MICROCONTROLLER (M68HC11) FOR FLUID FLOW CONTROL

NOR ARINA BT ADAM

This thesis is submitted as partial fulfillment of the requirements for the award of the
Bachelor Degree of Electrical Engineering (Electronics)

Faculty of Electrical & Electronics Engineering
Kolej Universiti Kejuruteraan & Teknologi Malaysia

APRIL, 2006

ABSTRACT

Process control becomes one of the important elements in industry and chemical laboratory. The purpose of this project is to control fluid flow by using microcontroller (M68HC11) and other component such as solenoid valve, keypad and level detector. When entering value of liquid by using keypad, user can get the amount needed correctly. The opening and closing of the solenoid valve depending on the programming design. By using automatic control, it will prevent overflow occur to the system and suitable for industry and chemical laboratory where accuracy of liquid needed is critical. In daily life, microcontroller has made an invaluable contribution to electronics and logic design everywhere. On the other hand, their flexible and programmable natures have led us to embrace embedded functionality in every aspect of our worldly interactions. The implementation of the usage of microcontroller in controlling fluid flow would be an advantage in process control system. The integration between each part of hardware will determine the successfulness of this project.

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

1.1 Overview of Fluid Flow Control

There are many ways fluid affect our lives. In our daily life, we use water for many different purposes as drinking, cleaning, bathing, cooking and watering plant. At the industry, the importance of fluid in cooling system could not be neglect. Fluid flow control is an important function within any organization that employs fluids to carry on its daily operations. For example, when we buy gasoline from a service station, the pumps system include a flow meter to indicate how many liters we pumped so we can pay for just the amount we put into our car. Another significant application of fluid flow control is in the chemistry lab when the precise amount of liquid needed is critical. Fluid can be described as liquid and gasses. However, in this project water will use as an example of the fluid.

1.2 Project Objectives

The overall objective of this project is to give an alternative way to construct fluid flow control in automated design rather than using manual technique.

However the objectives of Part 1 of this project:

a. To get to know the function of microcontroller in control system
b. To design hardware and program for controlling valve in fluid flow control system.

c. To get familiar with the devices use in fluid flow system and to determine how they are functioning.

1.3 Scope of Project

Basically the scope of this project is designing hardware and program for the fluid flow control. There will be several hardware components to form this system including microcontroller (M68HC11A1), tanks, solenoid valve, water level detector and keypad. The system used microcontroller to control the level of fluid in tank 1. The solenoid valve is used to let the fluid flow through it. The keypad will be the input to the system which represents the amount of liquid needed.

The need of this project is to get familiarize with microcontroller by doing research and finding details either for the hardware or programming parts. Other than that, I need to understand how each component works individually and also their integration. The successfulness of this project depends on interaction ability between each component.

1.4 Problem Statement

Several industries usually use much of liquid in the process of produce and also for cooling. Controlling the amount of liquid in control system became one of the important matters to the industry and also in chemical laboratory. This can be achieving by using microcontroller and valve as control elements. Valve is a device that regulates the flow of liquid in a pipe or other enclosure. By using valve, the amount of fluid flow can be controlled depending on the programming design. This will prevent overflow occur to the system. This project is the best solution to which the accuracy of fluid measurement needed is critical.

1.5 Thesis Outline

Chapter 1 explains the overview of fluid flow control and its importance in daily life. The chapter also discusses the project objectives, scope of the project and thesis outline.

Chapter 2 focuses on the overview of the hardware and software needed in this project. This chapter also includes theoretical of control system.

Chapter 3 explains the literature reviews of this project based on journals and other references.

Chapter 4 discusses the process implementation of fluid flow control system hardware and programming. The chapter also explains in more details about microcontroller, solenoid valve, relay, keypad and Jbug11 that have been used in this project.

Chapter 5 presents all the results obtained and the limitation of the project. The discussion focused on the results based on the experiment and the analysis of the result.

Chapter 6 concludes overall progress of the project with discussion of the problem and the recommendation for this project and overall system for the future development or modification.

CHAPTER II

OVERVIEW OF HARDWARE AND SOFTWARE

2.1 Introduction

This chapter explains about microcontroller in details including application, versions, I/O Ports, Programming Model, Memory, Programming Language, Modes of Operation and Serial Communication System. Next, the discussion is about functions of relay, solenoid valve and the software requirement of this project. Lastly, this chapter presents theoretical concept of the control system and its histories.

2.2 Application of Microcontroller

A microcontroller is by definition is a computer on a chip. It includes all the necessary parts (including the memory) all in one IC. You just need to apply the power (and possibly clock signal) to that device and it starts executing the program programmed to it.

There is countless number of small electronic devices, which are nowadays based on microcontroller. A modern home can include easily tens or hundreds of microcontrollers, as almost every modern device which has electronics have a microcontroller (or more than one) inside.

A special application that microcontrollers are well suited for is data logging. We can monitor and record environmental parameters (temperature, humidity, rain, etc) easily. Small size, low power consumption, and flexibility make these devices ideal for unattended data monitoring and recording. The automotive market is probably the most important single driving force in the microcontroller market, especially at its high end.

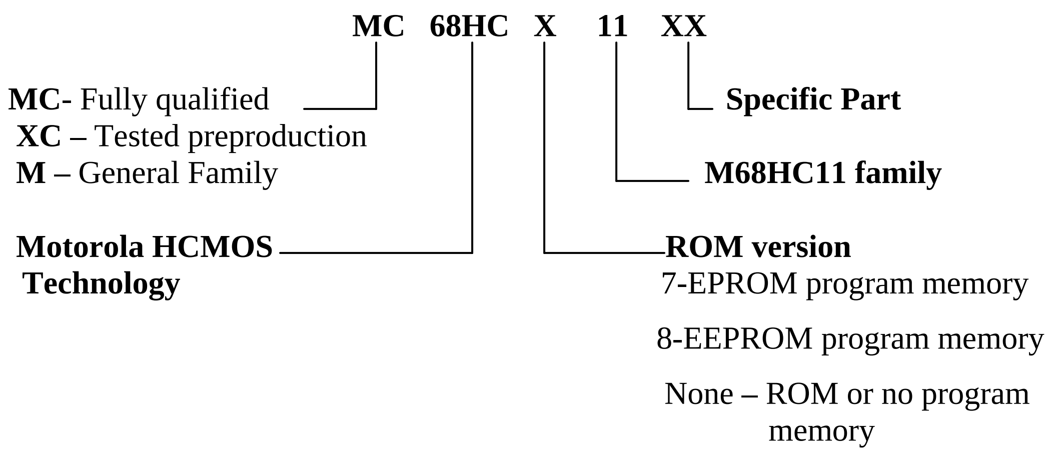
Several microcontroller families are specifically developed for automotive applications. For example the microcontroller can be used programmed to handle conditions when the engine is cold or warm, and when it’s accelerating or cruising. The driver can construct the control unit to optimize speed and acceleration performance of their vehicle.[?]

Microcontroller has made an invaluable contribution to electronics and logic design everywhere. Their flexible and programmable natures have led us to embrace embedded functionality in every aspect of our worldly interactions.

2.3 Microcontroller M68HC11

Microcontroller is a single chip used to control other devices. It contains a CPU, memory and I/O (input/output) circuits. M68HC11 is one of the microcontroller families from Freescale Semiconductor. It is an 8-bit controller chip with added features such as Analog to Digital Converter (ADC) and output compare. M68HC11 uses logic signals 5V for logic 1 and signals 0V for logic 0.

2.3.1 Versions of M68HC11



Nowadays, there are various version of M68HC11 available in market with different features.

A series

The is an advanced 8 bit MCU featuring 8Kbytes ROM, 256 bytes of RAM, 512 bytes of EEPROM, sophisticated on chip peripheral and a nominal bus speed of 3MHZ. 8 channel A/D converter with eight bits of resolution.

D series

The 68HC11D3 chip with 4 Kbytes ROM offers an economical alternative for applications when advanced 8 bit performance is required with fewer peripherals and less memory.

E series

In this series, the 68HC11E9 flexible I/O capability allows facilities to be configured to best match application. It was the first in the family to combine EEPROM and EPROM on a single chip. It also offers multiple memory sizes in a pin compatible package.

F series

High speed expanded systems required the development of the 68HC11F1. This particular series stands out with its extra I/O ports, an increase in static RAM, chip select and a 4 MHz non-multiplexed bus.

G series

The M68hc11G5 is the first family member to offer 10 bit A/D resolution. This series also includes the most sophisticated timer systems in the family.

K series

High performance device, the M68HC11K4 offers high speed, large memories, pulse width modulation (PWM) and plenty of I/O.

L series

The M68HC11L6 is a high speed, low power chip with a multiplexed bus capable of operation up to 3 MHz. The high performance design is based on M68HC11E9 include 16 Kbytes of ROM, plus an additional bidirectional port. Its fully static design allows operation at frequencies down to dc.

M series

These enhanced high performance microcontrollers are derived from the M68HC11K4 and include large memory modules, a 16-bit math coprocessor.

P series

The 68HC11P2 offers a power saving programmable PLL-based clock circuit along with many I/O pins, large memory and 3 SCI ports. All M68HC11 family members have on chip SCI and SPI. Most members have EEPROM and A/D converter.

2.3.2 I/O Ports

There are several ports connected to the microcontroller depends on its version. All ports in M68HC11 are multiplexed which allow each port performs various functions with one task at one time[4].

i. **Port A** - Parallel I/O or timer /counter
ii. **Port B** – Output port or upper address (A8 –A15) in expanded mode
iii. **Port C** – I/O port or lower address (A0-A7) and data bus (D0-D7) in expanded mode
iv. **Port D** – 6 bits I/O port or Serial Communication Interface(SCI) and Serial Peripheral Interface(SPI)
v. **Port E –** Input port or 8 channel input analog to Analog to Digital Converter(ADC)

2.3.3 M68HC11 Programming Model

The CPU registers are an important part of the microcontroller that I need to know when designing a program[1]. Figure 2.1 shows the register and condition code register (CCR) for M68HC11.

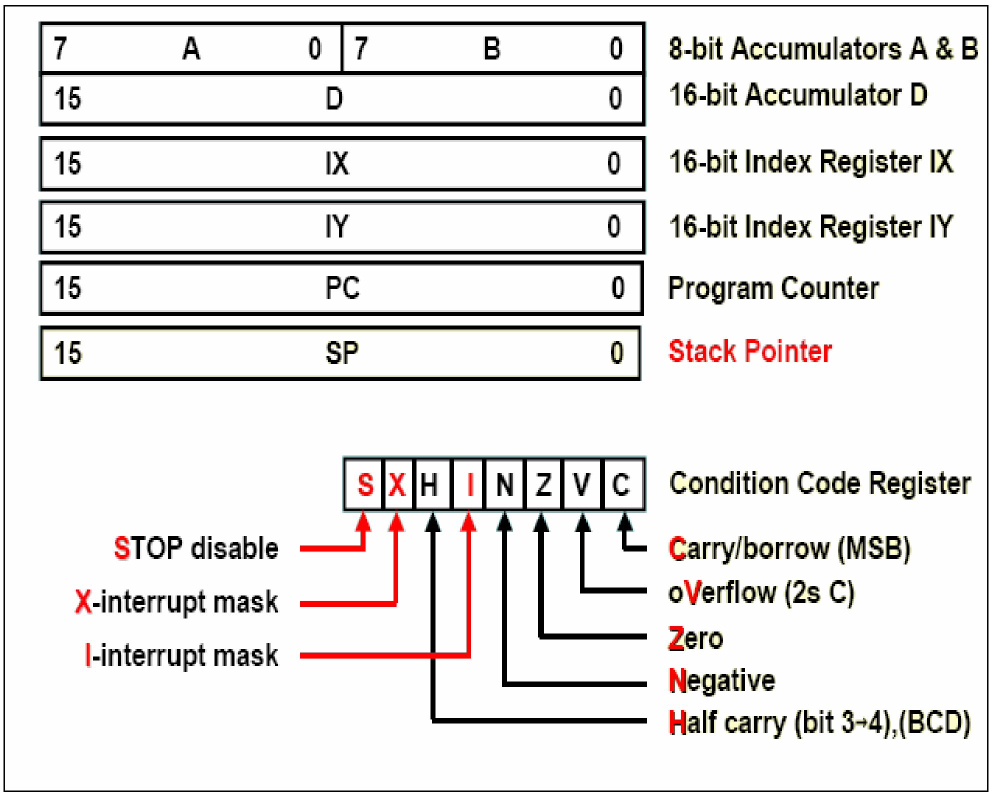


Figure 2.1: M68HC11 Programming Model

Accumulator A and B are general purpose used to hold operands and results of arithmetic calculation or data manipulation. Some instruction treats the two of this 8-bit microcontroller as a double 16-bit accumulator, accumulator D. Program counter represent the address of next instruction. Two 16-bit index register (IX and IY) is the register that will be use in index mode operation. Stack pointer (SP) shows the block of address used to store temporary data that can be accessed. Condition Code Register indicates status of the operation by using flags.

2.3.4 Memory

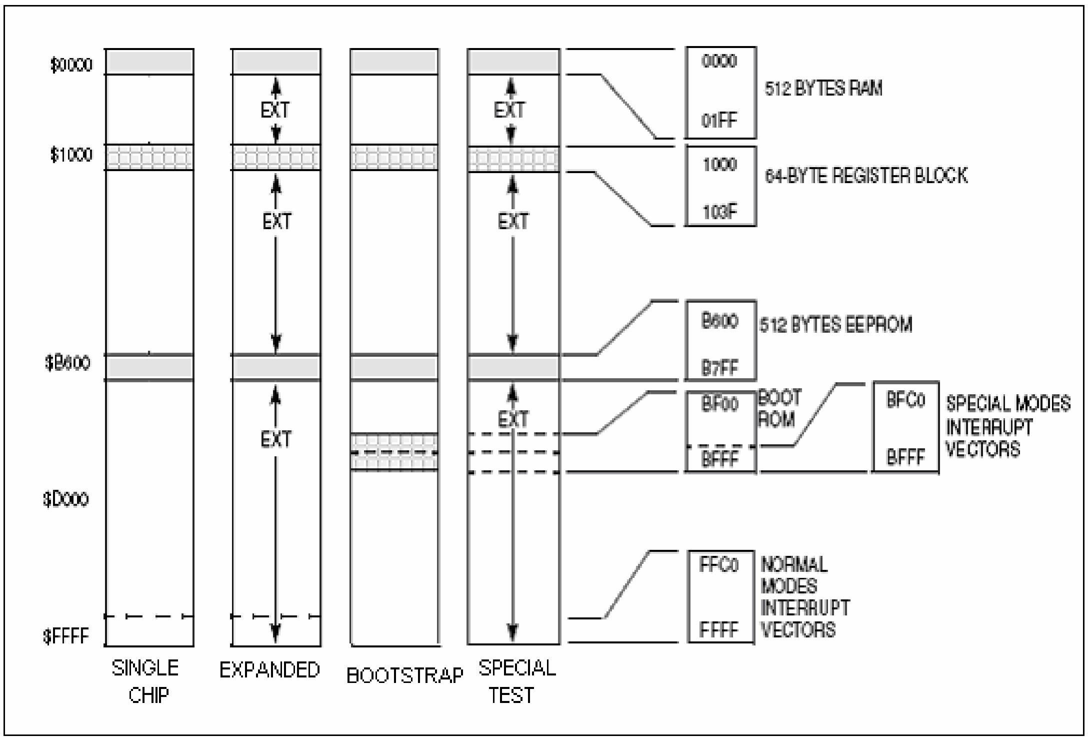


Figure 2.2 : Memory block of M68HC11

Memory is a place to store data and program code. Basically, there are three types of memory: *read only memory (ROM), random access memory (RAM), electrically erasable programmable ROM (EEPROM)* but some microcontrollers do have *erasable programmable ROM (EPROM)* instead of ROM. Figure 2.2 show memory maps for each mode of operation in M68HC11. I will explain the different of each memory in brief.

i. **Read Only Memory (ROM)**

ROM is nonvolatile, as the user will not lose the data even when the power is shut off. It contains permanent data such as program code and constant data. The main use of this memory is to hold the user's application program instructions. These instructions could not be change since it is programmed into the microcontroller unit (MCU) when it is manufactured[1]. User must specify the data that need to be put in the internal ROM when purchasing it.

ii. **Random Access Memory (RAM)**

RAM is volatile because the data will lose when the power is turned off. Each location of this memory is accessible, independent of its physical location[5]. The program can be writing into the memory by using programming instruction. There are 2 types of RAM: *Dynamic RAM (DRAM)* and *Static RAM (SRAM)* but M68hc11 use static as its internal RAM. This is because SRAM are faster and simpler rather than DRAM.

iii. **Electrically Erasable Programmable ROM (EEPROM)**

By using EEPROM, user can program and erase an addressed byte electrically[2]. The EEPROM can store small program codes that are different for each user and to back up critical data. It does not require external high voltage to program and erasing as it has built in voltage charge pump.

iv. **Erasable Programmable ROM (EPROM**)

EPROM is a user programmable and erasable device. User can program the EPROM by using its programmer and erase by exposing the window transparent quartz (in the chip) to high intensity ultraviolet (UV) light for a period of time.

2.3.5 Programming Language

Microcontroller use machine language to execute the instruction given by the user. Machine language is the only language understands by the microcontroller and it differs for each processor and normally it is upward compatible [4]. For example, the 68HC11 could not run programs written in the machine language for ATMEL microcontroller and a 68HC12 can run program written for 68HC11. The assembly language is binary numbers written in special code.

User does not write binary instruction but they used other language that will be translated into machine language. One of these languages is assembly language. Different CPU will require different assembly language and it is the mnemonic form of machine language. Usually assembly language is more efficient than the one that written in high level language [5]. The assembly language is used for application, which is more directly to hardware.

High-level language is portable as user can write a program and translate to machine code for each type of CPU. This language is easier to understand because they are more human like language [4]. Examples of high-level language including Basic, COBOL, C and Java.

2.3.6 Modes of Operation

M68HC11 has four modes of operation. Modes of operation can be determined by using MODA and MODB pins during the time of reset [3]. Table 1 below shows the connection used to select the M68HC11 mode.

Table 2.1: Selection Mode of M68HC11

Inputs		
MOD A	**MOD B**	**Mode Description**
0	0	Special Bootstrap
0	1	Single Chip
1	0	Special Test
1	1	Expanded

i. Single Chip Mode

In this mode, only one single chip is required and no extra memory or I/O chips needed. The single chip mode ideally suited for simple system with few parts connections. All I/O are located on the microcontroller chip and external bus is not required. Advantages of using this mode are cost saving and more reliable as less external connection is necessary.

ii. Expanded Mode

This mode is applicable when user requires more memory or I/O subsystem than are provided in the chip. The expanded mode uses ports B and C of the microcontroller as an address and data bus. Port B is used for upper address bits A15-A8 of a 16 bits addresses while Port C carries the low byte of the address and the data byte. The address and data byte are multiplexed onto Port C as they cannot appear on the bus at the same time[1].

iii. Special Bootstrap Mode

Bootstrap mode is used only to load a test program. The test operation can be done in a process called bootstrapping. When M68HC11 operates in bootstrap mode, it uses different ROM than the other two modes I discussed before. The accessible block of this mode is a 192 byte-block with an address range of $BF40 to $BFFF [3]. It also uses different vectors named as bootstrap mode interrupt vector. The ROM has a bootloader program that is permanent part of the chip. This mode has other optional task which is writing to EEPROM.

iv. Special Test Mode

This mode is only used by manufacturer (Motorola) to test the chip in factory. Special test mode allows Motorola to change some register in the microcontroller.

2.3.7 Serial Interface

There are two ways computer can transfer data: *serial* and *parallel* connections. Parallel connections require more lines to transfer data to a device resulting in faster transmission [4]. However, the distance between devices is limited and design of connection is more complex than serial. The serial communication use of single line allowing the system to be cheaper and can be use in greater length. It contains two subsystems: *serial communication interface (SCI)* and *serial peripheral interface (SPI).*

SCI has on chip baud rate generator derives standard baud rate frequencies from the MCU oscillator. The SCI transmitter and receiver are independent but use the same data format and baud rate [3]. SCI also known as asynchronous communication transfers a single byte at a time. The transmitter can send characters at any rate resulting in time delay between the transmissions of each character.

Synchronous communication a.k.a SPI has flexible I/O pin control which allows the direction of each pin to be controlled by software. It can be configured to interface directly with numerous products available in the market. The system uses Port D and when the SPI subsystem is enabled; all SPI defined inputs are configured to be inputs regardless of the data direction bits.

2.4 Valve

A valve is a mechanical device that controls the flow of fluid and pressure within a system or process [13]. A valve controls system or process fluid flow and pressure by stopping and starting fluid flow Varying (throttling) the amount of fluid flow. Valve also controlling the direction of fluid flow and regulate downstream system or process pressure. In addition, it relieve component or piping over pressure.

There are many valve designs and types that satisfy one or more of the functions identified above. A multitude of valve types and designs safely accommodate a wide variety of industrial applications for instance globe valve, gate valve, butterfly valve, solenoid valve and many more.

2.5 Relay

A relay is best defined as a switch that is operated by an electromagnet. A relay controller is a device that is used to control a bank of switches. A relay controller works by turning on and off magnetic coils.

Relays are ideally suited for controlling everything from lights and motors to telecommunication, audio, and video signals. Some relays can be used for switching radio frequency signals. Relays come in many sizes and ratings.

Relays typically have two or three connections: Common, Normally Open, and Normally Closed. The Common is the part of the relay that actually makes a mechanical movement [12]. By default, many relays have their common (COM) lead connected to the normally closed lead (NC).

Normally-open contacts connect the circuit when the relay is activated; the circuit is disconnected when the relay is inactive. It is also called Form A contact or "make" contact. Form A contact is ideal for applications that require to switch a high-current power source from a remote device.

Normally-closed contacts disconnect the circuit when the relay is activated; the circuit is connected when the relay is inactive. It is also called Form B contact or "break" contact. Form B contact is ideal for applications that require the circuit to remain closed until the relay is activated.

Change-over contacts control two circuits: one normally-open contact and one normally-closed contact. It is also called Form C contact [12]

2.6 Assembler and Software

The Assembler is a two-pass assembler. During the first pass, the source program is read to develop the symbol table. During the second pass, the object file is created (assembled) utilizing the table developed in pass one. It is during the second pass that the source program listing is also produced [1].

Each source statement is processed completely before the next source statement is read. As each statement is processed, the Assembler examines the label, operation code, and operand fields. The operation code table is scanned for a match with a known opcode. During the processing of a standard operation code mnemonic, the standard machine code is inserted into the object file. If an Assembler directive is being processed, the proper action is taken. The Assembler output includes an optional listing of the source program and an object file, which is in the Motorola S Record format.

The Assembler listing has the format:

LINE# ADDR OBJECT CODE BYTES [# CYCLES] SOURCE LINE

i. The *LINE#* is a 4 digit decimal number printed as a reference, and is used in the cross reference.
ii. The *ADDR* is the hex value of the address for the first byte of the object code for this instruction.
iii. The *OBJECT CODE BYTES* are the assembled object code of the source line in HEX. If a source line causes more than 6 bytes to be output (e.g. a long FCC directive), additional bytes (up to 64) are listed on succeeding lines with no address preceding them.
iv. The *# CYCLES* will only appear in the listing if the "c" option is in effect. It is enclosed in brackets which helps distinguish it from the source listing.
v. The *SOURCE LINE* is reprinted exactly from the source program, including labels.

The S-record output format encodes program and data object modules into a printable (ASCII) format. This allows viewing of the object file with standard tools and allows display of the module while transferring from one computer to the next or during loads between a host and target.

The S-record format also includes information for use in error checking to insure the integrity of data transfers. S-Records are character strings made of several fields which identify the record type, record length, memory address, code/data, and checksum [5]. Each byte of binary data is encoded as a 2-character

S-Record example

The following is a typical S-record module:

S1130000285F245F2212226A000424290008237C2A

S11300100002000800082629001853812341001813

S113002041E900084E42234300182342000824A952

S107003000144ED492

S9030000FC

The module consists of four code/data records and an S9 termination record. The first S1 code/data record is explained as follows:

i. **S1** S-record type S1, indicating a code/data record to be loaded/verified at a 2- byte address

ii. **13** Hex 13 (decimal 19), indicating 19 character pairs, representing 19 bytes of binary data, follow.

iii. **00** Four-character 2-byte address fields: hex address 0000, indicates the location where the following data is to be loaded.

The next 16 character pairs are the ASCII bytes of the actual program code/data

2.7 .. Theoretical of Control System

Control system can be defined as a system consisting of a computer, process control equipment, and a process interface. One type of control system is automatic control that has become an important part in manufacturing and industrial processes. Application of automatic control is not only limited to automobile industries or in the design of autopilot system (aerospace industries) but also essential in industrial operations as controlling pressure, temperature, humidity and flow in process industry.

2.7.1 History of Automatic Control

J.C. Maxwell provided the first rigorous mathematical analysis of a feedback control system in 1868. Thus, relative to this written language, the period before about 1868 is called the *prehistory* of automatic control. Following Friedland [1986], the period from 1868 to the early 1900's the *primitive period* of automatic control. It is standard to call the period from then until 1960 the *classical period*, and the period from 1960 through present times the *modern period*

The Industrial Revolution in Europe enlightened the introduction of self driven machines. By that time, the invention of the machines is more focused on grain mills, furnaces, boilers, and the steam engine. These devices could not be adequately regulated by hand, and the idea of automatic control systems is come into view.

A variety of control devices was invented, including float regulators, temperature regulators, pressure regulators, and speed control devices. The speed control of a steam engine in the eighteenth century is one of the significant designs done by James Watt.

The mathematical analysis of control systems had heretofore been carried out using differential equations in the *time domain*. At Bell Telephone Laboratories during the 1920's and 1930's, the *frequency domain* approaches developed by P.-S. de Laplace (1749-1827), J. Fourier (1768-1830), A.L. Cauchy (1789-1857), and others were explored and used in communication systems.

H. Nyquist [1932] derived *Nyquist stability criterion* based on the polar plot of a complex function. H.W. Bode in 1938 used the magnitude and phase *frequency response plots* of a complex function [Bode 1940]. He investigated closed-loop stability using the notions of *gain and phase margin*.

The development of *nuclear reactors* during the 1950's was a major motivation for exploring industrial process control and instrumentation. This work has its roots in the control of chemical plants during the 1940's.

By 1970, with the work of K. Åström [1970] and others, the importance of digital controls in process applications was firmly established. The work of C.E. Shannon in the 1950's at Bell Labs had revealed the importance of sampled data techniques in the processing of signals. The applications of *digital filtering theory* were investigated at the Analytic Sciences Corporation [Gelb 1974] and elsewhere.

As we know, classical design techniques could be employed by hand using graphical approaches. On the other hand, modern controls design requires the solution of complicated nonlinear matrix equations. It is privileged that in 1960 there were major developments in another area- digital computer technology. Without computers, modern control would have had limited applications.

2.7.2 Terminologies of Control System

There are several terminologies that contribute to the definition of control system:

Processes. The Merriam-Webster Dictionary defines a process to be natural, progressively continuing operation or development marked by a series of gradual changes that succeed one another in a relatively fixed way and lead toward a particular result or end; or an artificial or voluntary, progressively continuing operation that consists of a series of controlled actions or movements systematically directed toward a particular result or end

Systems. A system is a combination of various components that work together to achieve a certain objective.

Controlled Variable and Manipulated Variable. The controlled variable is the quantity that is measured and can be managed. The manipulated variable is the quantity that is varied by the controller to affect the value of the controlled variable.

Feedback Control. An operation of reducing the differentiation between input and output system.

Disturbances. A disturbance can be described as a signal that can affect the output value of a system