGURES	45
	LIST OF TABLES	46

1. INTRODUCTION

1.1 Introduction

The rapid growth in science and information technology has brought several advantages of using microcontrollers, microprocessors, integrated circuits, embedded chips to mention but few, in designing virtual home assistants, smart systems, robotics, industrial applications, simulation and optimization, automobiles and others. The key thing is that these systems make use of one or more microcontrollers to obtain information from sensor systems, to process this information, and to control actuators based on this information.

Furthermore, in order to design these microcontroller-based systems, it is imperative for student to have strong theoretical knowledge regarding embedded systems and practical skills. Microcomputer Programming subject is one of the mandatory subjects offered for Information and Control Technology Engineering undergraduate program in Tomas Bata University. Typically, this course program is taught primarily focusing on software and the hardware architecture.

Nevertheless, development in semiconductor electronics have changed the way firm resolves production and control process problems. The advancement and the increased use of microcontroller in firms have led to new trends in microcontroller in higher institutions.[1][2][10]

1.2 Project Scope

This thesis entails the following:

- Understanding microcontroller's design and functionality.
- To familiarize with the techniques needed in designing embedded software for microcontrollers.
- Understanding the KL25 MCU components.
- Developing a microcontroller software using C language.

1.3 Project Objectives

The overall aim of this thesis is to build and test a program for controlling of a model washing machine using KL25 microcontroller, EDU-mod and Kinetis Design Studio IDE.

Below are the main objectives of this project:

- Describe the equipment used for the course of microcontroller programming including the washing machine model.

- Design and implement the software for controlling the model of a washing machine connected to the development board.
- Prepare tasks for students focused on understanding and expanding the software .
- Create documentation for the software with description of the source code.
- Prepare teaching materials with instructions for students and solution for prepared tasks.

1.4 Layout of the Thesis

Besides, this thesis is a document that presents the ideas generated and the concept implemented. Furthermore, this thesis comprises five chapters namely; Introduction, Literature Review, Basic Part, Analysis of the Project and Conclusions.

Chapter one describes the overview of the research spot, motivation driving the work, scope of the thesis, pinpointed objectives and the arrangement of the thesis.

Chapter two gives comprehensive study of the literature about this project. It further explains the research carried out and data obtained from journals, internet, books etc.

Chapter three clearly explains the hardware components used and the methodology applied in this project. It lays emphasis on the key features of the hardware components used in this project and detailed idea about these components associated with the microcontroller.

Chapter four of this thesis explicitly discusses the output of this project as the product of the applied methodology.

Chapter five shows the conclusion and discussion about the whole project findings.

I. THEORY

2. LITERATURE REVIEW

2.1 General Overview of Microcontroller

A microcontroller is a solitary chip microcomputer formed from VLSI fabrication. It is an electronic device which belong to the microcomputer family. A microcontroller can be referred to as a computer in a single integrated circuit, capable of performing one task and executing specific application. Currently, there are microcontrollers with different word length starting from 4-bit, 8-bit, 16-bit, 32-bit, 64-bit to 128-bit.

In a broader sense, microcontroller basically consists of one or more components namely, memory (RAM, ROM), integrated processor core, input and output peripherals, serial interface module, interrupt controls etc. Microcontrollers are integrated in devices which allow user to exert a certain degree of control. They are designed to implement a specific function such as displaying of a character or string on an LCD display module. Figure 1.1 below shows the architecture of a microcontroller.[3][4]

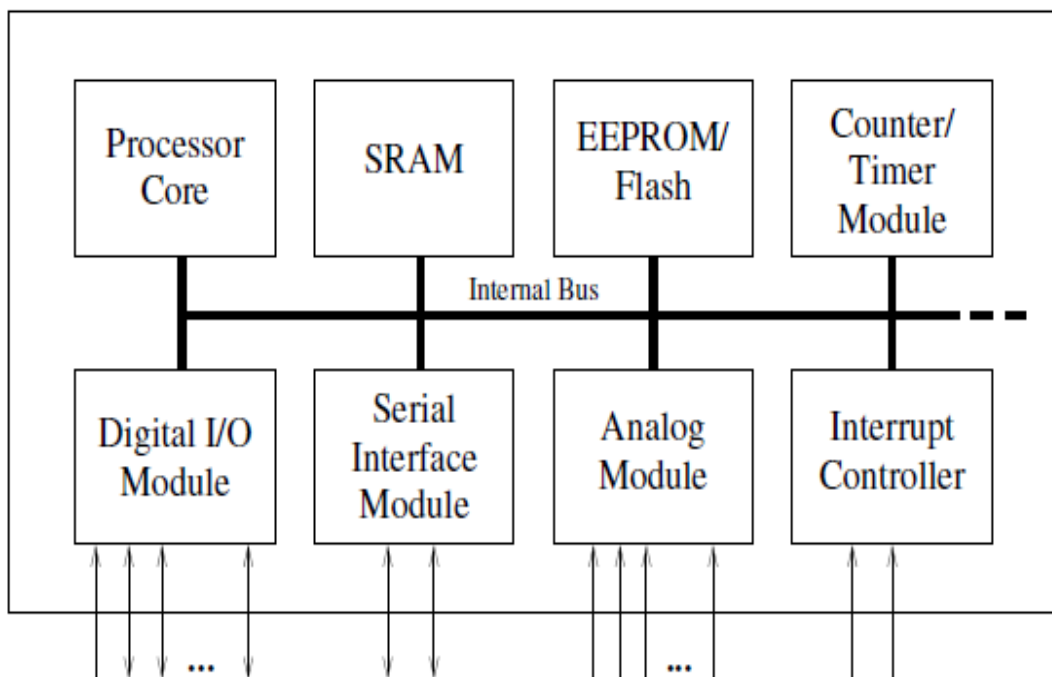


Figure 2.1: Microcontroller Structure.

2.1.1 Main Parts of Microcontroller

Microcontrollers are classified according to the following:

Buses

A bus is simply a pathway that allow digital signals to rapidly transfer data. It serves as a communication channel for the data and control signals that are moving between the crucial components of the computer systems. These buses are made up of 8, 16 or more wires of the microcontroller. Consequently, these can carry 8-bits and 16-bits simultaneously.

Basically, there are three main internal buses associated with processors namely, the data bus, address bus and control bus. Jointly, these three internal buses make up the system bus and each of them has its own different capabilities and functions. The system bus is an internal bus, designed to link the processor with internal hardware devices.[3]

The system bus combines the functions of the three main buses, which are stated as follows:

- The control bus consists of the control, timing and cordination signals to manage different functions across the system.
- The address bus is used to specify memory positions for the data signal being transfered (it specifies where data should be sent to). Also, the width of the address bus regulates the amount of memory a system can address.
- The data bus is a bi-directional channel that carries the actual data between the processor, the memory and the peripherals. The width of the data bus reflects the maximum amount of data that can be executed and delivered at a time.

2.1.2 Input and Output Module

Input and output module or to be more general, the capacity to directly check and control hardware, is the primary characteristic of microcontrollers. Most cases, microcontroller is applied in embedded systems to control the operation of machines in the microcontroller. Accordingly, to connect it to other devices or peripherals microcontroller interface requires input and output interfacing ports. Input and output ports are used to interface numerous peripherals such as printers, external memories, LEDs, and LCDs to the microcontroller.

Furthermore, another important feature of input and output pins is maximum current they can source. Majority of the microcontroller obtain current via pin which is sufficient to trigger low current devices ranging from 10 – 20mA. A microcontroller that has multiple input and output pins tends to have lower current in one pin. Basically, each of the input and

output port is controlled by a special function register (SFR), meaning that the bit of that register evaluates the condition of the corresponding pin. For instance, by putting logic one to one bit for control register SFR, the appropriate port pin is automatically set as input. It implies that voltage sent to that pin is read as logic 1 or 0. Otherwise, by putting zero to the register SFR, the appropriate port pin is set as output. Input and output voltage (0V or 5V) equal to the state of the suitable bit of the port register.

Another primary feature of the pin is to have pull-up resistor. The resistor links the pin to power supply and the result is visible when the pin is set as input attached to automated switches or push buttons.[3]

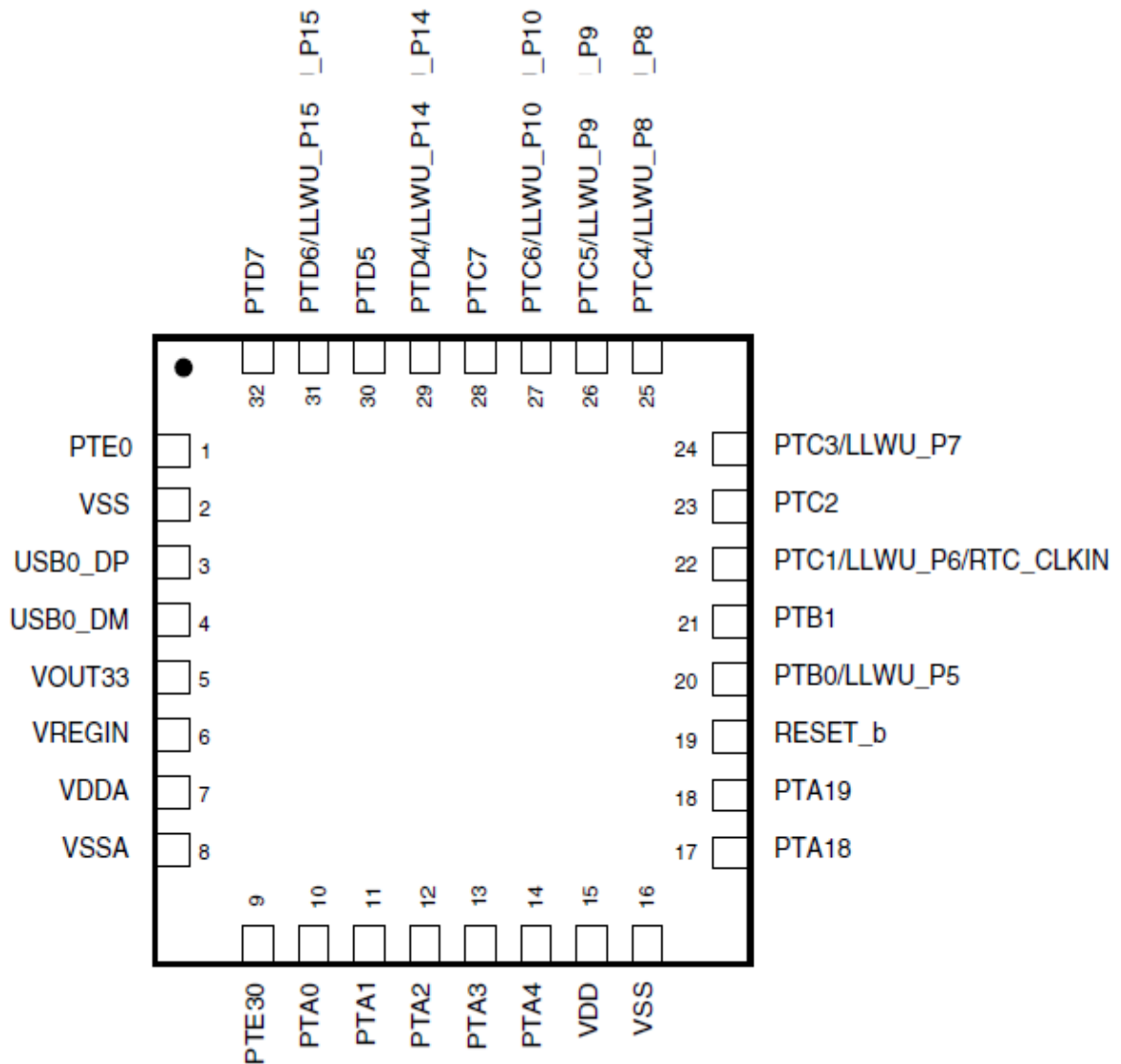


Figure 2.2: KL25 32-pin Pinout Diagram.

Interrupts

Microcontrollers are deployed in systems that react to events. Events in this context means changes in the controlled systems and normally require some sort of reaction by microcontroller. In real-time process, to handle certain conditions satisfactorily, the actual task must be suspended for a certain period of time – it takes necessary action and then later return to the main task. Interrupts are necessary in executing such programs. Interrupt refers to an event that halts the main program for a limited period of time, pushes the control to a special section of code, implements the event-related function and resumes where it had left off in the main program. The function of interrupt is very useful as it helps in instance of emergency operations.

To summarize, indications for applying interrupts are:

- I. Event occurs infrequently.
- II. Long intervals between two or more events.
- III. Event is also caused by HW, no bouncing effects or spikes.

Microprocessor

A microprocessor is built to implement arithmetic and logic operations that make use of small number-holding areas known as registers. Typically, microprocessor operations consist of subtracting, adding, comparing two values and fetching values from one area to another.

Besides, a microprocessor can be referred to as a digital device that operates in binary numbers 0 and 1, which has a computing and decision capabilities equivalent to central processing unit of a computer. It executes the instruction based on a stored program and output the information in digital form. Every microprocessor has a defined set of instructions in the binary form which is referred to as a machine learning.

A microprocessor is connected with I/O devices, memory. Consequently, combination of these component forms a microcomputer as shown below in the block diagram of figure.

2.2. [3][2]

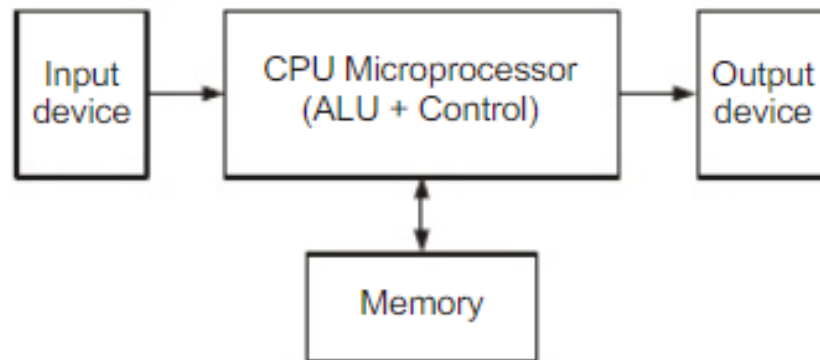


Figure 2.3: Block diagram of a Microprocessor.

The microprocessor is made up of arithmetic and logic unit (ALU) and control unit. The ALU performs arithmetical and logical operations on data sent from the memory or input device.

Arithmetic and logic unit (ALU)

ALU is the unit of microprocessor where numerous computing operations are performed on the data. These functions are arithmetic operations such as additions, subtraction and logical operations – AND, OR etc. Results are saved either in registers or in memory or forwarded to output devices. ALU consists of the following; accumulator, flag register and temporary register.

Control unit

Control unit provides necessary timing and control signal required in the operation of the microcomputer. It help generates signals within microprocessor to implement the instructions, which has been decypted. With the help of control unit, flow of data signal within the peripherals are controlled (input, output and memory). Control unit itself consists of the following: namely, the instruction register (IR), instruction decoder and machine cycle decoder and timing contol unit.

Registers

Register stores the machine code of the intruction(such as a bit sequence). Typically, they are used mainly to store data temporarily during the execution of a program. They are usually at the top of the memory hierarchy and they make provision of accessing data quickly. More so, during execution of a program in the microprocessor, it reads the opcode from the memory, this opcode is stored in the intruction register.

2.1.3 Timers

This feature is one of the important constituent parts of a microcontroller. Timers are used to carry out timing operations concurrently with the program, by ensuring that the program is faster and more efficient.

Additionally, a microcontroller may have multiple timer and counter. With the help of timers and counters, it is possible, for instance to control the brightness of LEDs, control the angle of servo shafts, receive sensors data transmit in PWM etc. Most microcontrollers have clocks embedded in them or they use an external clock. More so, they require real-time clock so that the program can be processed in rhythm with the clock especially with processes that are dependent on time.

Counter

Each timer is typically a counter which can be either incremented or decremented upon every clock tick. The up and down counter direction is either fixed or configurable. The current count set can be read via a count register and can be set to a specific value by the user. Most importantly, care must be taken when the timer length exceeds the word length of the controller for instance when using a 16-bit, count value must be done in two levels which could lead to irregular values.

Timers usually raise an interrupt whenever they encounter an overflow of the count value. This can be applied during implementing a rudimentary periodic signal by positioning the count value to a given start and then halting for the overflow. However, this technique does not give exact period, the reason is that the timer has to be set to its starting value by the program after the overflow. Consequently, the time from the overflow until the start value is reloaded into the timer must either be incorporated into the main start value, or the time will be extended than desired.

In order to avoid this pitfall, some timer provides a modulus mode which automatically refills the start value once the timer overflows. Another technique for getting exact periods is to use PWM characteristic. Timers and counters can be applied in operations which include modulation, frequency generation, making oscillations, clock functions and pulse generations. [2][3][7]

2.2 Kinetis Design Studio – IDE Overview

This section seeks to discuss about the general overview of Kinetis Design Studio, its key features with microcontrollers and benefits of IDE in building this project.

Firstly, Kinetis Design Studio IDE is a software application that consolidates vital tools needed to write and test programs. Basically, IDE is also an application that aids software development. It provides a programming platform to streamline developing and debugging software application. [5][11][12]

Thus, it offers a central easy-to-use interface, capability to run and debug programs from one screen, including the following:

- **Code Editor:**

This attribute is a text editor built for writing and editing source code. The source code editors is different from text editors because they simplify writing and editing code.

- **Compiler:**

It is a computer program that transforms source code from human readable or writeable language into an executable form by a computer(object code). The primary reason for wanting to transform source code is to build a program that can be executed on a computer system or embedded system.

- **Debugger:**

This feature is a computer program that is used during testing to help debug other programs. It offers more sophisticated operations for example executing a program step by step, breaking at some event by means of breakpoint(stopping the program to check the current state). It also has the capability to revise the program while it is running.

2.3 Kinetis Design Studio IDE Features

This section describes the features of Kinetis Design Studio IDE and the steps in the building and creating of this project.

Running Kinetis Design Studio

By default, when KDS is launched, user is prompted to select the workspace and it is enabled when KDS is installed for the first time. See figure 2.3 below.

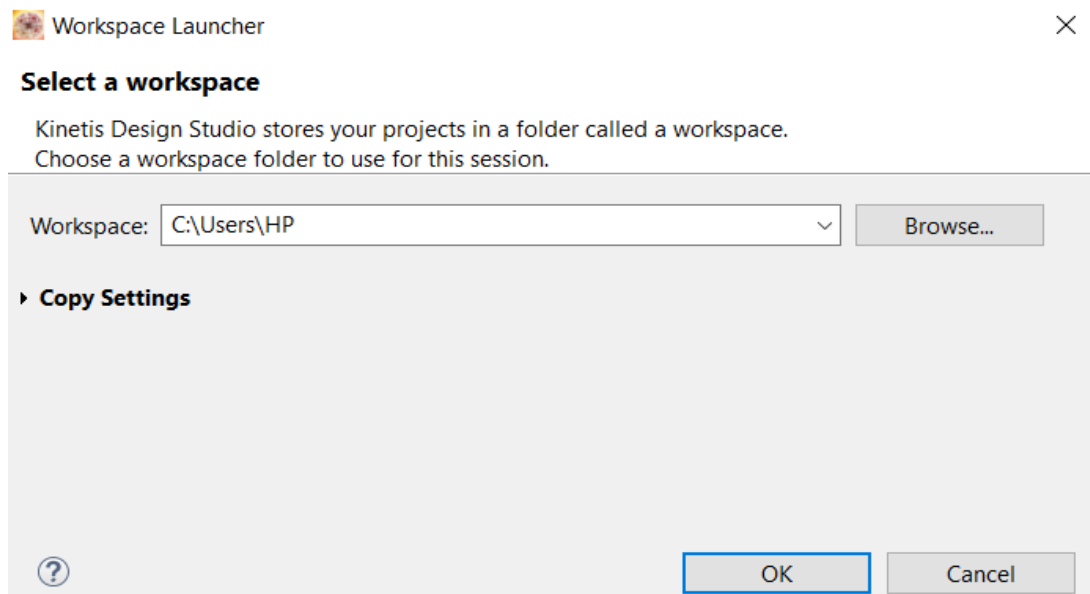


Figure 2.4: Workspace Dialog Box.

If KDS is launched and the workspace launcher dialog box is not prompted, user has the option to switch the workspace using the following steps, File > Switch Workspace.[5]

Running the Program (uploading to the microcontroller)

Firstly, before starting program on KDS, the debug interface needs to be properly set and configured. In order to set the debug interface, the following steps are used: select the project and right click it. From the drop-down menu select Debug As and finally Debug Configurations. See figure 2.4.[5]

From the Debug Configuration window, expand PEMicro Interface Debugging and select the project. Once the debug panel is opened, click on the Debugger tab from the right. The following options are selected: see figure 2.5.

- Interface: OpenSDA
- Port: Once the evaluation board is connected, it will be indicated in this port.

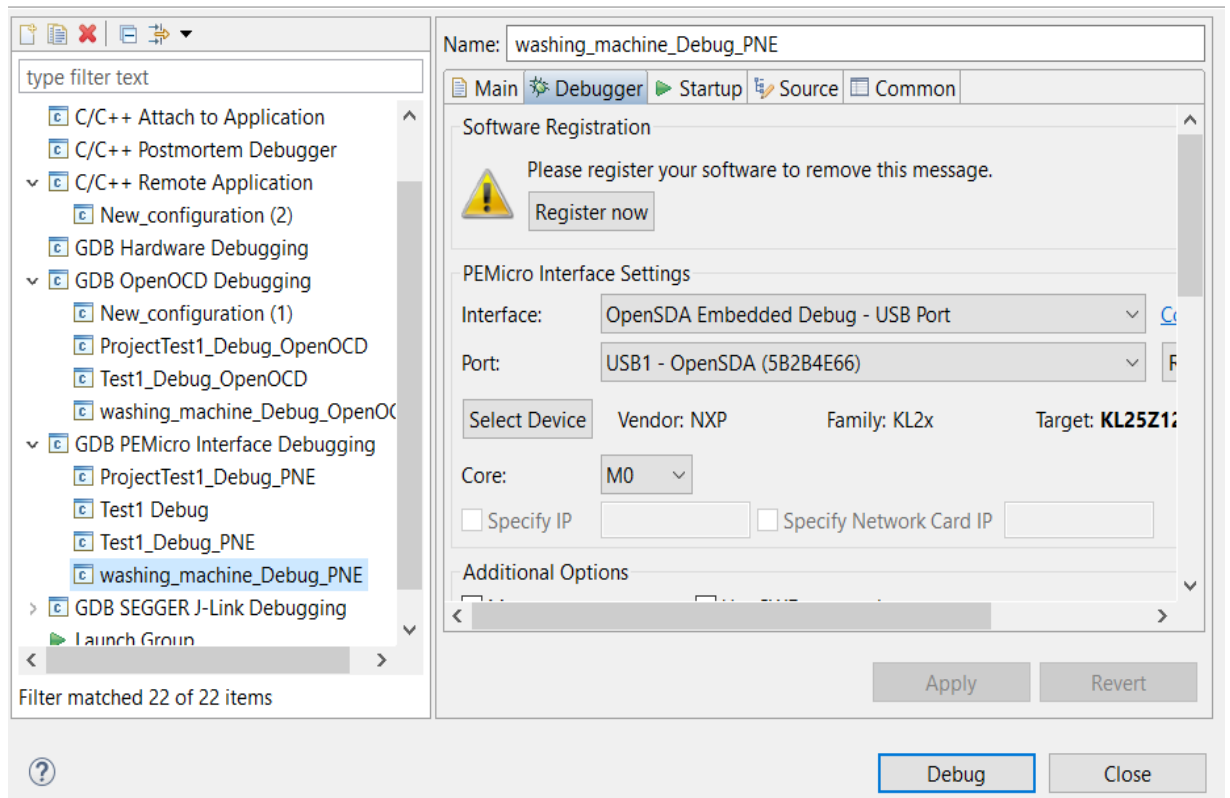


Figure 2.5: Debugging Interface.

Besides, by clicking the Debug button in the lower right corner of the window, the program starts the upload and debugging.

When the program starts, it is possible to expand the menu in the Debug button in toolbar and select your debug configuration. See figure 2.6.

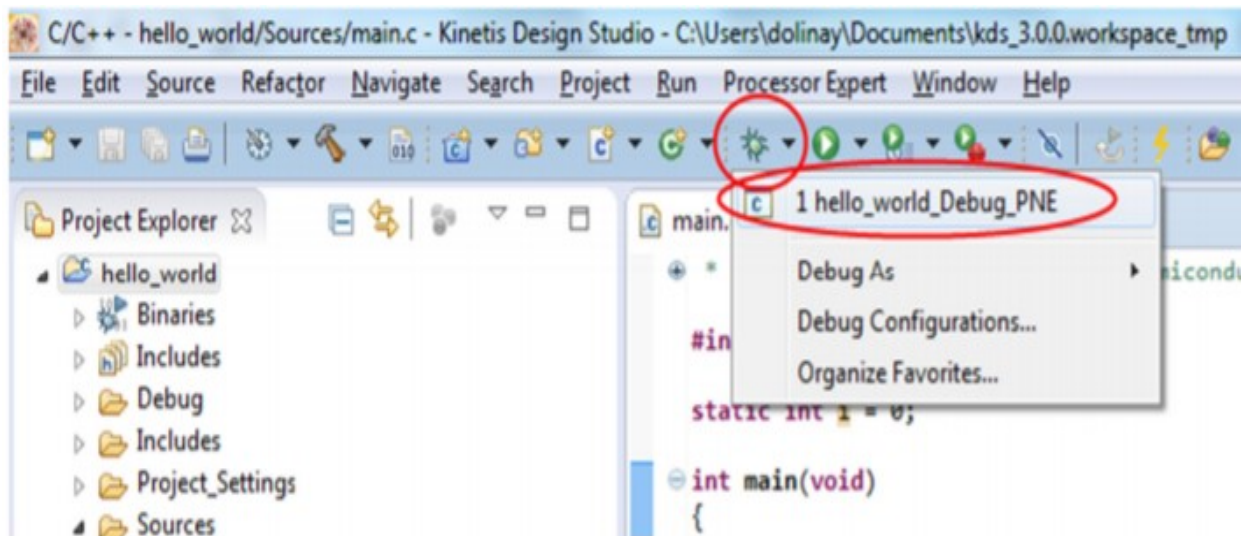


Figure 2.6: Selecting a Program to Debug.

In most cases, KDS prompts user to switch perspective by selecting Yes or No in order to rearrange the program. KDS windows usually rearrange the program to make it convenient for debugging. Example of the prompted window is shown below in figure 2.7.

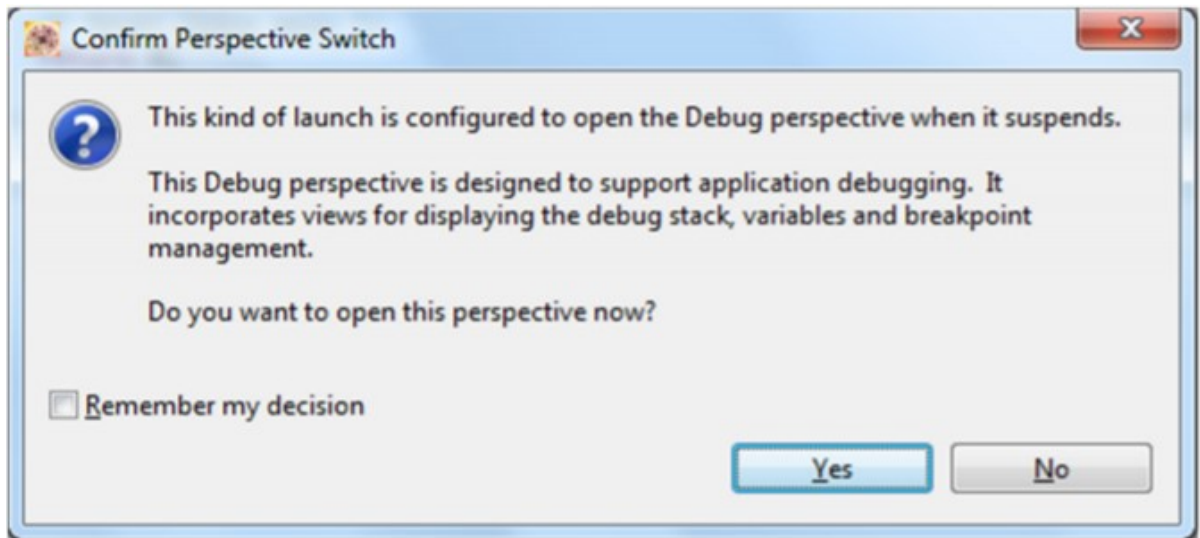


Figure 2.7: Debug Perspective Switch Dialog Box.

On top of that, once debugging is finished, user can easily switch back to perspective for writing code using C/C++ button on the top right corner of the KDS as shown below in figure 2.8.

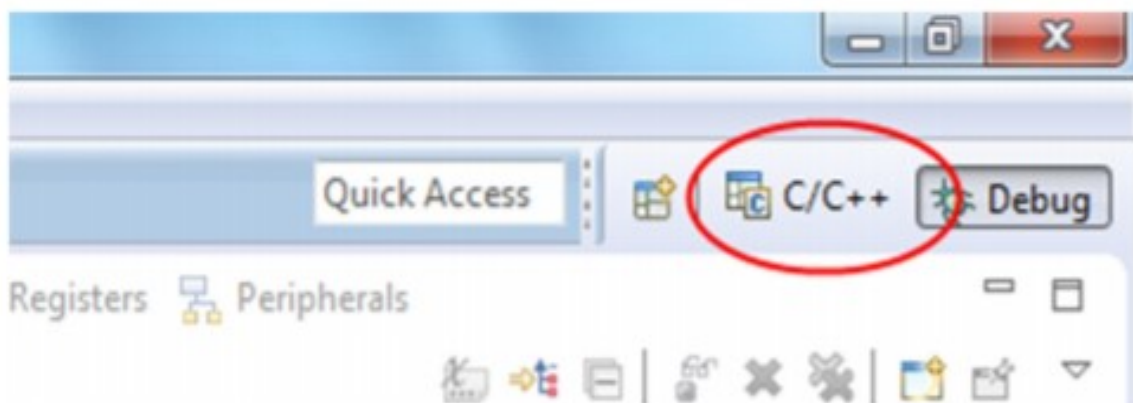


Figure 2.8: C/C++ Button Interface.

Adding Files to Project

New file can be added to project by selecting the following options: New > Source File.

Also, existing file (driver code) can be added using the following steps:

- The file will be located in the project
- There will be a link to the file in the project.(located in the original position).

Alternatively, another way to add the file to the project is to drag and drop it to the Sources folder in the KDS from any file manager software such as windows explorer.

A pop-up window will appear, which will allow users to select where to copy the files or create a link as shown below in the figure 2.9.

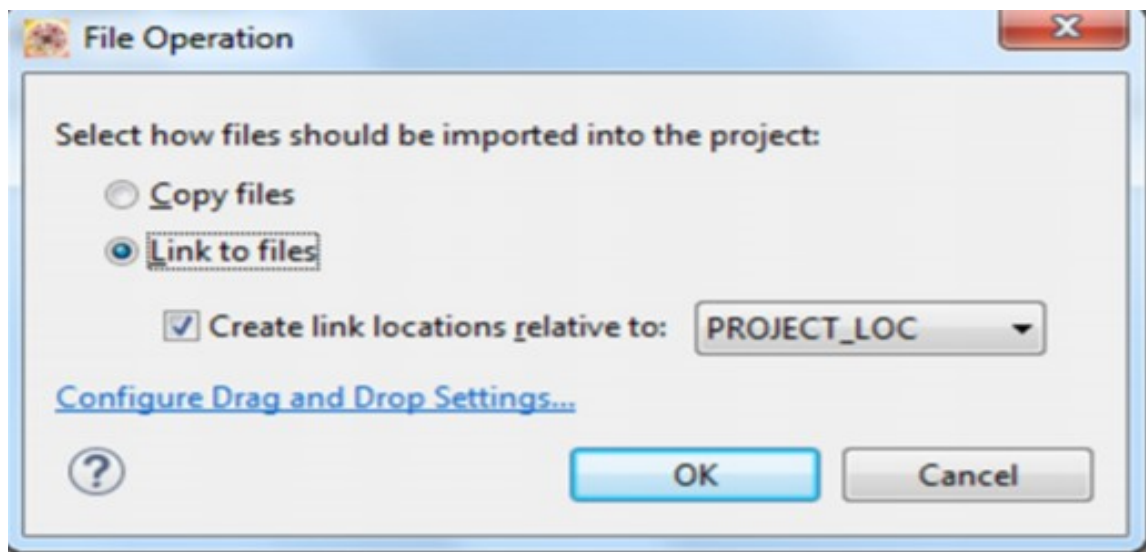


Figure 2.9: Confirm Import Option Dialog Box.

Adding Drivers to Project

This section explains how the peripheral drivers can be added to the program. Normally such drivers are stored in one source file such as drv_adc.c and one header file e.g. drv_adh.h. Adding drivers, the following steps are performed:

- Add the driver source and header files (.h .a .c) into the projects. This can be done by drag and drop.
- Select Link to files option.
- Add the path to the folder where the driver is located to project settings in Properties>C/C++ Build>Settings>Cross ARM C Compiler>Includes

In order to add the path to header files of a peripheral driver into compiler settings, the following steps are taken:

- Open project Preferences.
- Select C/C++ Build> Settings > Cross ARM C Compiler > Includes.
- In the Include Paths list click the Add button in upper right.
- In the Add directory path window click File System button.
- Browse the location of the driver and click Ok.

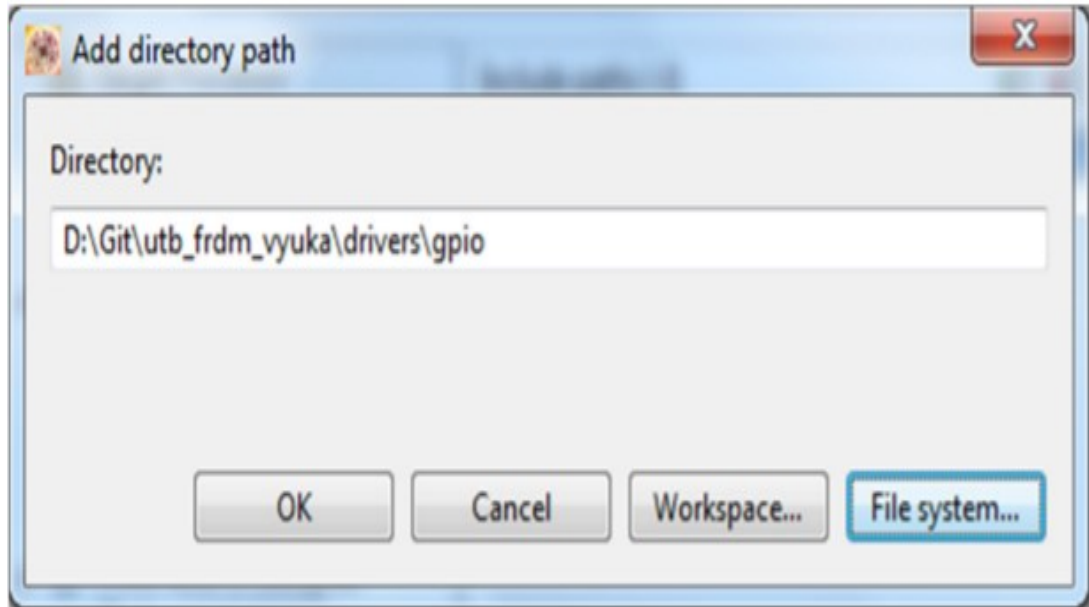


Figure 2.10: Select Directory Path.

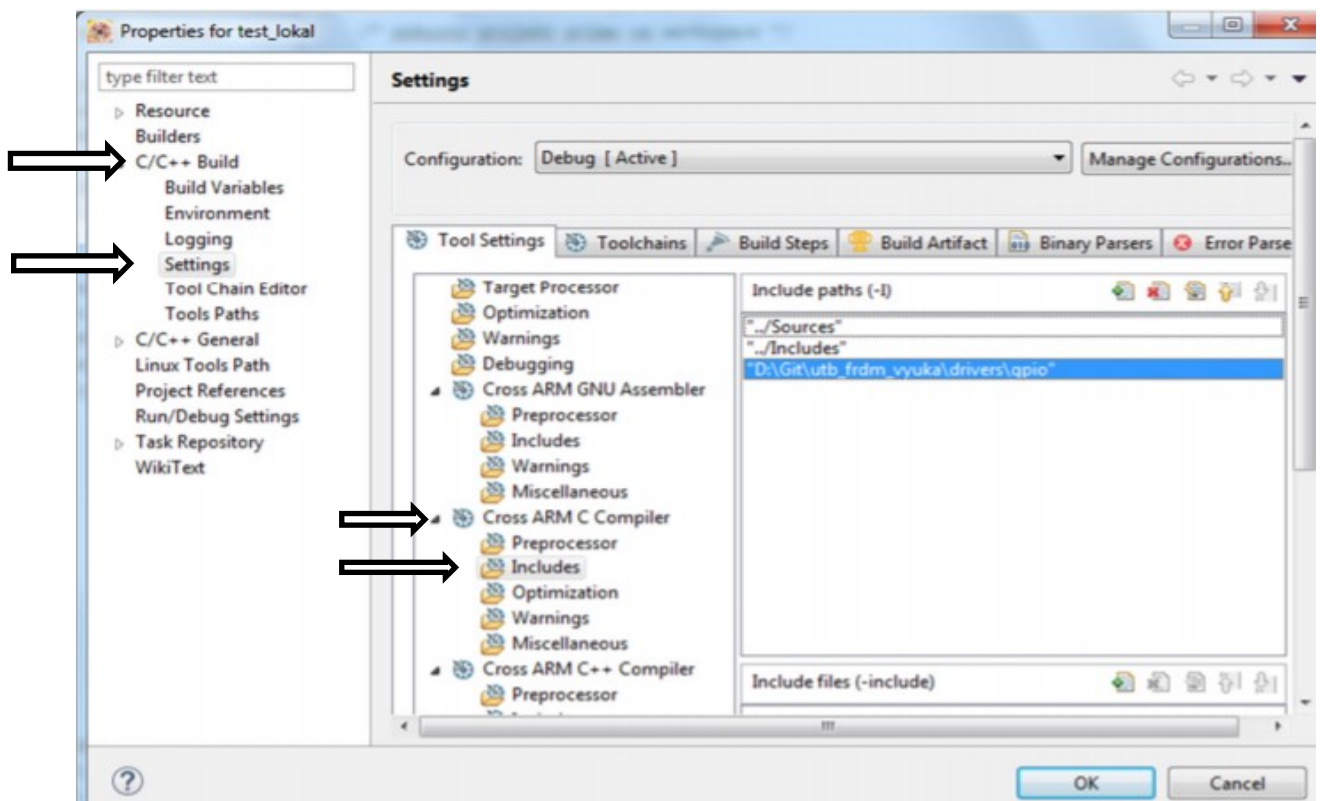


Figure 2.11: Selecting Drivers to Project.

2.4 General Debug Solutions

This section describes the use of Kinetis Design Studio in debugging in this project. It further explains the usefulness of debugger in identifying invalid data in Software.




More so, a debugger is a software tool that can be used in testing and debugging other applications in order to remove possible bugs. The components involved in this tool, make use of algorithm and data structures to execute their objectives. In most cases, debuggers are mostly used to analyze and identify invalid data in a program that does not run properly as expected. They are very useful in helping to identify the root cause of a software glitch.








Adjusting the application in the debugger

Once the breakpoint on main is reached, you be allowed to step through the lines of code, double click from the left side of the code statement to set a breakpoint, pause execution, resume execution, restart and terminate the debug session.

The table below describes the major buttons which are used to control the execution of program in the debugger.[5]

Table 2.1: Buttons and Features in KDS.

S/N	Button	Action	Description
1.		Resume	This feature allows the suspended application or program to resume or continue.
2.		Suspend	Suspend feature allows the execution of the program to be suspended.
3.		Instruction stepping mode	The debugger switches to the instruction stepping mode automatically when the disassembly view has focus.

4.		Step over	This feature helps execute the current line, after execution inside a routine.
5.		Step into	This feature helps execute the current line, with any routines and proceed to the next statement.
6.		Step return	This feature permits the execution to the end of the current routine, the follow execution to the routine's caller.
7.		Restart	This feature simply restart the selected debug session.
8.		Disconnect	This feature helps detach the debugger from the selected debug session.
9.		Terminate	When this feature is selected, it terminates the debug session.
10.		Debug	This feature allows user to select from the drop down list the debug configuration for debugging.

Debugging Program Using KDS

Once the program is loaded into the microcontroller unit, it will be stopped at the beginning of the main function. The following function can be performed:

- Run the program immediately using the Resume button. See the table above.
- Execute the program in single steps (lines of code) using Step Into or Step Over buttons.
- Allow the program to run to a particular line, in which you select using Run to line command after right-click on the program line.
- Set a breakpoint at the position where the program should stop. Right-click the left margin next to the desired line and select Toggle Breakpoint from the context menu. In order to remove it, you can easily use the same command.

3. BASIC PART

This chapter explains the key features of the hardware components used in the control of a model washing machine project.

3.1 KL25 Microcontroller Unit

The FRDM-KL25Z was chosen for this project. It has 32-bit ARM Cortex-MO+ core running at a speed of 48MHZ, based on the Freescale KL25. It includes the following: 128kb Flash, 16kb RAM and interfaces such as USB host, USB device, SPI, ADC, DAC, PWM, touch sensor and other input/output interfaces. The figure below shows the KL25Z microcontroller used in this project.[6]

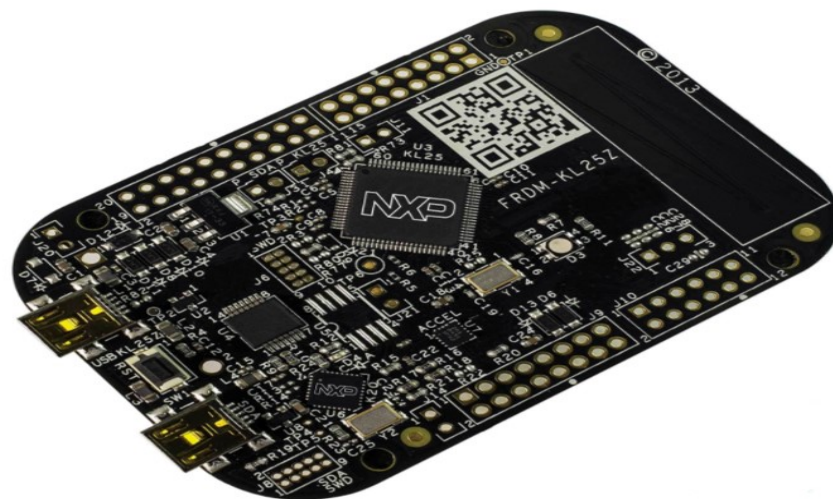


Figure 3.1: The FRDM-KL25Z Board.

(source: freescale.com)

3.2 General Description of KL25

This section explains the technical specification and features of the KL25 microcontroller unit used in this project. [6]

The KL25Z microcontroller include the following features:

1. Core and architecture:

- a. The density is better than 8-bit and 16-bit MCUs. It reduces flash size, system cost and power consumption.
 - b. It has up to 4-channel DMA for peripheral and memory servicing with minimal CPU intervention.
2. Ultra low-power:
 - a. It has a multiple low-power modes, including new operation clocking option which help reduce dynamic power consumption shutting off bus and system clocks for lower power core processing.
 3. Timing and control:
 - a. It has powerful timer modules which support general purpose, PWM, and motor control functions.
 - b. There is a Periodic Interrupt Timer for RTOS task scheduler time base
 4. System:
 - a. It has GPIO with pin interrupt functionality.
 - b. It also has wide operating voltage range from 1.71V to 3.6V with flash programmable down to 1.71 V with fully functional flash and analog peripherals.

3.3 Analog-to-Digital Converter (ADC)

Features of the ADC module includes the following:

- Linear successive approximation algorithm with up to 16-bit resolution.
- It has up to four pairs of differential and 24 single-ended external analog inputs.
- Output modes:
 - Differential 16-bit, 13-bit, 11-bit and 9-bit modes.
 - Single-ended 16-bit, 12-bit, 10-bit and 8-bit modes.
- Output in right-justified unsigned format for single-ended.
- Single or continuous conversion, which is automatic return to idle after single conversion.
- Configurable sample time and conversion speed or power.
- Conversion complete or hardware average complete flag and interrupt.
- Input clock selectable from up to four sources.
- It has operation in Low-Power modes for lower noise.
- Asynchronous clock source for lower noise operation with option to output clock.
- Temperature sensor.
- Self-Calibration mode.

Block diagram

The following figure depicts the ADC module block diagram. See figure 3.2.

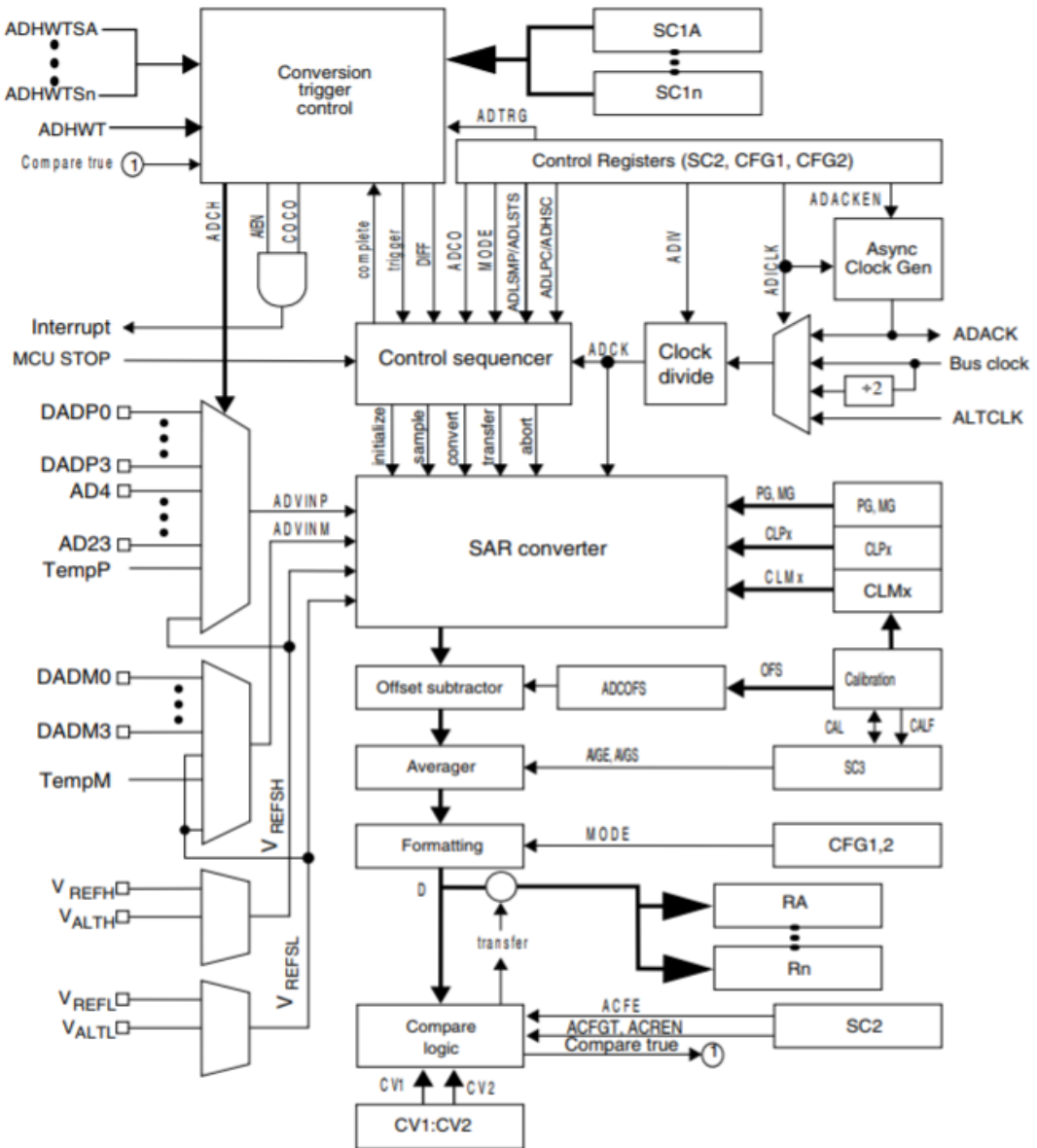


Figure 3.2: ADC Block Diagram of KL25 MCU.

3.4 Development Board

To get started with this project, we need two major things which are development board and an IDE (Integrated Development Environment). A microcontroller development board is simply a printed circuit board (PCB) which has circuitry and hardware designed to ease experimentation with a certain microcontroller boards feature.

Generally, development boards are integrated with a processor, chipset, memory including peripherals like USB, serial port, ADC, motor driver, SD card slot, ethernet, etc. with debugging attributes. The specifications of microcontroller boards are usually made up of bus type, memory, number of ports, port type and operating system.

The KL25 was the development board that was selected for this project and it is shown in figure 3.1 above.

3.5 EDU-Mod Hardware

This section explains the function of the EDU-mod hardware used in this project.

Firstly, EDU-mod is an education-oriented hardware device which is connected to the Development Board. The major function of EDU-mod hardware is that it has the designed features and mechanism of a model washing machine. On top of that, within the interface it has output notifications and the button which can be controlled during and after the program is executed. [8]

The figure below shows the EDU-mod hardware device which is used in this project. See figure 3.3.



Figure 3.3: Edu-Mod Hardware.

Model Features of EDU-mod Hardware

EDU-mod hardware used in this project, has the following features:

- EDU-mod hardware has internal processor unit that regulates LEDs simulating water level, temperature sensors and error notifications.
- Two main features are used in the rotation of the drum. The rotation of the drum is displayed by eight circular LEDs arranged in a circular pattern.
- The water level is scanned at two levels, 50% and 100% and it is displayed on the LED.
- When heating the water, the time constants are shortened so that it does not have to wait for a long time before it tunes the water. Typically, it is designed to take up to 60s to heat up to 90⁰ C. On the EDU-mod hardware, the temperature ranges from 30⁰ C, 40⁰C, 60⁰C to 90⁰ C.
- Initialization state: when the power is switched on or after a restart (RESET button), the program automatically sets to initialization. During this process, it empties the drum and starts reading the temperature.
- EDU-mod hardware has error notifications displayed on the interface. There are two classes of error that typically shows on the device:
 - **Fixable Error:**
 - This type of error occurs if the program to activate the rotation of the washing machine's drum in clockwise and anticlockwise direction are executed at the same time instead of one at a time.
 - The ERR LED starts flashing and the drum stops rotating. Once the error is removed, the drum operates and runs normally.
 - **Non-fixable Error:**
 - This occurs if the temperature is above 90⁰ C.
 - In order to clear this kind of error in the program is by pressing the RESET button on the EDU-mod hardware.

II. ANALYSIS

4. ANALYSIS OF THE PROJECT

4.1 Introduction

This chapter explains the detail design and the implementation of the “Control of a Model of Washing Machine”. Furthermore, emphasis is laid on the teaching material of the course subject used, including explanation of the software implementation.

Besides, in Microcomputer Programming course, it is suitable to use the washing machine as a case study. Students are familiar with this type of object and at the same time it is relatively uncomplicated to control or simulate. In this project, EDU-mod hardware is used which is connected to the development board to display the output of the “Control of a Model Washing Machine”. More information about EDU-mod hardware is explained in the previous chapter. [8] See the figure 3.3.

4.2 Teaching Material for the Course Subject

The microcontroller course is one of the crucial courses for students of Informatics. The criteria for this course are Computer Programming (with C and C++), Computer Hardware and Architecture, Programmable Controllers etc. The microcontroller course covers the basics and the principles of microcontrollers including emphasis on laboratory exercises, logic, programming with C language and software design.

More so, the microcontroller course offers to engineering students the preparation and training needed to practically use an integrated development board for building their software programs in assembly and C language.[2][11][13] The following is a list of topics which is covered by this course subject:

- Introduction to Microcontroller Architecture and Embedded Systems.
- Microcontroller Operations.
- Representation of Data and Memory.
- Introduction to Assembly and C Programming.
- Memory Models.
- KL25 Family MCU Review.
- Analog Input and Output Interfacing (GPIO).
- Basics and Principles of Serial Port Communication.
- Arithmetic Instructions and Logic.
- Basics of Pulse Width Modulation (PWM).
- Pin Description, Drivers and Understanding of Hex File.

- Programming External Hardware and Serial Communication Interrupt.
- Sample Projects with Microcontrollers (e.g. bicycle light, mixing unit).

Course Materials

Course materials (learning materials) in teaching Microcomputer Programming subject are essential to the better understanding of the subject and success of student attainment. In other words, the instructional module of seminar planning in teaching this course is dependent on the choosing of the course materials. Furthermore, it describes the resources and contents the teacher uses in delivering instruction. The course materials for microcontroller subject come in various dimensions, but they all have unique capability to support student theoretically and practically in understanding this course.

In order to effectively fulfill the course objectives which, consist of the theoretical and practical aspect including exercises, this course material should be arranged as the following:

- There should be availability of resources for Microcomputer Programming course which contain microcontrollers, embedded systems, development board, design examples etc. Such resources should be easily accessible and valid.
- Resources should include: Laboratory manuals, textbooks, etc.
- Microcontroller kits chose for this course should be easily accessible and less expensive for students.
- The IDE selected for this course should be efficient and robust. E.g MCUXpresso IDE which is Eclipse-based development environment and equally support C/C++ libraries.
- There should be provision for less expensive development board for Microcomputer Programming, so that student can equally make use of it in their home.

4.3 Software Implementation

This section seeks to explain step by step process used in the implementation and operation of the control of a model washing machine, including the output using EDU-mod hardware. The control of a model washing machine design entailed the following:

- Flow chart of the program.
- Hardware architecture.
- Software components used in this project.
- Building and running the program.

Flow Chart of the Program

Program for the control of a model washing machine is written in C language using Kinetis Design Studio (KDS) IDE. The flow chart shown below, represents the workflow used in the implementation of the project. See figure

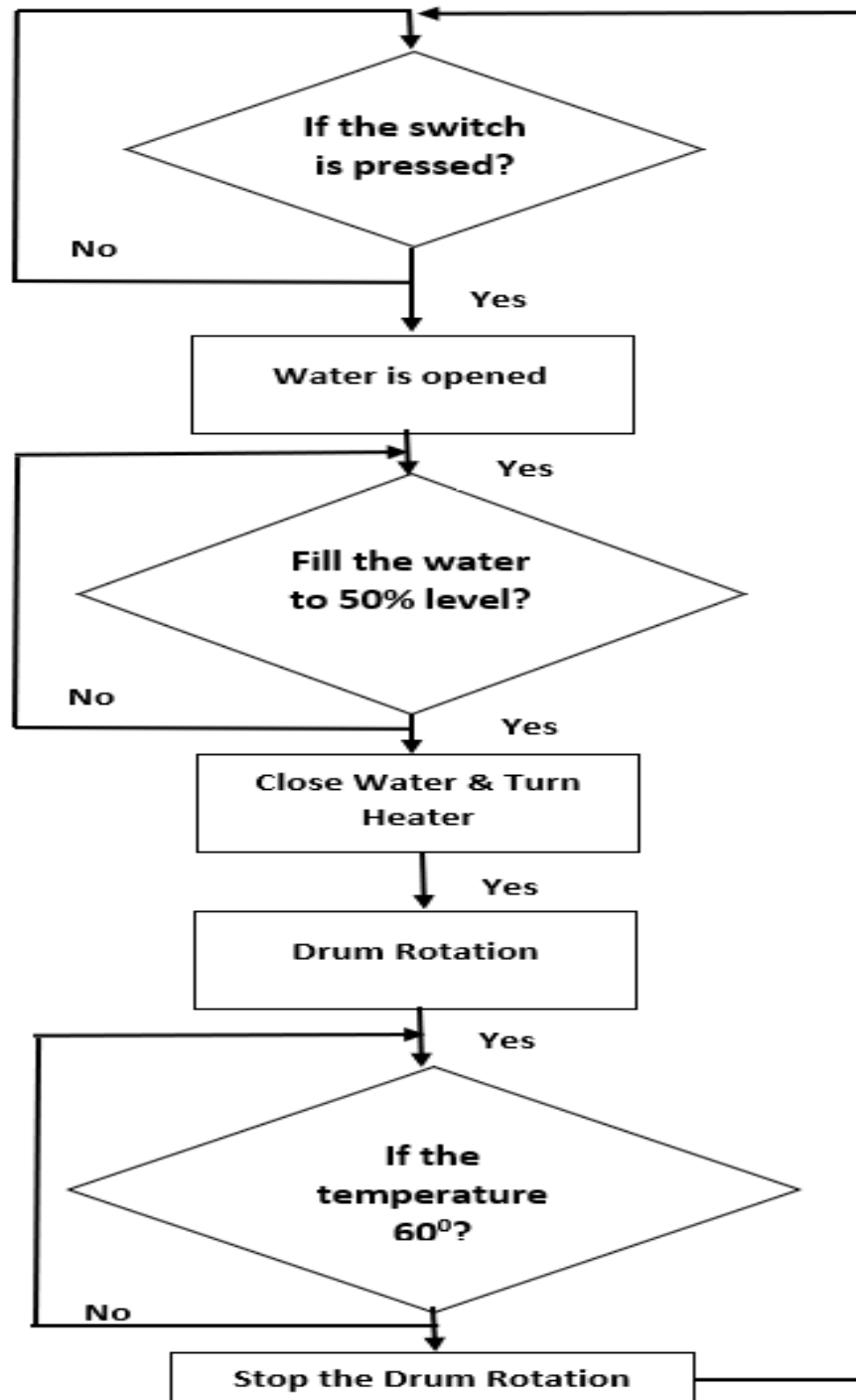


Figure 4.1: Flow Chart of the Control of a Model Washing Machine.

Hardware Architecture

The project hardware design is depicted below in form of a block diagram. See fig. 4.2.

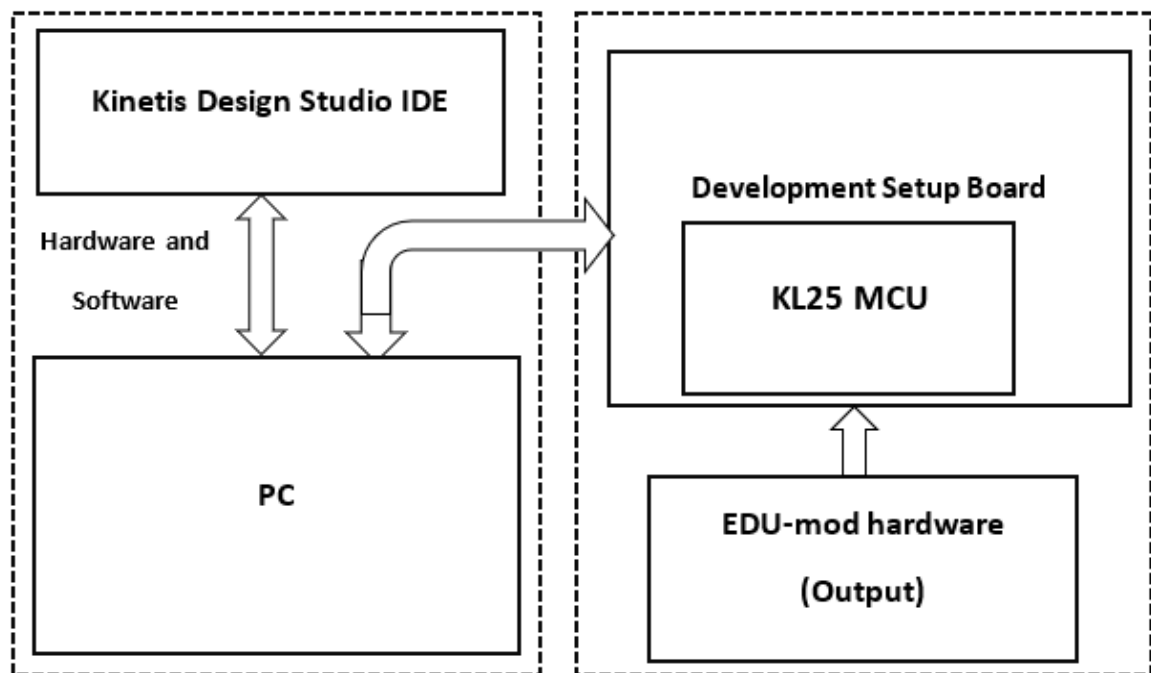


Figure 4.2: Block Diagram of the Hardware Architecture.

Software Design

In this project, we use the following software components in controlling the outputs on EDU-mod hardware. The features include the following:

- Super Loop software architecture:
In every embedded C application, there is a set of statement which needs to be implemented continually. Thus, to execute set of statement, there is need for a loop that must run non-stop. Such type of loops are referred to as Super Loop. In this project, Super Loop statement is used in order to keep the program running. See figure 4.3.

```

// Prototype
void init(void);
static inline int IsKeyPressed(int pin);

int main(void)
{
    // Initialize the pins, interrupts,
    init();

    // Wait for the user to press switch SW1. // When pressed, continue with the next step
    while ( !IsKeyPressed(SWITCH_PIN) );
    // Fill WATER to level 50%
    // Open water
    PTE->PSOR = (1 << WATER_PIN);

    // Wait for sensor LEVEL50% to go HIGH, then close WATER:
    // while ( LEVEL50% is low ) do nothing;
    while ( (PTC->PDIR & (1 << LEVEL50_PIN )) == 0 );
    // Close WATER
    PTE->PCOR = (1 << WATER_PIN);

    //time delay
    delay(2000);

    return 0;
}

```

Figure 4.3: The Super Loop Architecture Program.

- Software to control GPIO pins.
- Software to read the status of GPIO pins.

Input and outputs of the model (assigned pin)

The model is connected to the development board. The status of the model is controlled using GPIO pins. The table below shows the ports and pins for this model.

Table 4.1: Input and Output Pins.

S/N	MCU Pin	Model function
1.	E0	Water
2.	E1	Heater
3.	E4	Drum high speed
4.	E3	Not used
5.	E5	Drum rotation →
6.	C1	Drum rotation ←
7.	B1	Not used
8.	D0	Pump (Empty the machine)

9.	C16	Temperature 40°C
10.	D5	Not used
11.	D4	Water level 100%
12.	D2	Temperature 60°C
13.	D3	Temperature 30°C
14.	C7	Not used
15.	C6	Water level 50%
16.	C5	Temperature 90°C

Building and Execution of the Program

Below are the steps used in building this program:

- Create new project in Kinetis Design Studio.
- In the main.c file, the code is written to initialize the GPIO pins.
 - Enable the clock for the port of the pin.
 - Set the pin function to GPIO.
 - Set the pin direction to output or input.
- Write the code to control the washing machine. Algorithm used in this program is stated below:
 - Wait for the user to press the switch D0
 - Fill water to 50% level, pin C6.
 - Turn on the heater, pin E1.
 - Check the temperature, pin D2.
 - The spin process should begin both left and right direction, pin E5 and C1.
 - Empty the WM, pin D0.
 - Ready for next button press Reset button.
- Next step in this case is to convert each of the statements into real code in C language. For example, the figure below shows if the user presses switch 1 on the development board.

```
#include "MKL25Z4.h"

// Pin names
#define SWITCH_PIN      (4)      // A4
#define WATER_PIN      (0)      // E0
#define HEATER_PIN     (1)      // E1
#define LEVEL100_PIN   (4)      // D4
#define LEVEL50_PIN    (6)      // C6

// Prototype
void init(void);
static inline int IsKeyPressed(int pin);

int main(void)
{
    // Initialize the pins, interrupts,
    init();

    // Wait for the user to press switch SW1.
    while ( !IsKeyPressed(SWITCH_PIN) );
}
```

Figure 4.4: Switch Control.

- Now we can build the program and run it.
- When the program loads, there is an option to step through the code using Step Over button in debugger. When we step over the function, *while(!IsKeyPressed(SWITCH_PIN))* the program will run until the switch SW1 is pressed on the learning kit. Then, once it stops at the next line, Stepping can be continued.

The Program

In this section, the program is designed and implemented to detect both the connection between the computer and development board via USB and the switch pressed on the development board. The figure below depicts the full software part of this project using Kinetis Design Studio IDE. See figure 4.5.

```

#include "MKL25Z4.h"

// Pin names
#define SWITCH_PIN      (4)      // A4
#define WATER_PIN      (0)      // E0
#define HEATER_PIN     (1)      // E1
#define LEVEL100_PIN   (4)      // D4
#define LEVEL50_PIN    (6)      // C6

// Prototype
void init(void);

static inline int IsKeyPressed(int pin);

int main(void)
{
    // Initialize the pins, interrupts,
    init();

    // Wait for the user to press switch SW1.
    // When pressed, continue with the next step.
    while ( !IsKeyPressed(SWITCH_PIN) );

    // Open water
    // Fill WATER to level 50%
    PTE->PSOR = (1 << WATER_PIN);

    // Wait for sensor LEVEL50% to go HIGH, then close WATER:
    // while ( LEVEL50% is low ) do nothing;
    while ( (PTC->PDIR & (1 << LEVEL50_PIN)) == 0 );

    // Close WATER
    PTE->PCOR = (1 << WATER_PIN);

    return 0;
}

// Initialize the pins for the Mix model
void init(void)
{
    // Enable clock for ports A, B, C, D, E
    SIM->SCGC5 |= (SIM_SCGC5_PORTA_MASK | SIM_SCGC5_PORTB_MASK |
SIM_SCGC5_PORTC_MASK | SIM_SCGC5_PORTD_MASK | SIM_SCGC5_PORTE_MASK );

    // Set pin function to GPIO
    PORTA->PCR[SWITCH_PIN] = PORT_PCR_MUX(1);
    PORTE->PCR[WATER_PIN] = PORT_PCR_MUX(1);
    PORTE->PCR[HEATER_PIN] = PORT_PCR_MUX(1);
    PORTD->PCR[LEVEL100_PIN] = PORT_PCR_MUX(1);
    PORTC->PCR[LEVEL50_PIN] = PORT_PCR_MUX(1);

    // Set pin direction
    PTE->PDDR |= (1 << WATER_PIN); // WATER is output
    PTE->PDDR |= (1 << HEATER_PIN); // HEATER is output

    // LEVELs are inputs (sensors)
    PTD->PDDR &= ~(1 << LEVEL100_PIN);
    PTC->PDDR &= ~(1 << LEVEL50_PIN);

    // pull ups are not needed.

    // Set outputs to LOW
    PTE->PCOR = (1 << WATER_PIN);
    PTE->PCOR = (1 << HEATER_PIN);
}

```

```

/* Return 1 if the switch on given pin is pressed, 0 if not pressed.
**/
static inline int IsKeyPressed(int pin)
{
    if ((PTA->PDIR & (1 << pin)) == 0)
        return 1;
    else
        return 0;
} //EOF

```

Figure 4.5: Software Implementation using KDS.

4.4 Practical Tasks for Students

In this section, here is a common practical example for student to complete the program according to the flow chart above. See figure 4.1. Thus, student can control the rotation of the drum and the temperature of the washing machine using the below program. See figure 4.6 and figure 4.7.

The figure below is the solution for the control of the rotation of the drum and the temperature of the washing machine. See figure 4.6

```

//drum rotation, clockwise -->
PTE->PSOR = (1 << DRUM_PIN);

//Stop the rotation of the drum,
PTE->PCOR = (1 << DRUM_PIN);
///
PTE->PSOR = (1 << Heater_PIN);
//waiting for temp. to be 60C
while ( (PTD->PDIR & (1 << TEMP60_PIN )) == 0 );

///STOP the heating
PTE->PCOR = (1 << Heater_PIN);

//stop the rotation,
PTE->PCOR = (1 << DRUM_PIN);

/* Never leave main */
return 0;
}

```

Figure 4.6: Solution of the Task.

```

PORTE->PCR[DRUM_PIN] = PORT_PCR_MUX(1); // Pin for the rotation of the Drum
PORTD->PCR[TEMP60_PIN] = PORT_PCR_MUX(1);

// Set pin direction
PTE->PDDR |= (1 << WATER_PIN); // WATER is output
PTE->PDDR |= (1 << HEATER_PIN); // HEATER is output

//Direction of the drum rotation
PTE->PDDR |= (1 << DRUM_PIN ); // DRum rotation in clockwise direction

// LEVELs are inputs (sensors)
PTD->PDDR &= ~(1 << LEVEL100_PIN);
PTC->PDDR &= ~(1 << LEVEL50_PIN);
PTD->PDDR &= ~(1 << TEMP60_PIN); // setting the input temp to 60

// pull ups are not needed.

// Set outputs to LOW
PTE->PCOR = (1 << WATER_PIN);
PTE->PCOR = (1 << HEATER_PIN);
//
PTE->PCOR = (1 << DRUM_PIN);

```

Figure 4.7: Initialization of the Pins.

4.5 Theoretical Tasks and Solutions for Students

This section describes common tasks, questions student should understand in terms of Microcomputer Programming course. Below are the common questions and answers student should know:

- What are the various prerequisites to choose microcontroller?

Answer:

- Availability of software development apparatus for instance, compilers, debuggers, assemblers.
- Reliable sources or manufactures.
- Power consumption rate.
- The capacity of RAM and ROM on the chip.
- Cost per unit.

- What are the different types of memories used in microcontroller or microprocessor?

Answer:

- RAM - Random Access Memory.
- ROM - Read Only Memory.
- PROM - Programmable Read Only Memory.
- EPROM - Erasable Programmable Read Only Memory.

- EEPROM - Electrically Erasable Programmable Read Only Memory.

- What is the difference between microcontroller and microprocessor?

Answer:

- The main difference in both of them is the external peripheral. Microcontrollers have RAM, ROM, I/O ports embedded in them while microprocessor has no such features, but it uses external circuits.
- Peripheral of microcontroller are on single chip whereas in microprocessor it is bulky.
- The processing speed of microcontrollers ranges from 8MHz to 50MHz, whereas the processing of microprocessors is greater than 1GHz and works much faster compared to microcontrollers.
- Most importantly, microcontrollers are based on Harvard pattern in which the program memory and data memory are separate whereas microprocessor are based on von Neumann model in which program and data are saved in the same memory module.

- Why is infinite loop(super loop) needed in embedded systems?

Answer:

- Every embedded systems requires infinite loops for constantly processing or checking the state of the program. For instance, the case of a program state that is continuously being check in case of any exceptional errors that can occur during run time. Such errors could be memory outages, divide by zero, etc. Absence of infinite loop, could cause the embedded system to come to halt which is not the desired outcome.

- Analyse which buses are used for communication in embedded system?

Answer:

Buses used for communication in embedded systems includes the following:

- USB – In most cases, USB is used for communication between CPU and devices like development board.
- I2C – the I2C helps in communication between multiple ICs.
- CAN – it is applied in automobiles with centrally controlled network.

- Mention some of the commonly found errors in Embedded Systems?

Answer:

Some of the commonly errors found in embedded systems are:

- Data lines malfunctioning.
- Address line malfunctioning usually because of a short in circuit.
- Improper insertion of memory device into the memory slot.
- Due to some errors or garbage, some memory locations can be inaccessible in storage.
- Bad control signals.

5. CONCLUSIONS

Upon the closure of this project, we have described the material necessary for the course of Microcomputer Programming, EDU-mod hardware used as the control of a model washing machine.

Besides, the main features of EDU-mod hardware and its function along with the connection to the development board has been described in this project.

On top of that, the objective of this project has been clearly met. Most importantly the design and implementation of the software for controlling the model of washing machine connected to the development board has been structured and implemented in chapter four of this thesis. Preparation of tasks for students focused on understanding and expanding the software has been discussed in chapter four of this thesis. Lastly, the KL25 MCU, which is used in the control of a model washing machine was described in chapter 3.

The effort towards this project can be summarized as follows:

- Full study on microcontroller and its applications.
- Proper research on KL25 microcontroller and others.
- Proper study of EDU-mod hardware output during the running of the program.
- Flow chart of the program
- Schematic design of the hardware architecture of this model. (See chapter 4).
- Selection for the ideal microcontroller in regards to this project.
- Choosing KDS IDE in building the software in this project.
- Writing test code and driver software for the modules.
- Making the flow chart changes along with the design, based on the test results.

BIBLIOGRAPHY

- [1] H. Fan *et al.*, “Innovations of Microcontroller Unit Based on Experiment,” *2019 IEEE Int. Symp. Circuits Syst.*, pp. 1–5, 2019.
- [2] T. R. Lambert, “An introduction to microcontrollers and embedded systems,” no. March, 2018.
- [3] P. P. Debono, *PaulOS: Part I - An 8051 Real-Time Operating System*. 2015.
- [4] M. Barr, *Programming Embedded Systems in C and C++*. 1999.
- [5] D. Number and K. Rev, “Kinetic Design Studio V3.0.0- User’s Guide,” pp. 1–71, 2015.
- [6] F. Semiconductor, “FRDM-KL25Z User’s Manual,” *Free. Man.*, pp. 1–14, 2012.
- [7] CATSOULIS, John. *Designing embedded hardware*. 2nd ed. Sebastopol, CA: O’Reilly, 2005 xvi, 377p. ISBN 0596007558.
- [8] KOHOUT, Ludek. *Edumat.cz*[online]. [cit. 2018-11-26]. Available from: <http://edumat.cz/produkty.php?produkty=edumod>.
- [9] MCCONNELL, Steve. *Code complet*. 2nd. Redmond, Wash.: Microsoft Press, c2004. ISBN 978-0735619678.
- [10] MORTON, Todd D. *Embedded microcontrollers*. Upper Saddle River, N.J.: Prentice Hall, c2001. ISBN 0139075771.
- [11] SMITH, Warwick A. *C programming for embedded microcontrollers*. 2nd ed. Susteren: Elektor International Media BV, 2008. ISBN 978-090-5705-804.
- [12] Wikipedia. *Debugger* [online]. [Accessed 10th May 2019]. Available: <https://en.m.wikipedia.org/wiki/Debugger>.
- [13] Wikipedia. *Microcontrollers* [online]. [Accessed 10th May 2019]. Available: <https://simple.wikipedia.org/wiki/Microcontroller>.

LIST OF ABBREVIATIONS

ALU	-	Arithmetic logic unit.
ADC	-	Analog to digital converter.
ARM	-	Acorn RISC Machine.
CAN	-	Controller area network.
CPU	-	Central processing unit.
DAC	-	Digital to analog converter.
DMA	-	Direct memory channel.
EEPROM	-	Electrically erasable programmable read only memory.
EPROM	-	Erasable programmable read only memory.
ERR	-	Error.
GPIO	-	General purpose input output.
HW	-	Hardware.
IDE	-	Integrated development environment.
I2C	-	Inter integrated circuit.
I/O	-	Input and output.
IR	-	Instruction register.
KDS	-	Kinetis design studio.
LCD	-	Liquid crystal display.
LED	-	light-emitting diode.
MCU	-	Microcontroller unit.
PC	-	Personal computer.
PCB	-	Printed circuit board.
PROM	-	programmable read only memory.
PWM	-	Pulse width modulation.
RAM	-	Random access memory.

ROM	-	Read-only memory.
RTO	-	Real time optimization.
SFR	-	Special function register.
SPI	-	Serial peripheral interface.
USB	-	Universal serial bus.
VLSI	-	Very large-scale integration
WM	-	Washing machine.

LIST OF FIGURES

- Figure 2.1 Microcontroller Structure.
- Figure 2.2 KL25 32-pin Pinout Diagram.
- Figure 2.3 Block diagram of a Microprocessor.
- Figure 2.4 Workspace Dialog Box.
- Figure 2.5 Debugging Interface.
- Figure 2.6 Selecting a Program to Debug.
- Figure 2.7 Debug Perspective Switch Dialog Box.
- Figure 2.8 C/C++ Button Interface.
- Figure 2.9 Confirm Import Option Dialog Box.
- Figure 2.10 Select Directory Path.
- Figure 2.11 Selecting Drivers to Project.
- Figure 3.1 The FRDM-KL25 Board.
- Figure 3.2 ADC Block Diagram of KL25 MCU.
- Figure 3.3 Edu-Mod Hardware.
- Figure 4.1 Flow Chart of the Control of a Model Washing Machine.
- Figure 4.2 Block Diagram of the Hardware Architecture.
- Figure 4.3 The Super Loop Architecture.
- Figure 4.4 Switch Control.
- Figure 4.5 Software Implementation using Kinetis Design Studio.
- Figure 4.6 Solution of the Task.
- Figure 4.7 Initialization of the Pins.

LIST OF TABLES

Table. 2.1 Buttons and Features in KDS.

Table. 4.1 Input and Output Pins.