SHEET METAL THICKNESS CONTROLLER

A Project Report

Submitted By

Karthick. A	-	71206107016
Nandha Kumar. S	-	71206107025
Ramkumar. E	-	71206107036
Arvind Sachidev Chilkoor	-	71206107302

In partial fulfilment for the award of the degree

Of

Bachelor of Engineering

In

ELECTRONICS & INSTRUMENTATION ENGINEERING

Department of Electronics & Instrumentation Engineering

KUMARAGURU COLLEGE OF TECHNOLOGY

COIMBATORE – 641 006

ANNA UNIVERSITY CHENNAI: CHENNAI 600 025

APRIL 2010

ANNA UNIVERSITY CHENNAI: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report entitled **"Sheet Metal Thickness Controller"** is the bonafide work of,

Karthick. A	- Register No. 71206107016
Nandha Kumar. S	- Register No. 71206107025
Ramkumar. E	- Register No. 71206107036
Arvind Sachidev Chilkoor	- Register No. 71206107302

who carried out the project work under our supervision.

[illegible]

Signature of H.O.D

Prof. R. Annamalai

H.O.D

1. [illegible] 13\04\10

2. [illegible] 13/04/10.

Signature of SUPERVISORS

1. Mr. S. Arun Jayakar (Sr.Lecturer)

2. Mr. S. Prem Anand (Lecturer)

Certified that the candidate(s) with the University Register No(s).

1. **71206107016**
2. **71206107025**
3. **71206107036**
4. **71206107302**

Were examined in project viva voce examination held on 15 / 04 / 2010

[illegible] 15\4\2010

External Examiner

Date:

[illegible] 15|04|10

Internal Examiner

Date: 15|04|10

Department of Electronics & Instrumentation Engineering

स्टील अथॉरिटी ऑफ इण्डिया लिमिटेड
STEEL AUTHORITY OF INDIA LIMITED
सेलम इस्पात संयंत्र
SALEM STEEL PLANT

Ref No TR-15(4)/1863

March 25, 2010

Certificate

Certified that
Shri Arvind Sachidev Chilkoor
Shri A Karthick
Shri S Nandha Kumar
Shri E Ramkumar
IV year BE(E&I) students of
Kumaraguru College Of Technology
Coimbatore
have done a Project on
"SHEET METAL THICKNESS CONTROLLER"
in **CENTRALISED ELECTRICAL MAINTENANCE**
department of Salem Steel Plant
From **03/02/2010** to **25/03/2010**

[illegible]

R RAMANI
Asst Manager(HRD)

Abstract

In the steel manufacturing plant, it is a common format for the steel to be shaped into sheet metal rolls as a final product. It is often a laborious task to accurately maintain the thickness of the sheet metal. It is manually pre-calibrated before the rolling begins, by the use of gauges that measure the gap between the rollers, also due to the extremely high temperature (approx 1280 Deg Celsius) no human interference is possible to take corrective action, in the event that an error has occurred.

The **"Sheet Metal Thickness Controller"** proposes to automate the entire process, and also accurately maintain the thickness by a unique combination of integrating a Stepper Motor, PIC controller and Ultrasonic sensors. The objective is to constantly keep monitoring the sheet metal that passes out of the roller upon each pass, any discrepancy noted will be detected by the sensors and corresponding changes will be incorporated through the rollers with respect to the previously set thickness value.

Special Acknowledgements

We would like to express our sincere thanks to **SAIL (Salem)** who had given us the opportunity to do our final year project in their esteemed establishment. Their valuable guidance, suggestions and encouragement paved way for the success of this project.

We would like to extend our sincere gratitude to our external guide,

Mr. Puliannan, Deputy Manager, CRM Division, SAIL (Salem)

who was instrumental in the success of the project with his valuable experience and knowledge in every step of the project work.

Acknowledgements

We would like to sincerely thank our respected

Principal Dr. S. Ramachandran, respected ***H.O.D, Prof. R. Annamalai.***

Our ***internal guide Mr. S. Prem Anand M. E.***

Our ***project coordinator Mr. S. Arun Jayakar M. Tech.***

Our ***class advisor Mrs. B. Veena Abirami M.E,***

for their kind support in providing the necessary facilities, timely guidance and encouragement to complete the project.

We also wish to extend our sincere gratitude to all the teaching and non-teaching staff in the E&I department for their help and cooperation.

Finally we would like to thank our parents for rendering their complete support and blessings, also to god for his kind benevolence and giving us the strength and wisdom to face obstacles with ease.

TABLE OF CONTENTS

List of Figures

List of Tables

Chapter – 1

1.1 INTRODUCTION

Steel is an alloy consisting mostly of iron, with carbc n content between 0.2% and 2.1% by weight, depending on the grade. Carbon is the most cost-effective alloying material for iron, but various other alloying elements are used, such as manganese, chromium, vanadium, and tungsten. Carbon and other elements act as a hardening agent, preventing dislocations in the iron atom crystal lattice from sliding past one another. Varying the amount of alloying elements and form of their presence in the steel (solute elements, precipitated phase) controls qualities such as the hardness, ductility, and tensile strength of the resulting steel. Steel with increased carbon content can be made harder and stronger than iron, but is also less ductile. Alloys with higher carbon content are known as cast iron because of their lower melting point and castability. Steel is also distinguished from wrought iron, which can contain a small amount of carbon, but it is included in the form of slag inclusions. Two distinguishing factors are steel's increased rust resistance and better weldability.

Though steel had been produced by various inefficient methods long before the Renaissance, its use became more common after more efficient production methods were devised in the 17th century. With the invention of the Bessemer process in the mid-19th century, steel became an inexpensive mass-produced material. Further refinements in the process, such as basic oxygen steelmaking, further lowered the cost of production while increasing the quality of the metal. Today, steel is one of the most common materials in the world, with more than 1300 million tons produced annually. It is a major component in buildings, infrastructure, tools, ships, automobiles, machines, appliances, and weapons.

Modern steel is generally identified by various grades of steel defined by various standards organizations.

Iron, like most metals, is found in the Earth's crust only in the form of an ore, ie. combined with other elements such as oxygen or sulphur. Typical iron-containing minerals include Fe_2O_3—the form of iron oxide found as the mineral hematite, and FeS_2—pyrite (fool's gold). Iron is extracted from ore by removing oxygen and combining the ore with a preferred chemical partner such as carbon. This process, known as smelting, was first applied to metals with lower melting points, such as tin, which melts at approximately 250 °C (482 °F) and copper, which melts at approximately 1,000 °C (1,830 °F). In comparison, cast iron melts at approximately 1,370 °C (2,500 °F). All of these temperatures could be reached with ancient methods that have been used since the Bronze Age. Since the oxidation rate itself increases rapidly beyond 800 °C, it is important that smelting take place in a low-oxygen environment. Unlike copper and tin, liquid iron dissolves carbon quite readily. Smelting results in an alloy (pig iron) containing too much carbon to be called steel. The excess carbon and other impurities are removed in a subsequent step.

Other materials are often added to the iron/carbon mixture to produce steel with desired properties. Nickel and manganese in steel add to its tensile strength and make austenite more chemically stable, chromium increases hardness and melting temperature and vanadium also increases hardness while reducing the effects of metal fatigue. To prevent corrosion, at least 11% chromium is added to steel so that a hard oxide forms on the metal surface; this is known as stainless steel. Tungsten interferes with the formation of cementite, allowing martensite to form with slower quench rates, resulting in high speed steel. On the other hand, sulphur, nitrogen, and phosphorus make steel more brittle, so these commonly found elements must be removed from the ore during processing.

The density of steel varies based on the alloying constituents, but usually ranges between 7.75 and 8.05 g/cm^3 (0.280–0.291 lb/in^3).

Even in the narrow range of concentrations which make up steel, mixtures of carbon and iron can form a number of different structures, with very different properties. Understanding such properties is essential to making quality steel. At room temperature, the most stable form of iron is the body-centered cubic (BCC) structure α-ferrite. It is a fairly soft metallic material that can dissolve only a small concentration of carbon, no more than 0.021 wt% at 723 °C (1,333 °F), and only 0.005% at 0 °C (32 °F). If the steel contains more than 0.021% carbon then it transforms into a face-centered cubic (FCC) structure, called austenite or γ-iron. It is also soft and metallic but can dissolve considerably more carbon, as much as 2.1% carbon at 1,148 °C (2,098 °F)), which reflects the upper carbon content of steel.

When steels with less than 0.8% carbon, known as hypo eutectoid steel, are cooled from an austenitic phase the mixture attempts to revert to the ferrite phase, resulting in an excess of carbon. One way for carbon to leave the austenite is for cementite to precipitate out of the mix, leaving behind iron that is pure enough to take the form of ferrite, resulting in a cementite-ferrite mixture. Cementite is a hard and brittle intermetallic compound with the chemical formula of Fe_3C. At the eutectoid, 0.8% carbon, the cooled structure takes the form of pearlite, named after its resemblance to mother of pearl. For steels that have more than 0.8% carbon the cooled structure takes the form of pearlite and cementite.

Perhaps the most important polymorphic form is martensite, a metastable phase which is significantly stronger than other steel phases. When the steel is in an austenitic phase and then quenched it forms into martensite, because the atoms "freeze" in place when the cell structure changes from FCC to BCC. Depending on the carbon content the martensitic phase takes different forms. Below approximately 0.2% carbon it takes and α ferrite BCC crystal form, but higher carbon contents take a body-centered tetragonal (BCT) structure. There is no thermal activation energy for the transformation from austenite to martensite. Moreover, there is no compositional change so the atoms generally retain their same neighbours.

Martensite has a lower density than austenite does, so that transformation between them results in a change of volume. In this case, expansion occurs. Internal stresses from this expansion generally take the form of compression on the crystals of martensite and tension on the remaining ferrite, with a fair amount of shear on both constituents. If quenching is done improperly, the internal stresses can cause a part to shatter as it cools. At the very least, they cause internal work hardening and other microscopic imperfections. It is common for quench cracks to form when water quenched, although they may not always be visible.

The steel industry in India is one of the major industries that govern our **GDP**.

SAIL or **Steel Authority of India Ltd** is a state run steel production plant that has a major stake in the steel market within the country as well as the outside world.

SAIL is India's largest steel producing company with a turnover of **Rs.48,681 crore** annually. The company is among the top five highest profit earning companies within India. **SAIL** has five integrated steel plants, 3 special steel plants and one subsidiary in different parts of the country.

The Salem steel plant is a special steels plant of **SAIL**. It has pioneered the supply of wider width stainless steel sheets / coils in India. The plant can produce Austenitic, Ferritic, Martensitic and low-nickel stainless steel in the form of coils and sheets with an installed capacity of 70,000 tonnes / year in Cold Rolling Mill and 1,86,000 tonnes / year in **Hot Rolling Mill**. In addition, the plant has country's first top-of-the-line stainless steel blanking facility with a capacity of 3,600 tonnes / year of coin blanks and utility blanks / circles.

The plant is facilitated with **Hot Rolling Mill** which can roll both stainless and carbon steels and the mill caters mainly to the input needs of stainless steel coils for the cold rolling mills. Special grades of carbon steels other than structural steels are also rolled from the facility includes weathering steels, high strength low alloy steels, etc., which are extensively used in industrial sectors.

1.2 Steel Manufacturing Process

Flow Chart

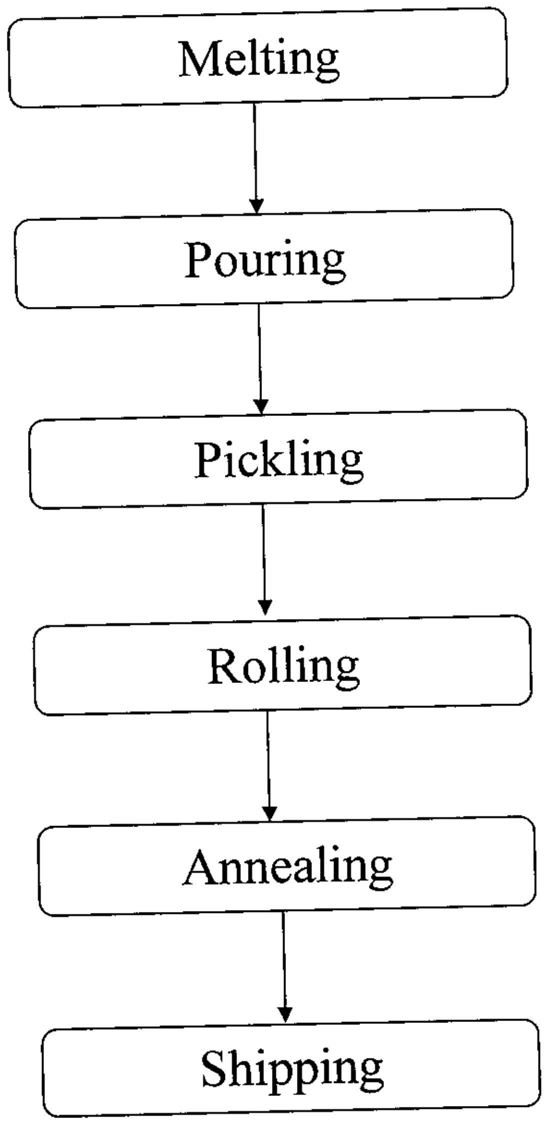


Fig. 1

Chapter – 2

2.1 Literature Review

[1]. One of the methods used where pretty cumbersome, it consisted of engaging a roller that possessed no automatic rollers, but a set of big presses, to make them into slabs. These methods were practiced during the early 1600

[2]. As the industry begin to progress, the need for sheets became important with the onset of the industrial revolution late 19th century and early 20th century. The automobiles heralded the need even at a greater pace. The method involved crude rolling with no direct impression on thickness, since proper measuring systems failed to exists.

[3]. The two world wars shocked the world with great industrial evolution and pace. The steel industry bludgeoned into a massive and decisive recognition for states to display their economic powers. During this period rolling improved with great hydraulic presses making the sheets, causing personnel discomfort to the working personnel. The presence of the extremely high temperature also discomforted them. Crude thickness measurement techniques as calipers were employed to measure.

[4]. The current system of process in **SAIL** for **Hot Roll Mill** is basically a **Batch Process** that means the sheet metal is produced in pre-determined batches of customer required specifications. The personnel physically calibrate the gap between the rollers and make the necessary adjustments as necessary before the process starts. Each time the slab passes under the roller, it is termed as a pass; the entire time taken for the sheet metal to be processed is 15-20 minutes. The roller comprises of 2 sets namely the top roller and the bottom roller. The total number of passes is 7

in number, it is staged in such a way that a slab of 220 mm thickness is slowly reduced and elongated into sheet of thickness 1.6 mm in the final or 7th stage/pass. The load applied on the sheet metal by the rollers is about 3M Newton. The speed of the rollers is 100-300 meters/sec. The roller diameter is 1040 mm and the length being 1425 mm. It must also be noted the existing temperature is around 1280 deg Celsius.

From the above detailed study, it was found that it was necessary to have a composite package that actually handles and overcomes certain core issues were identified. They were,

- The process is slow, since the calibration takes time apart from the pressing of the slab into sheet metal.
- The temperature in that zone is extremely high approx 1280 deg Celsius.
- Automation does not exist.
- If any variation takes place, the process has to be stopped entirely.

2.2 Objective

The objective of the project is to try and solve the above mentioned issues and automate the following steps in the process

- Automatic thickness calibration
- To control the pressure by either extending or retracting the top roller through automatic means

The proposal is to use the following items to achieve the task

- Stepper Motor Driver
- Ultrasonic Sensors
- PIC Micro-controller

Chapter – 3

3.1 Introduction to Hot Rolling Mill

A **rolling mill** is a machine that is used to shape and process metal by passing it between a pair of working rollers. The metal is passed between two tough cylinders numerous times, with the distance between the cylinders being decreased with each pass so the metal becomes thinner and thinner. Depending on the temperature of the metalworking application rolling mills are typically hot or cold rolling mills.

Hot rolling is a metalworking process that occurs above the recrystallization temperature of the material. After the grains deform during processing, they recrystallize, which maintains an equiaxed microstructure and preventing the metal from work hardening. The starting material is usually large pieces of metal, like semi-finished casting products, such as slabs, blooms, and billets. If these products came from a continuous casting operation the products are usually fed directly into the rolling mills at the proper temperature. In smaller operations the material starts are room temperature and must be heated. This is done in a gas- or oil-fired soaking pit for larger workpieces and for smaller workpieces induction heating is used. As the material is worked the temperature must be monitored to make sure it remains above the recrystallization temperature. To maintain a safety factor a *finishing temperature* is defined above the recrystallization temperature; this is usually 50 to 100 °C (122 to 212 °F) above the recrystallization temperature. If the temperature does drop below this temperature the material must be re-heated before hotter rolling.

Hot rolled metals generally have little directionality in their mechanical properties and deformation induced residual stresses. However, in certain instances non-metallic inclusions will impart some directionality and work pieces less than

20 mm (0.79 in) thick often have some directional properties. Also, non-uniformed cooling will induce a lot of residual stresses, which usually occurs in shapes that have a non-uniform cross-section, such as I-beams and H-beams. While the finished product is of good quality, the surface is covered in mill scale, which is an oxide that forms at high-temperatures. It is usually removed via pickling or the smooth clean surface process, which reveals a smooth surface. Dimensional tolerances are usually 2 to 5% of the overall dimension.

Hot rolling is used mainly to produce sheet metal or simple cross sections, such as rail tracks

The temperature approximately that is normally sensed is approximately 1280 deg Celsius in a hot rolling mill.

The processed and liquid metal is cast into slabs and then sent to the roughing mill, the slabs and transformed into sheets.

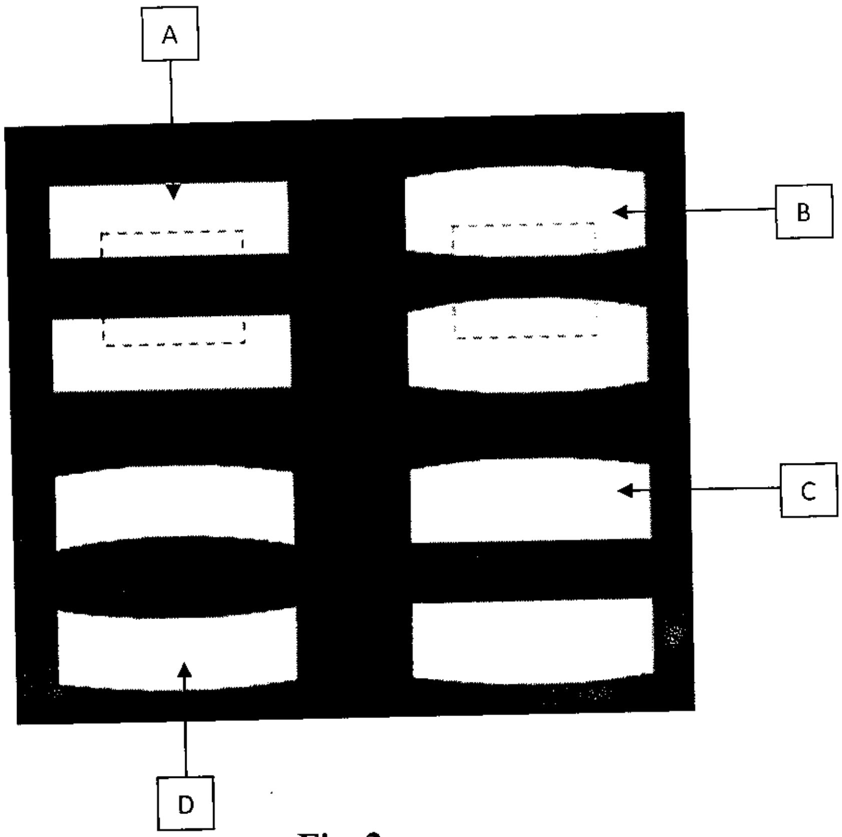


Fig. 2

The rollers A, B, C and D are the shapes of the rollers according to which the sheet metal will take shape.

3.2 General Sketch of the Rolling Mill

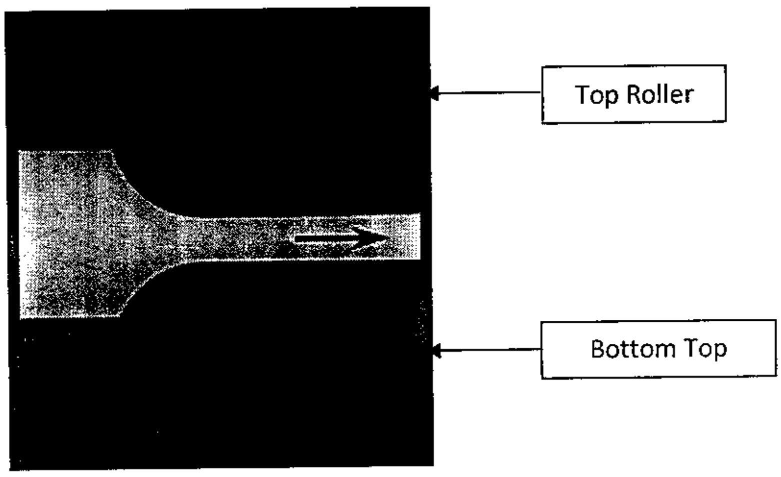


Fig. 3

The above diagram describes how the rolling mill works, where metal slabs are being transformed to sheets.

The top roller is the pressure roller; the bottom roller is the driving roller. The system allows pressure roller to convert the slab to sheets.

Chapter – 4

4.1 Introduction

Based on the purpose of this project a description regarding the platform being used PIC.

PIC – Peripheral Interface Controller

PIC is a family of Harvard architecture microcontrollers made by Microchip Technology, derived from the PIC1640 originally developed by General Instrument's Microelectronics Division. The name PIC initially referred to "**Programmable Interface Controller**".

The PIC architecture is distinctively minimalist. It is characterized by the following features:

- Separate code and data spaces (Harvard architecture)
- A small number of fixed length instructions
- Most instructions are single cycle execution (4 clock cycles), with single delay cycles upon branches and skips
- A single accumulator (W), the use of which (as source operand) is implied (i.e. is not encoded in the opcode)
- All RAM locations function as registers as both source and/or destination of math and other functions.
- A hardware stack for storing return addresses
- A fairly small amount of addressable data space (typically 256 bytes), extended through banking
- Data space mapped CPU, port, and peripheral registers

- The program counter is also mapped into the data space and writable (this is used to implement indirect jumps).

Unlike most other CPUs, there is no distinction between memory space and register space because the RAM serves the job of both memory and registers, and the RAM is usually just referred to as the register file or simply as the registers.

Data space (RAM)

PICs have a set of registers that function as general purpose RAM. Special purpose control registers for on-chip hardware resources are also mapped into the data space. The addressability of memory varies depending on device series, and all PIC devices have some banking mechanism to extend the addressing to additional memory. Later series of devices feature move instructions which can cover the whole addressable space, independent of the selected bank. In earlier devices (i.e., the baseline and mid-range cores), any register move had to be achieved via the accumulator.

To implement indirect addressing, a "file select register" (FSR) and "indirect register" (INDF) are used: A register number is written to the FSR, after which reads from or writes to INDF will actually be to or from the register pointed to by FSR. Later devices extended this concept with post- and pre- increment/decrement for greater efficiency in accessing sequentially stored data. This also allows FSR to be treated almost like a stack pointer (SP).

External data memory is not directly addressable except in some high pin count PIC18 devices.

Code space

All PICs feature Harvard architecture, so the code space and the data space are separate. PIC code space is generally implemented as EPROM, ROM, or flash ROM. In general, external code memory is not directly addressable due to the lack of an external memory interface. The exceptions are PIC17 and select high pin count PIC18 devices.

Word size

The word size of PICs can be a source of confusion. All PICs handle (and address) data in 8-bit chunks, so they should be called 8-bit microcontrollers. However, the unit of addressability of the code space is not generally the same as the data space. For example, PICs in the baseline and mid-range families have program memory addressable in the same wordsize as the instruction width, ie. 12 or 14 bits respectively. In contrast, in the PIC18 series, the program memory is addressed in 8-bit an increment (bytes), which differs from the instruction width of 16 bits. In order to be clear, the program memory capacity is usually stated in number of (single word) instructions, rather than in bytes.

Stacks

PICs have a hardware call stack, which is used to save return addresses. The hardware stack is not software accessible on earlier devices, but this changed with the 18 series devices. Hardware support for a general purpose parameter stack was lacking in early series, but this greatly improved in the 18 series, making the 18 series architecture more friendly to high level language compilers.

Instruction set

A PIC's instructions vary from about 35 instructions for the low-end PICs to over 80 instructions for the high-end PICs. The instruction set includes instructions to perform a variety of operations on registers directly, the accumulator and a literal constant or the accumulator and a register, as well as for conditional execution, and program branching. Some operations, such as bit setting and testing, can be performed on any numbered register, but bi-operand arithmetic operations always involve W; writing the result back to either W or the other operand register. To load a constant, it is necessary to load it into W before it can be moved into another register. On the older cores, all register moves needed to pass through W, but this changed on the "high end" cores. PIC cores have skip instructions which are used for conditional execution and branching. The skip instructions are: 'skip if bit set', and, 'skip if bit not set'. Because cores before PIC18 had only unconditional branch instructions, conditional jumps are implemented by a conditional skip (with the opposite condition) followed by an unconditional branch. Skips are also of utility for conditional execution of any immediate single following instruction.

The PIC architecture has no (or very meager) hardware support for automatically saving processor state when servicing interrupts. The 18 series improved this situation by implementing shadow registers which save several important registers during an interrupt.

In general, PIC instructions fall into 5 classes:

1. Operation on W with 8-bit immediate ("literal") operand. E.g. movlw (move literal to W), andlw (AND literal with W). One instruction peculiar to the PIC is retlw, load immediate into W and return, which is used with computed branches to produce lookup tables.
2. Operation with W and indexed register. The result can be written to either the W register (e.g. addwf *reg*,w). or the selected register (e.g. addwf *reg*,f).
3. Bit operations. These take a register number and a bit number, and perform one of 4 actions: set or clear a bit, and test and skip on set/clear. The latter are used to perform conditional branches. The usual ALU status flags are available in a numbered register so operations such as "branch on carry clear" are possible.
4. Control transfers. Other than the skip instructions previously mentioned, there are only two: goto and call.
5. A few miscellaneous zero-operand instructions, such as return from subroutine, and sleep to enter low-power mode.

Performance

Many of these architectural decisions are directed at the maximization of top-end speed, or more precisely of speed-to-cost ratio. The PIC architecture was among the first scalar CPU designs, and is still among the simplest and cheapest. The Harvard architecture—in which instructions and data come from conveniently separate sources—simplifies timing and microcircuit design greatly, and this pays benefits in areas like clock speed, price, and power consumption.

The PIC is particularly suited to implementation of fast lookup tables in the program space. Such lookups are 0 (1) and can complete via a single instruction taking two instruction cycles. Basically any function can be modeled in this way. Such optimization is facilitated by the relatively large program space of the PIC (e.g. 4096 x 14-bit words on the 16F690) and by the design of the instruction set, which allows for embedded constants.

The simplicity of the PIC, and its scalar nature, also serve to greatly simplify the construction of real-time code. It is typically possible to multiply the line count of a PIC assembler listing by the instruction cycle time to determine execution time. (This is true because skip-based instructions take 2 cycles whether the skip occurs or doesn't.) On other CPUs (even the Atmel, with its MUL instruction), such quick methods are just not possible. In low-level development, precise timing is often critical to the success of the application, and the real-time features of the PIC can save crucial engineering time.

A similarly useful and unique property of PICs is that their interrupt latency is constant (it's also low: 3 instruction cycles). The delay is constant even though instructions can take one or two instruction cycles: a dead cycle is optionally inserted into the interrupt response sequence to make this true. External interrupts have to be synchronized with the four clock instruction cycle; otherwise there can be a one instruction cycle jitter. Internal interrupts are already synchronized.

The constant interrupt latency allows PICs to achieve interrupt driven low jitter timing sequences. An example of this is a video sync pulse generator. Other microcontrollers can do this in some cases, but it's awkward. The non-interrupt code has to anticipate the interrupt and enter into a sleep state before it arrives. On PICs, there is no need for this.

The three-cycle latency is increased in practice because the PIC does not store its registers when entering the interrupt routine. Typically, 4 instructions are needed to store the W-register, the status register and switch to a specific bank before starting the actual interrupt processing.

Limitations

The PIC architectures have several limitations:

- Only a single accumulator
- A small instruction set
- Operations and registers are not orthogonal; some instructions can address RAM and/or immediate constants, while others can only use the accumulator
- Memory must be directly referenced in arithmetic and logic operations, although indirect addressing is available via 2 additional registers
- Register-bank switching is required to access the entire RAM of many devices

The following limitations have been addressed in the PIC18, but still apply to earlier cores:

- Conditional skip instructions are used instead of conditional jump instructions used by most other architectures
- Indexed addressing mode is very rudimentary

- Stack:
    - The hardware call stack is so small that program structure must often be flattened
    - The hardware call stack is not addressable, so pre-emptive task switching cannot be implemented
    - Software-implemented stacks are not efficient, so it is difficult to generate reentrant code and support local variables
- Program memory is not directly addressable, and thus space-inefficient and/or time-consuming to access. (This is true of most Harvard architecture microcontrollers.)

With paged program memory, there are two page sizes to worry about: one for CALL and GOTO and another for computed GOTO (typically used for table lookups). For example, on PIC16, CALL and GOTO have 11 bits of addressing, so the page size is 2048 instruction words. For computed GOTOs, where you add to PCL, the page size is 256 instruction words. In both cases, the upper address bits are provided by the PCLATH register. This register must be changed every time control transfers between pages. PCLATH must also be preserved by any interrupt handler.

Compiler development

These properties have made it difficult to develop compilers that target PIC microcontrollers. While several commercial compilers are available, in 2008, Microchip finally released their C compilers, C18, and C30 for their line of 18f 24f and 30/33f processors. By contrast, Atmel's AVR microcontrollers—which are competitive with PIC in terms of hardware capabilities and price, but feature a RISC instruction set—have long been supported by the GNU C Compiler.

Also, because of these properties, PIC assembly language code can be difficult to comprehend. Judicious use of simple macros can make PIC assembly language much more palatable, but at the cost of a reduction in performance. For example, the original Parallax PIC assembler ("SPASM") has macros which hide W and make the PIC look like a two-address machine. It has macro instructions like "`mov b,a`" (move the data from address *a* to address *b*) and "`add b,a`" (add data from address *a* to data in address *b*). It also hides the skip instructions by providing three operand branch macro instructions such as "`cjne a,b,dest`" (compare *a* with *b* and jump to *dest* if they are not equal).

Family core architectural differences

Baseline Core devices

These devices feature a 12-bit wide code memory, a 32-byte register file, and a tiny two level deep call stack. They are represented by the PIC10 series, as well as by some PIC12 and PIC16 devices. Baseline devices are available in 6-pin to 40-pin packages.

Generally the first 7 to 9 bytes of the register file are special-purpose registers, and the remaining bytes are general purpose RAM. If banked RAM is implemented, the bank number is selected by the high 3 bits of the FSR. This affects register numbers 16–31; registers 0–15 are global and not affected by the bank select bits.

The ROM address space is 512 words (12 bits each), which may be extended to 2048 words by banking. CALL and GOTO instructions specify the low 9 bits of the new code location; additional high-order bits are taken from the staus register. Note that a CALL instruction only includes 8 bits of address, and may only specify addresses in the first half of each 512-word page.

The instruction set is as follows. Register numbers are referred to as "f", while constants are referred to as "k". Bit numbers (0–7) are selected by "b". The "d" bit selects the destination: 0 indicates W, while 1 indicates that the result is written back to source register f.

PIC Instruction Set

- MOVLW
- MOVWF
- MOVF
- CLRW
- CLRF
- SWAPF
- ADDLW
- ADDWF
- SUBLW
- SUBWF
- ANDLW
- ANDWF
- IORLW
- IORWF
- XORLW
- XORWF
- INCF
- DECF
- RLF
- RRF
- COMF
- BCF
- BSF
- BTFSC

- BTFSS
- INCFSZ
- DECFSZ
- GOTO
- CALL
- RETURN
- RETLW
- RETFIE
- NOP
- CLRWDT
- SLEEP

Features of 18 Series

The 18 series inherits most of the features and instructions of the 17 series, while adding a number of important new features:

- much deeper call stack (31 levels deep)
- the call stack may be read and written
- conditional branch instructions
- indexed addressing mode (PLUSW)
- extending the FSR registers to 12 bits, allowing them to linearly address the entire data address space
- the addition of another FSR register (bringing the number up to 3)

The auto increment/decrement feature was improved by removing the control bits and adding four new indirect registers per FSR. Depending on which indirect file register is being accessed it is possible to postdecrement, postincrement, or preincrement FSR; or form the effective address by adding W to FSR.

In more advanced PIC18 devices, an "extended mode" is available which makes the addressing even more favorable to compiled code:

- a new offset addressing mode; some addresses which were relative to the access bank are now interpreted relative to the FSR2 register
- The addition of several new instructions, notable for manipulating the FSR registers.

These changes were primarily aimed at improving the efficiency of a data stack implementation. If FSR2 is used either as the stack pointer or frame pointer, stack items may be easily indexed—allowing more efficient re-entrant code. Microchip C18 chooses to use FSR2 as a frame pointer.

PIC 18F458 Pin Diagram

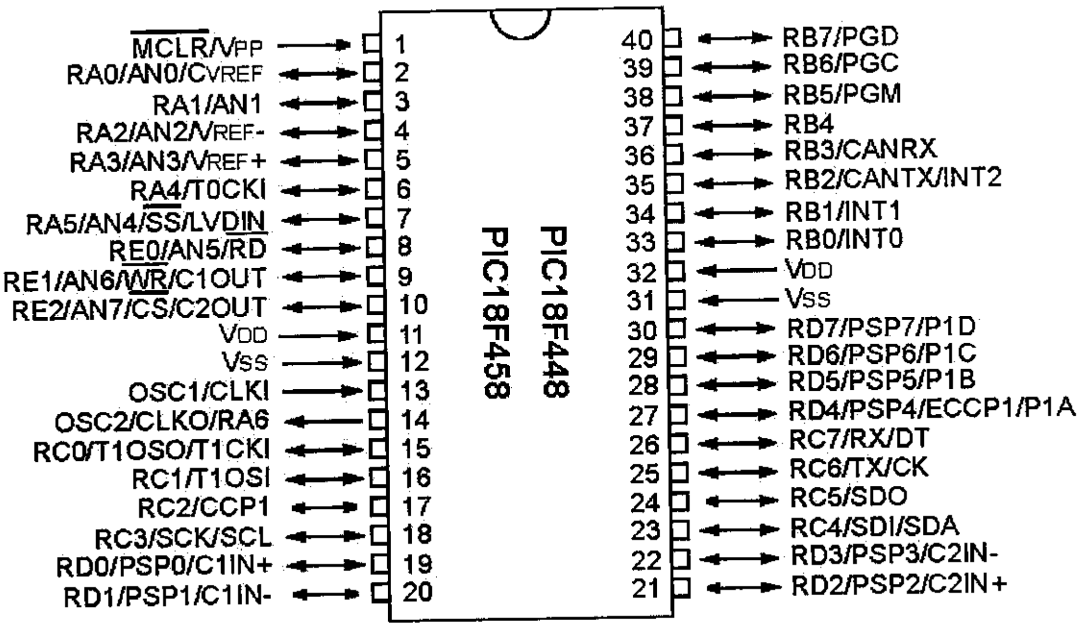


Fig. 4

PIC Architecture

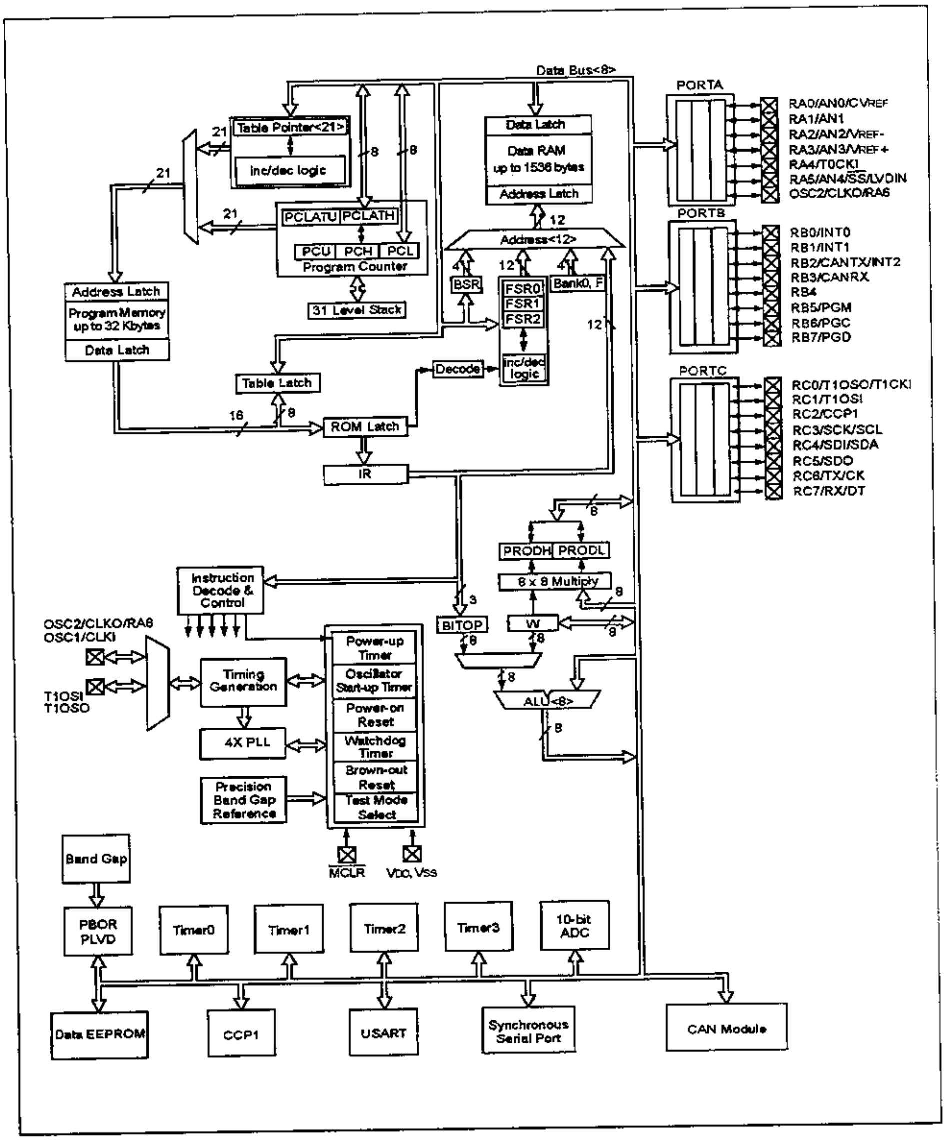


Fig. 5

Chapter – 5

Sheet Metal Thickness Controller

5.1 Introduction

The sheet metal thickness controller as discussed earlier aims to maneuver along the steps described. It aims to include a touch automation and increased accuracy along the way.

The project utilizes 5 sets of interface boards namely.

- Ultrasonic sensor interface board.
- LCD interface board.
- Stepper motor interface board.
- Power supply board.
- PIC – as explained in previous pages.

The individual descriptions of each are followed.

5.2 Overall Block Diagram

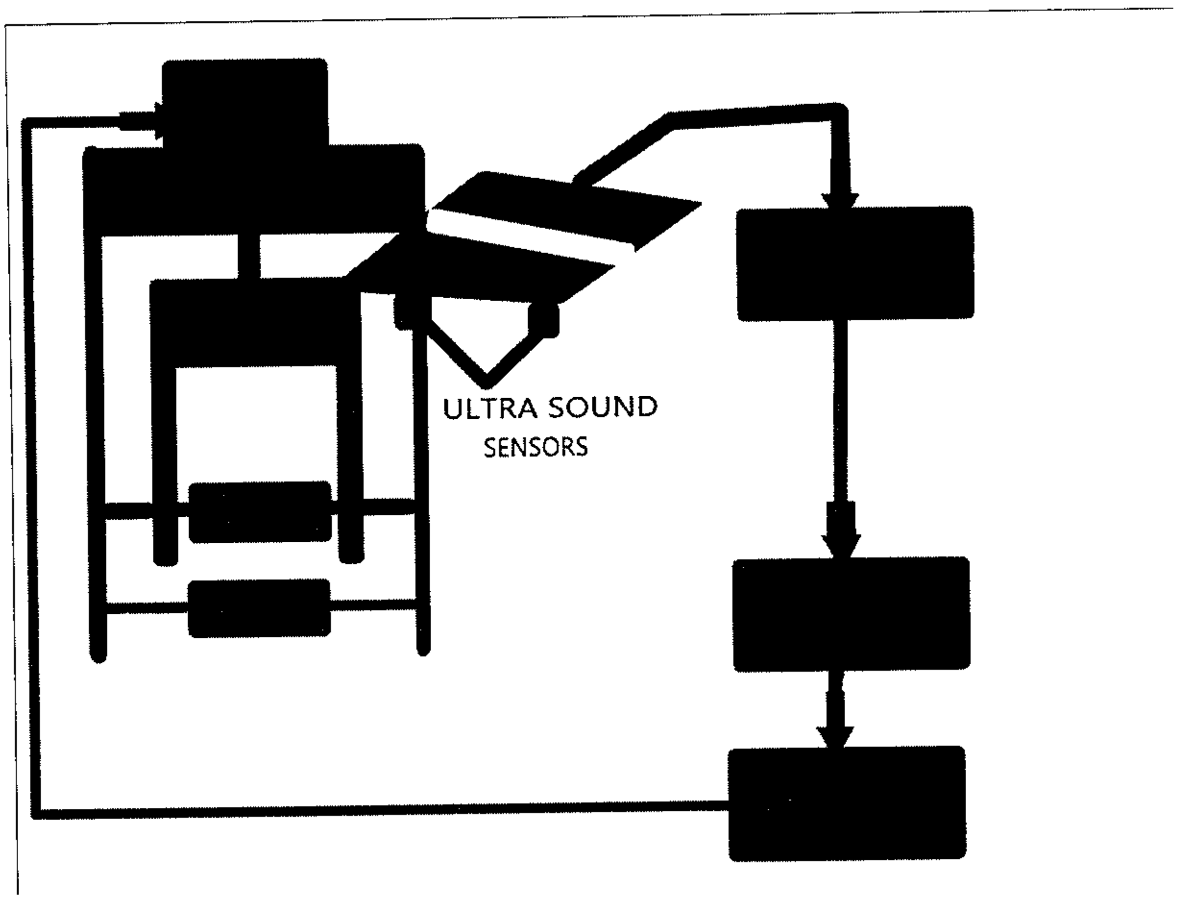


Fig. 6

5.3 Description

5.3.1 Mechanical Assembly

The mechanical assembly consists of 2 sets of rollers

- Top roller.
- Bottom roller- which acts as the driving roller.

The top roller is connected to the stepper motor by means of a screw rod, as seen in the diagram. The rotational motion of the stepper motor is converted to linear motion by this means. This linear motion causes the top roller to move up and down respectively creating the pressure for the slab to elongate into sheet metal.

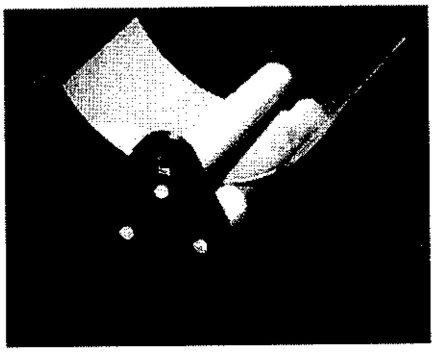


Fig. 7

5.3.2 Ultrasonic Sensors

5.3.2.1 Principle

Ultrasonic sensors (also known as **transceivers** when they both send and receive) work on a principle similar to radar or sonar which evaluate attributes of a target by interpreting the echoes from radio or sound waves respectively. Ultrasonic sensors generate high frequency sound waves and evaluate the echo which is received back by the sensor. Sensors calculate the time interval between sending the signal and receiving the echo to determine the distance to an object.

This technology can be used for measuring: wind speed and direction (anemometer), fullness of a tank and speed through air or water. For measuring speed or direction a device uses multiple detectors and calculates the speed from the relative distances to particulates in the air or water. To measure the amount of liquid in a tank, the sensor measures the distance to the surface of the fluid. Further applications include: humidifiers, sonar, medical ultrasonography, burglar alarms and non-destructive testing.

Systems typically use a transducer which generates sound waves in the ultrasonic range, above 20,000 hertz, by turning electrical energy into sound, then upon receiving the echo turn the sound waves into electrical energy which can be measured and displayed.

The technology is limited by the shapes of surfaces and the density or consistency of the material. For example foam on the surface of a fluid in a tank could distort a reading.

5.3.2.2 Working

Transducers

An ultrasonic transducer is a device that converts energy into ultrasound, or sound waves above the normal range of human hearing. While technically a dog whistle is an ultrasonic transducer that converts mechanical energy in the form of air pressure into ultrasonic sound waves, the term is more apt to be used to refer to piezoelectric transducers that convert electrical energy into sound. Piezoelectric crystals have the property of changing size when a voltage is applied, thus applying an alternating current (AC) across them causes them to oscillate at very high frequencies, thus producing very high frequency sound waves.

The location, at which a transducer focuses the sound, can be determined by the active transducer area and shape, the ultrasound frequency and the sound velocity of the propagation medium.

Detector

Since piezoelectric crystals generate a voltage when force is applied to them, the same crystal can be used as an ultrasonic detector. Some systems use separate transmitter and receiver components while others combine both in a single piezoelectric transceiver.

Alternative methods for creating and detecting ultrasound include magnetostriction and capacitive actuation.

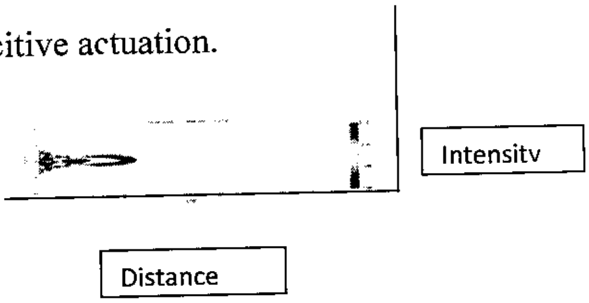


Fig. 8

The ultrasonic sensors used here are SRF04 module; they consist of a set of transmitter and receiver that sends out 8 bursts of 40 KHz pulses. When the pulses are triggered a timer is switched on, when any one of the pulses is echoed back to the receiver, then timer stops, the time difference between the timer trigger on and back is determined by the following formula

$$3.627*8*200\text{ns}=5.8\text{ns for 1mm}$$

Where 3.627 is the pulse width, 8 being the number of bursts and 200ns is the time delay, which translates to 5.8 ns for 1mm.

Schematic for Ultrasonic Sensor interface board

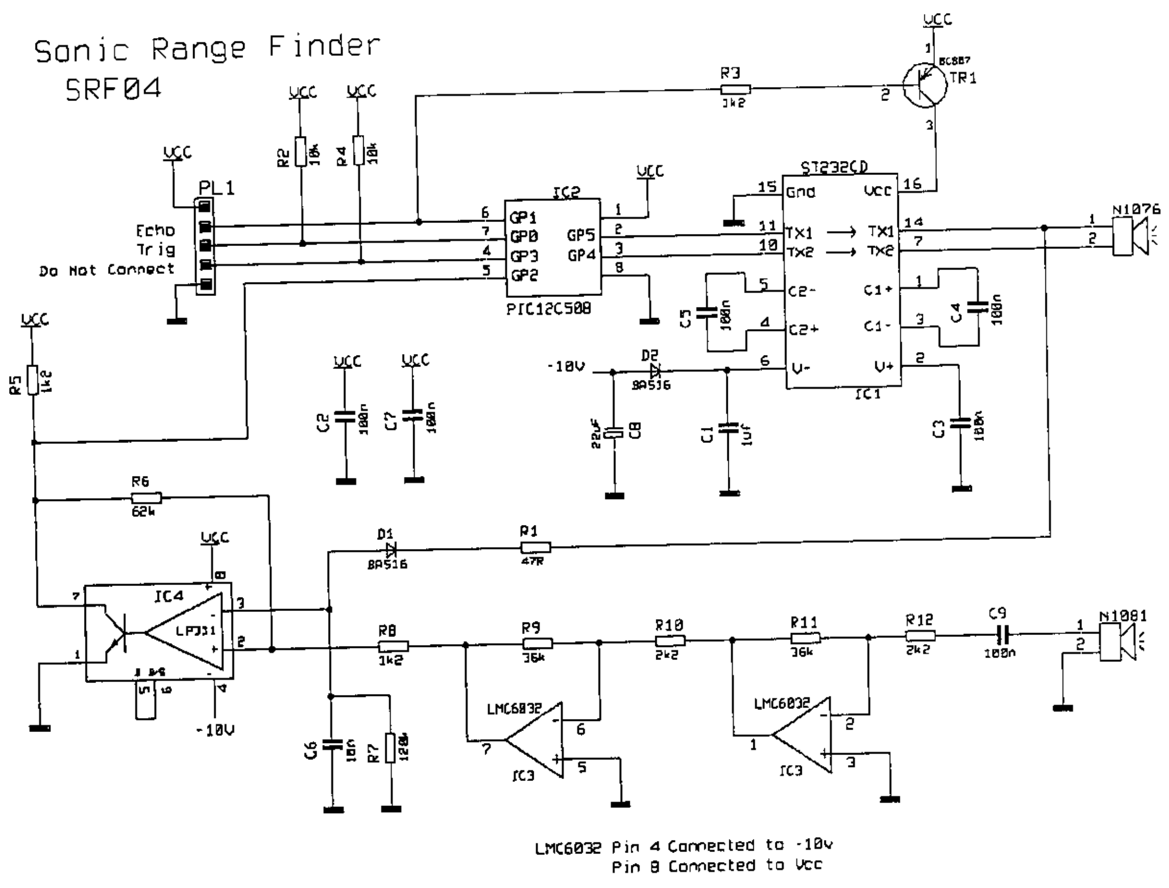


Fig. 9

5.3.3 LCD – Liquid Crystal Display

5.3.3.1 Principle

A **liquid crystal display** (LCD) is a thin, flat panel used for electronically displaying information such as text, images, and moving pictures. Its uses include monitors for computers, televisions, instrument panels, and other devices ranging from aircraft cockpit displays, to every-day consumer devices such as video players, gaming devices, clocks, watches, calculators, and telephones. Among its major features are its lightweight construction, its portability, and its ability to be produced in much larger screen sizes than are practical for the construction of cathode ray tube (CRT) display technology. Its low electrical power consumption enables it to be used in battery-powered electronic equipment. It is an electronically-modulated optical device made up of any number of pixels filled with liquid crystals and arrayed in front of a light source (backlight) or reflector to produce images in colour or monochrome.

5.3.3.2 Working

Each pixel of an LCD typically consists of a layer of molecules aligned between two transparent electrodes, and two polarizing filters, the axes of transmission of which are (in most of the cases) perpendicular to each other. With no actual liquid crystal between the polarizing filters, light passing through the first filter would be blocked by the second (crossed) polarizer.

The surfaces of the electrodes that are in contact with the liquid crystal material are treated so as to align the liquid crystal molecules in a particular direction. This treatment typically consists of a thin polymer layer that is unidirectionally rubbed using, for example, a cloth. The direction of the liquid crystal alignment is then defined by the direction of rubbing. Electrodes are made of a transparent conductor called Indium Tin Oxide (ITO).

Before applying an electric field, the orientation of the liquid crystal molecules is determined by the alignment at the surfaces of electrodes. In a twisted nematic device (still the most common liquid crystal device), the surface alignment directions at the two electrodes are perpendicular to each other, and so the molecules arrange themselves in a helical structure, or twist. This reduces the rotation of the polarization of the incident light, and the device appears grey. If the applied voltage is large enough, the liquid crystal molecules in the center of the layer are almost completely untwisted and the polarization of the incident light is not rotated as it passes through the liquid crystal layer. This light will then be mainly polarized perpendicular to the second filter, and thus be blocked and the pixel will appear black. By controlling the voltage applied across the liquid crystal layer in each pixel, light can be allowed to pass through in varying amounts thus constituting different levels of gray. The optical effect of a twisted nematic device in the voltage-on state is far less dependent on variations in the device thickness than that in the voltage-off state. Because of this, these devices are usually operated between crossed polarizers such that they appear bright with no voltage (the eye is much more sensitive to variations in the dark state than the bright state). These devices can also be operated between parallel polarizers, in which case the bright and dark states are reversed. The voltage-off dark state in this configuration appears blotchy, however, because of small variations of thickness across the device.

Both the liquid crystal material and the alignment layer material contain ionic compounds. If an electric field of one particular polarity is applied for a long period of time, this ionic material is attracted to the surfaces and degrades the device performance. This is avoided either by applying an alternating current or by reversing the polarity of the electric field as the device is addressed (the response of the liquid crystal layer is identical, regardless of the polarity of the applied field).

When a large number of pixels are needed in a display, it is not technically possible to drive each directly since then each pixel would require independent electrodes. Instead, the display is *multiplexed*. In a multiplexed display, electrodes on one side of the display are grouped and wired together (typically in columns), and each group gets its own voltage source. On the other side, the electrodes are also grouped (typically in rows), with each group getting a voltage sink. The groups are designed so each pixel has a unique, unshared combination of source and sink. The electronics or the software driving the electronics then turns on sinks in sequence, and drives sources for the pixels of each sink.

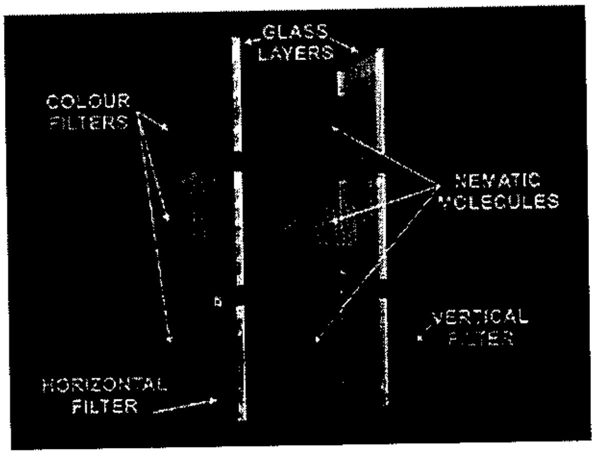


Fig. 10

The above is an image of the LCD in working.

There are many types of LCD formats such as,

- 16 x 1LCD
- 16 x 2 LCD
- 20 x 4LCD

In this project the 16 x 2 LCD format will be ideal for use.

LCD Schematic Diagram of the interface board

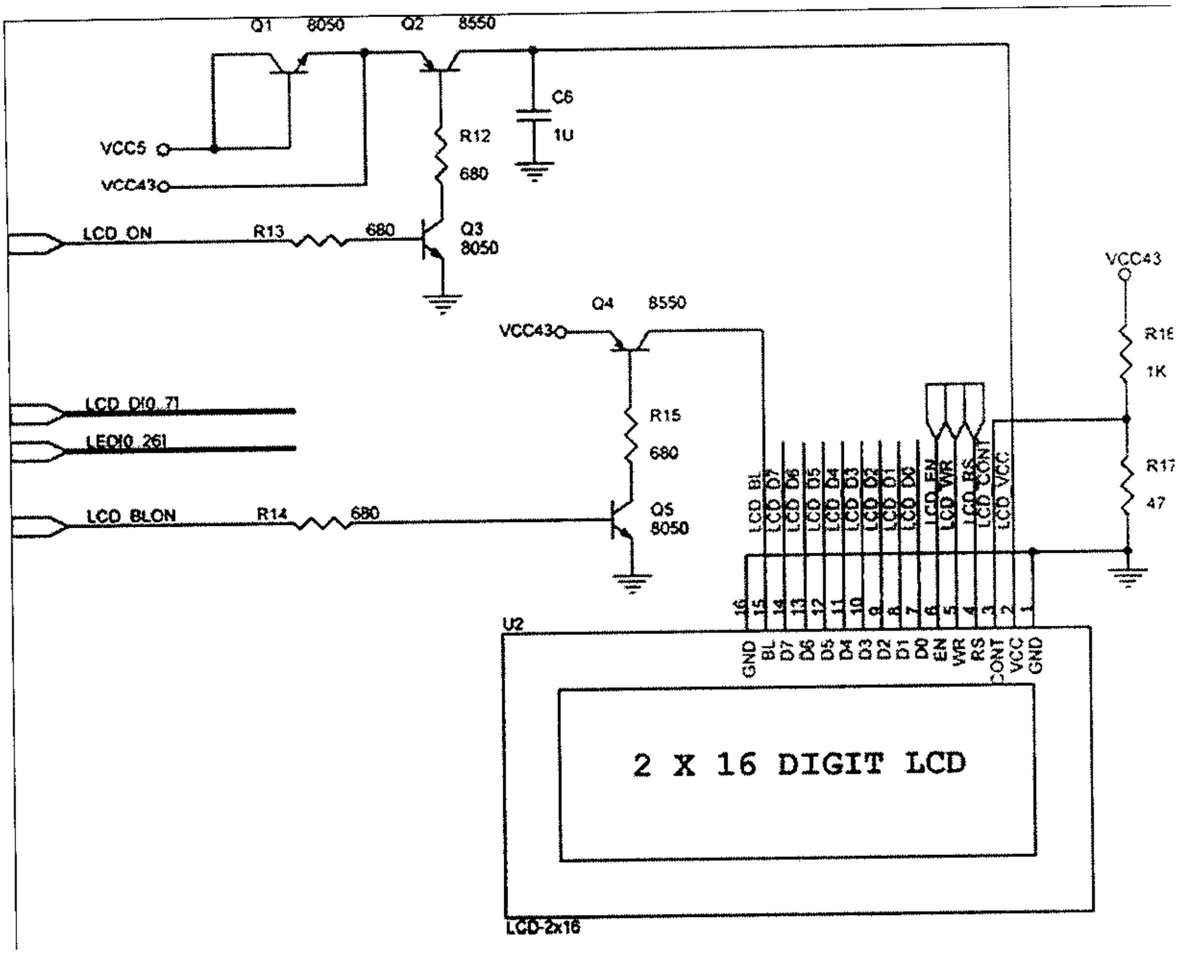


Fig. 11

5.3.4 Stepper Motor

5.3.4.1 Principle

A **stepper motor** (or **step motor**) is a brushless, synchronous electric motor that can divide a full rotation into a large number of steps. The motor's position can be controlled precisely without any feedback mechanism (see Open-loop controller), as long as the motor is carefully sized to the application. Stepper motors are similar to switched reluctance motors (which are very large stepping motors with a reduced pole count, and generally are closed-loop commutated.)

1. Stepper motors are constant power devices.
2. As motor speed increases, torque decreases. (most motors exhibit maximum torque when stationary, however the torque of a motor when stationary is of little use, torque is more important when the motor is actually spinning)
3. The torque curve may be extended by using current limiting drivers and increasing the driving voltage (sometimes referred to as a 'chopper' circuit, there are several off the shelf driver chips capable of doing this in a simple manner).
4. Steppers exhibit more vibration than other motor types, as the discrete step tends to snap the rotor from one position to another, (this is important as at certain speeds the motor can actually change direction).
5. This vibration can become very bad at some speeds and can cause the motor to lose torque (or lose direction). You must pay attention to the torque of the motor)
6. The effect can be mitigated by accelerating quickly through the problem speeds range, physically damping (frictional damping) the system, or using a micro-stepping driver.

7. Motors with a greater number of phases also exhibit smoother operation than those with fewer phases (this can also be achieved through the use of a micro stepping drive)

5.3.4.2 Working

Stepper motors operate differently from DC brush motors, which rotate when voltage is applied to their terminals. Stepper motors, on the other hand, effectively have multiple "toothed" electromagnets arranged around a central gear-shaped piece of iron. The electromagnets are energized by an external control circuit, such as a microcontroller. To make the motor shaft turn, first one electromagnet is given power, which makes the gear's teeth magnetically attracted to the electromagnet's teeth. When the gear's teeth are thus aligned to the first electromagnet, they are slightly offset from the next electromagnet. So, when the next electromagnet is turned on and the first is turned off, the gear rotates slightly to align with the next one, and from there the process is repeated. Each of those slight rotations is called a "step," with an integer number of steps making a full rotation. In that way, the motor can be turned by a precise angle.

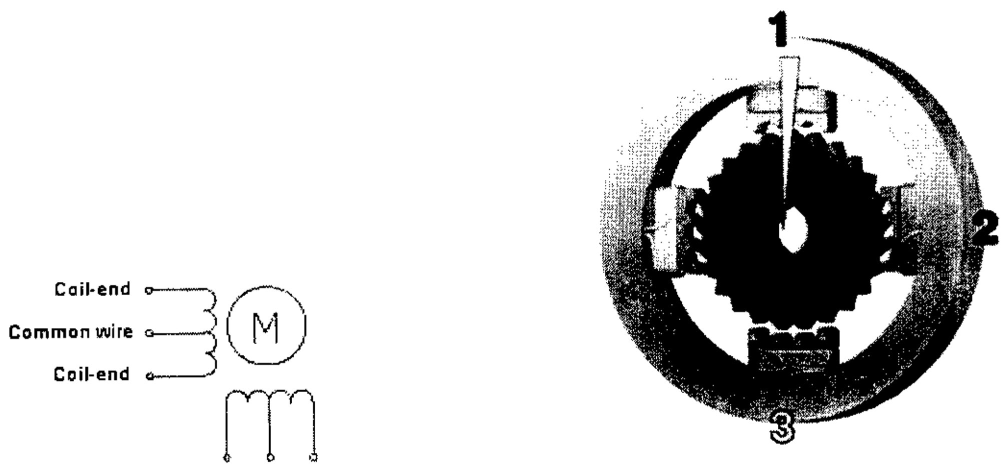


Fig. 12

The specifications that are used for this particular project is

- Uni-polar
- 0.9 deg step angle
- Rated current is 0.8 Amps
- Resistance id 2.4 Ohms

The algorithm is programmed in such a way that the following formula is applied.

155 counts produces 5mm variations either direction, i.e. up or down.

The next diagram describes the stepper motor used in this project.

Stepper Motor Schematic Diagram for its interface board

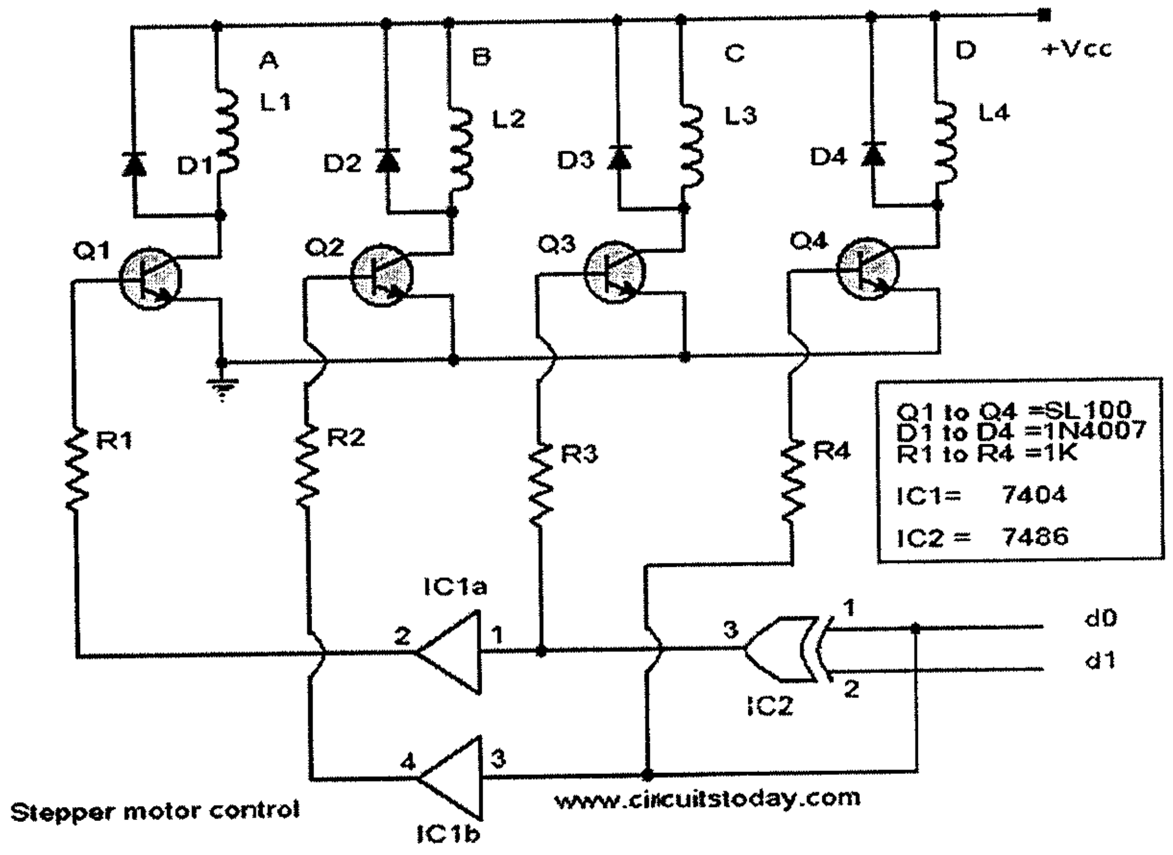


Fig.13

5.3.5 Power Circuit

The power for this project is driven by the individual power packs of 5V for the PIC and the ultrasonic sensors, 12V for the stepper motor.

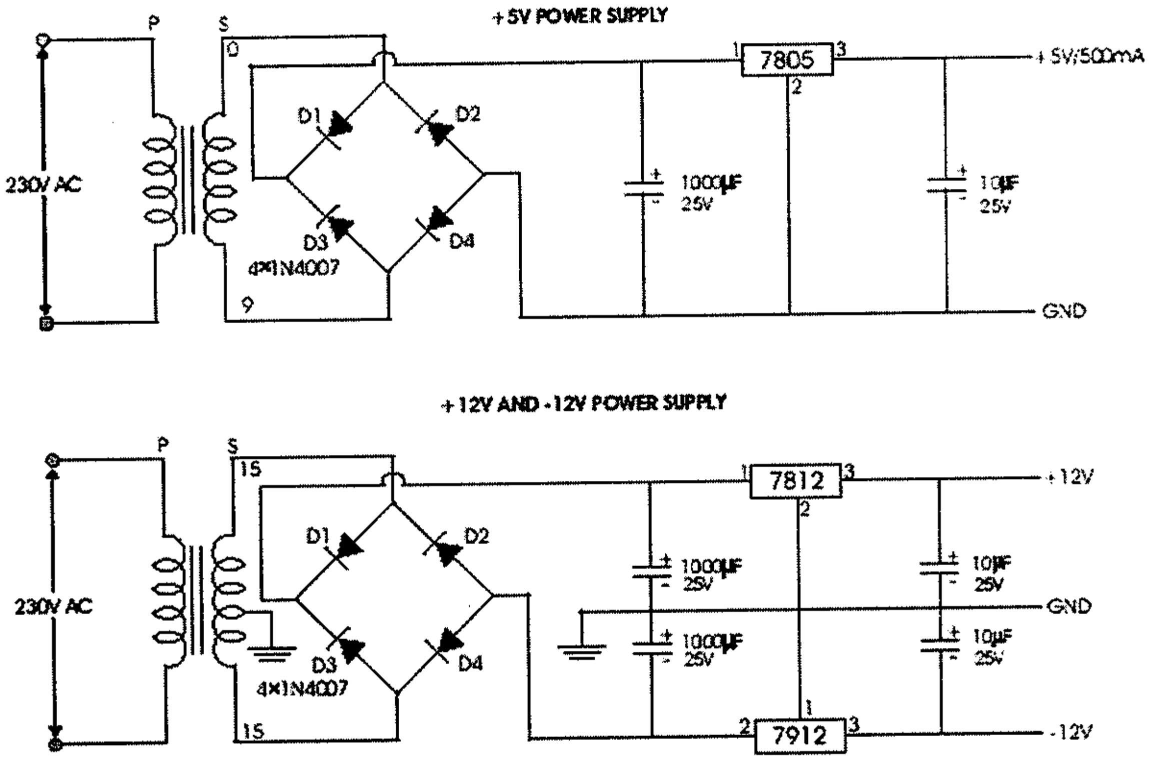


Fig. 14

Chapter – 6

6. Overall Circuit Diagram

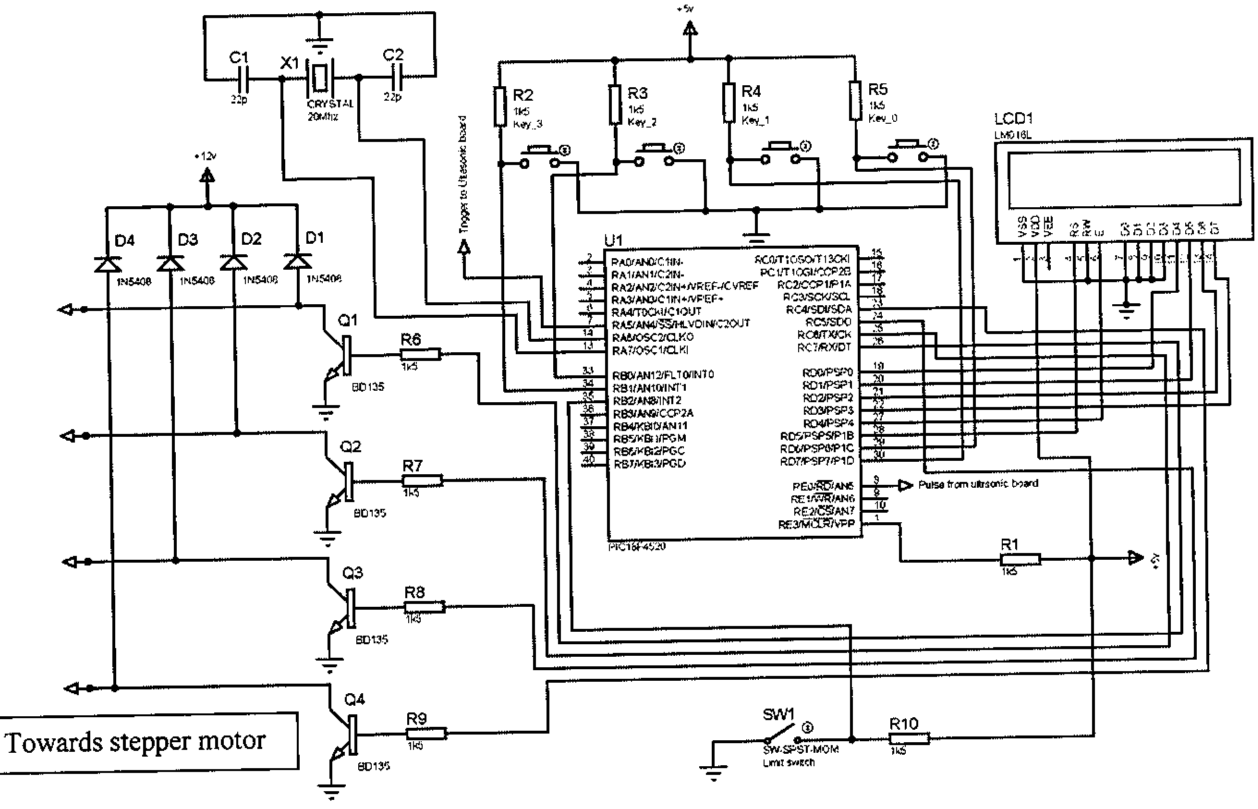


Fig. 15

Chapter – 7

7. Flow of Process

For us the process begins from the furnace to a dye cast where the molten liquid metal is converted to a slab of thickness 220 mm, still retaining the semi-hard-semi-molten state.

The conveyor transfers this slab from the furnace to the roller section called the Roughing Mill. Here the gap between the rollers is pre-determined to the final thickness, based on the following steps

Example: If final thickness required is say 2 mm, then

Stage 1 press 220 mm → 190 mm

Stage 2 press 190 mm → 160 mm

Stage 3 press 160 mm → 130 mm

Stage 4 press 130 mm→ 110 mm

Stage 5 press 100 mm → 70 mm

Stage 6 press 70 mm → 40 mm

Stage 7 press 40 mm → 2 mm

These stages are converted by decreasing the roller gap by 30 mm every single pass.

The entire setup is controlled by a PIC micro-controller. As each time the slab passes out of the rollers a set of ultrasonic sensors, provides necessary feedback regarding the thickness, by sending out short bursts of pulses at 40 kHz, the response time indicates thickness. This feedback is sent to the PIC, which correspondingly gives instructions for the stepper motor to move up and down the top roller thus, providing the necessary pressure for producing sheet metal.

Conclusion

Conclusion

The objective of this project which is to control and automate the sheet metal thickness is achieved with high degree of accuracy. This technique is used in metal working and production, automobile and aerospace industries.

To further improve the accuracy of sheet metal thickness, the use of LASER will pave a path for further development. The control aspect of the system may further be developed using PID algorithms to minimize the roller deflections.

Appendices

Appendix I

THICKNESS

The following is the data collected after the testing the mechanism.

Formula Used:

3.627 * 8 Pulses * 200 ns (time period) = 5.8 ns for 1 mm of thickness.

Time (ns)	Thickness (mm)
5.8	1
11.6	2
17.4	3
23.2	4
29	5
34.8	6
40.6	7
46.4	8
52.2	9
58	10

Table 1

Stepper Motor Counts

The following data is received after testing.

Counts	**Distance (mm)**
31	1
62	2
93	3
124	4
155	5
186	6
217	7
248	8
279	9
310	10

Table 2

Appendix II

PIC Code

The following is the PIC that is loaded into the micro-controller.

```
#include <pic18.h>


#define      trigger          RA5
#define pulse              RE0
#define rs                 RD5
#define en                 RD4
#define mode               RB5

#define key_0              RD6
#define key_1              RD7
#define key_2              RB0
#define key_3              RB1

#define key_4              RB2
#define key_5              RB4

#define key_set            RD6
#define key_next           RD7
```

```
#define key_pre          RB0
#define key_inc          RB1
#define key_cur          RB2
#define key_esc          RB4

#define coil1            RC4
#define coil2            RC5
#define coil3            RC6
#define coil4            RC7
#define limit_sw         RB2

void set_mode_data();
void tmr1_int();
void delay(int a);
void lcdadrr(int adrr);
void lcddata(int data);
void lcdint();
void portint();
void lcdadrr1(int adrr);
void enable();
void read_data();
void setting_menu();
```

```
	void set_esc();

	void clr_dis(int adrr);

	void slave_function();

	void menu_slave_dis(int dis);

	void next_pre_set_esc();

	void master_function();

	void menu_master_dis(int dis1);

	void split();

	void write();

	void read();

	void inc_dec_3();

const unsigned char head[]={"Sheet metal ctrl"}     //int he1=0,he2=10;


const unsigned char set_mode[]={"set.. value=00mm"};



bank1 unsigned long int sub_content,current_split,count1,content;  //split function

bank3 unsigned int seg0,seg,seg1,seg2,seg3,seg4,seg5;

bank2 float distancef;

bank1 unsigned long int temp_3;

unsigned int
count,ii,data1,data2,data,channel,data3,master_password,slave_password,location,t
imeout,sec,change,menu,main_menu,sub_menu,array_1,array_2,getin1,adrr,getin;

bank1 unsigned int
motor_count,temp_1,temp_2,temp_time1,temp_time2,temp_time3,temp_time4;
```

```
bank1 bit s1,s2,s3,s4,s5,s6,s7,s8,s9,s10;


bank1 unsigned int
sample1,sample2,sample3,sample,distance,motor,getout,set_value;


        void  main()
        {
        portint();
        adrr=123;
        read();
        set_value=data;
        lcdint();
        T1CON=0X30;
        TMR1L=0;
        TMR1H=0;
 if(key_0==1)
 {
 delay(1);
         if(key_0==1)
         {
                 while(key_0==1);
                 set_mode_data();
        }}
while(key_3==0)
{
coil1=coil2=coil3=coil4=0;
coil1=1;
delay(5);
coil1=coil2=coil3=coil4=0;
coil2=1;
delay(5);
coil1=coil2=coil3=coil4=0;
coil3=1;
delay(5);
coil1=coil2=coil3=coil4=0;
coil4=1;
delay(5);
count++;
digit_split(count,0xc0,5,0);                //155 count for 5mm
}
coil1=coil2=coil3=coil4=0;
```

```
motor_count=0;
        while(1)
        {


//-----------------------
// front
//-----------------------
        if(key_2==1)
        {
                delay(1);
                if(key_2==1)
                {
 lcdadrr1(0Xcf);
 lcddata('0');
 while(1)
 {
        delay(10);   //5ms
        trigger=1;
        delay(5);
        trigger=0;
        while(pulse==0);          //pre check
```

```
    TMR1ON=1;
    msbb=0;
    while(pulse==1);          //return pulse
    TMR1ON=0;
    content=TMR1H;
    content=content*256;
    content=content+TMR1L;        //convert into 16bit
//distancef=content/36.27;        //34.4;
//3.627*8*200ns=5.8ns for 1mm
    distancef=content/3.627;      //34.4;
    currentdis1=distancef;
    digit_split(content,0xc0,5,0);
    sample++;
if(sample==1)sample1=currentdis1;
if(sample==2)sample2=currentdis1;
/*if(sample==3)sample3=currentdis1; */
if(sample==2)
{
sample=0;
     if(sample1==sample2)
     {
//            if(sample2==sample3)
```

```
                {
                distance=sample2;
                }
/*              else {sample1=sample2=sample3=0;}*/
        }
else{sample1=sample2=sample3=0;}
}

//      digit_split(content,0xc0,5,0);
        lcdadrr1(0X80);
        lcddata('T');
        lcddata('H');
        lcddata('I');
        lcddata('C');
        lcddata('K');
        lcddata('N');
        lcddata('E');
        lcddata('S');
        lcddata('S');
        lcddata('=');
        content=distance;
        digit_split(content,0x8A,3,0);
        lcddata('m');
        lcddata('m');
        lcddata(' ');
        TMR1ON=0;
        TMR1L=0;
        TMR1H=0;
motor=0;
if(distance>=(set_value+2))
{
motor='f';
motor_drive();
}
if(distance<=(set_value-2))
{
motor='r';
motor_drive();
}
lcdadrr1(0Xc9);
if(motor==0) lcddata('T');
if(motor=='f') lcddata('f');
if(motor=='r') lcddata('r');
          }
        }
}
}
}
void motor_drive()
{
if(key_3==0)
{
        if(motor=='r')
        {
        coil1=coil2=coil3=coil4=0;
        coil1=1;
        delay(5);
        coil1=coil2=coil3=coil4=0;
        coil2=1;
        delay(5);
        coil1=coil2=coil3=coil4=0;
        coil3=1;
        delay(5);
        coil1=coil2=coil3=coil4=0;
        coil4=1;
        delay(5);
        if(motor_count>0)motor_count--;
        }
}
if(motor_count<100)
{
        if(motor=='f')
        {
        coil1=coil2=coil3=coil4=0;
        coil4=1;
        delay(5);
        coil1=coil2=coil3=coil4=0;
        coil3=1;
        delay(5);
        coil1=coil2=coil3=coil4=0;
        coil2=1;
        delay(5);
        coil1=coil2=coil3=coil4=0;
        coil1=1;
        delay(5);
        motor_count++;
        }
}
```

```
digit_split(motor_count,0xcd,3,0);              //155 count for 5mm
coil1=coil2=coil3=coil4=0;
}
        void write()
        {
                while(WR==1);
                EEADR=adrr;
                EEDATA=data;
                EEPGD=0;
                CFGS=0;
                WREN=1;
                GIE=0;
        //      sec1(6);
                EECON2=0X55;
                EECON2=0XAA;
                WR=1;
                WREN=0;
                GIE=1;
        }
```

```
void read()
        {
                EEADR=adrr;
                EEPGD=0;
                CFGS=0;
                RD=1;
                data=EEDATA;
        }

void set_mode_data()
{
        getout=0;
        lcdadrr(0x00);      //cursor blink & on/off
        lcdadrr(0x0f);      //cursor blink & on/off
        delay(10);
        digit_split(set_value,0x8c,2,0);
        adrr=123;
        read();
        set_value=data;
```

```
while(getout==0)
        {
                if(key_0==1)
                {
                delay(1);
                        if(key_0==1)
                        {
                                while(key_0==1);getout=1;
                        }
                }

                if(key_1==1)
                {
                delay(1);
                        if(key_1==1)
                        {
                        set_value++;
                        if(set_value>99)set_value=99;
                        digit_split(set_value,0x8c,2,0);
                        delay(20);
                        }
        }


        if(key_2==1)
        {
        delay(1);
                if(key_2==1)
                {
                if(set_value>0)set_value--;
                digit_split(set_value,0x8c,2,0);
                delay(20);
                }
        }
    }
    adrr=123;
    data=set_value;
    write();


    lcdadrr(0x00);      //cursor blink & on/off
    lcdadrr(0x0c);      //cursor blink & on/off
    delay(10);


}
```

```
void portint()
        {
        ADCON1=0X05;
        CMCON=0X07;
        TRISA=0x1f;
        TRISB=0xff;
        TRISC=0x03;
        TRISD=0xc0;
        TRISE=0x03;
        PORTA=PORTB=PORTC=PORTD=PORTE=0;
        }
        void lcd_display_menu()
        {
        while(array_1<=array_2)
        {
        lcddata(head[array_1]);
        array_1++;
        }
        }
```

```
void lcdint()
        {
        rs=0;
        en=0;
        delay(10);   //15ms
        lcdadrr(0x03);
        delay(5);     //5ms

        lcdadrr(0x03);
        delay(5);

        lcdadrr(0x03);
        delay(10);

        lcdadrr(0x02);
        delay(10);

        lcdadrr(0x02);
//      delay(10);
        lcdadrr(0x0c);       //function set
        delay(10);
        lcdadrr(0x00);
//      delay(10);
        lcdadrr(0x08);          //display off
        delay(10);

//      lcdadrr(0x00);
///     delay(100);
//      lcdadrr(0x01);
//      delay(100);

        lcdadrr(0x00);
//      delay(10);
        lcdadrr(0x01);          //display on 1
        delay(10);

        lcdadrr(0x00);
//      delay(10);
        lcdadrr(0x0f);          //entry mode set 6
        delay(10);
        lcdadrr(0x00);
//      delay(10);
        lcdadrr(0x0c);          //cursor blink & on/off
        delay(10);

        lcdadrr1(0x80);         //adrr
        delay(10);
//      while(1);

        array_1=0;
        array_2=15;
        lcd_display_menu();
        delay(80);

        array_1=0;
        lcdadrr1(0x01);         //adrr
        lcdadrr1(0x80);         //adrr
        while(array_1<16)
        {
        lcddata(set_mode[array_1]);
        array_1++;
        }
        digit_split(set_value,0x8c,2,0);


//      delay(80);
//      delay(80);
        }

        void lcdadrr(adrr)
        {
        rs=0;
        en=0;
        PORTD=adrr;
        enable();
        }
        void lcdadrr1(adrr)
        {
        rs=0;
        en=0;
        data1=adrr&0xf0;
        data2=adrr&0x0f;
        data=data1>>4;
        PORTD=data;
```

```
enable();
PORTD=data2;
enable();
}

void lcddata(data)
{
rs=0;
en=0;
data1=data&0xf0;
data2=data&0x0f;
data=data1>>4;
PORTD=data;
rs=1;
enable();
PORTD=data2;
rs=1;
enable();
}
```

```
void delay(int a)
        {
        T0CON=0Xc7;
        TMR0L=220;              //220
        a=a*4;
                while(a>0)
                {
                while(TMR0IF==0);
                TMR0IF=0;
                TMR0L=220;
                a--;
                }
        }


        void enable()
        {
        en=1;
        delay(4);
        en=0;
        }
```

```
void split()

{
        sub_content=1000000;split_fun();seg0=current_split;
        sub_content=100000;split_fun();seg=current_split;
        sub_content=10000;split_fun();seg1=current_split;
        sub_content=1000;split_fun();seg2=current_split;
        sub_content=100;split_fun();seg3=current_split;
        sub_content=10;split_fun();seg4=current_split;
        seg5=content;

}

void split_fun()
{
current_split=0;
current_split=content/sub_content;
content=content%sub_content;
}
```

```
      void digit_split(unsigned long int count,char place,char digit,char point)
      {
      content=count;
      split();
      lcdadrr1(place);
      if(digit>0)
      {
      if(digit==7){lcddata(seg0+0x30);digit--;} if(point==6)lcddata('.');
              if(digit==6){lcddata(seg+0x30);digit--;} if(point==5)lcddata('.');
                      if(digit==5){lcddata(seg1+0x30);digit--;}
if(point==4)lcddata('.');
if(digit==4){lcddata(seg2+0x30);digit--;}if(point==3)lcddata('.');
if(digit==3){lcddata(seg3+0x30);digit--;}if(point==2)lcddata('.');
if(digit==2){lcddata(seg4+0x30);digit--;}if(point==1)lcddata('.');
if(digit==1){lcddata(seg5+0x30);digit--;}
              }
      }
```

Appendix III

Datasheets

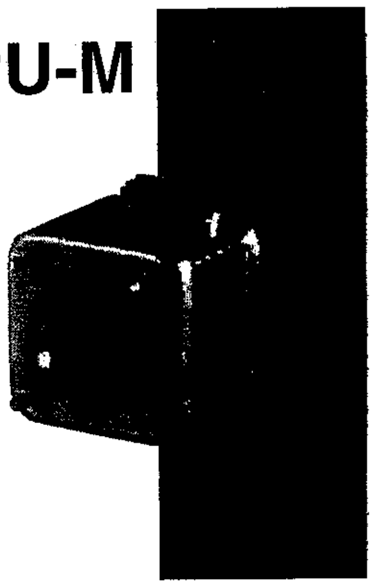
- **PIC 18 Series**
- **Stepper Motor**
- **Power Transistors**
- **Voltage Regulators**

NMB-MAT

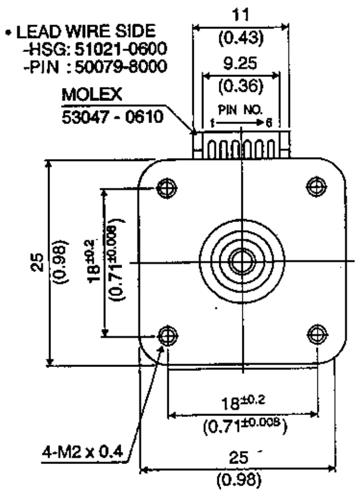
Minebea-Matsushita Motor Corporation

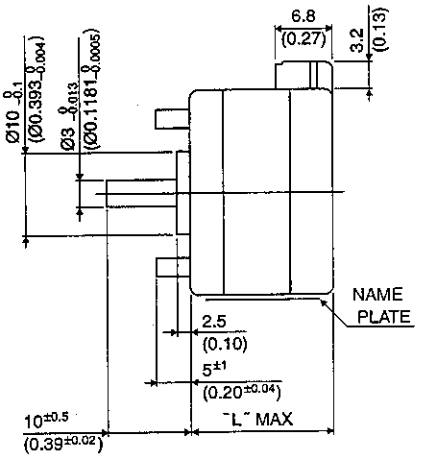
eMINEBEA. COM

10PU-M



■ 外観図　Outline





UNIT: $\frac{\text{mm}}{\text{(inch)}}$

[illegible]	[illegible]
10PU-M2	[illegible]
10PU-M0	23 (0.91)

結線表　PIN NO. VS PHASE

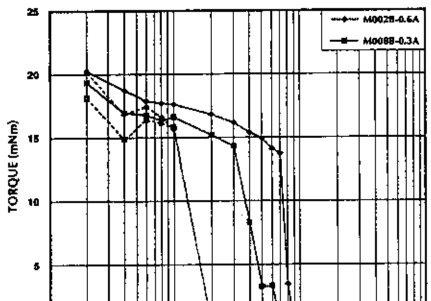
相 (PHASE)	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
ピン番号 (PIN NO.)	[illegible]	4	1	6	3

■ 仕様　Specifications

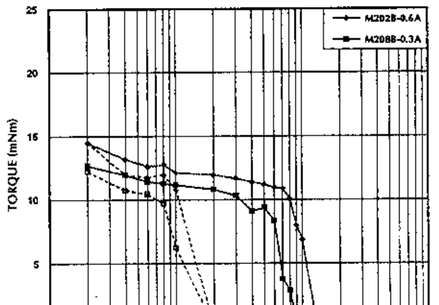
型式 Model	ステップ角 Step Angle (deg)	ドライブ方式 Drive Sequence	定格電流 Rated Current (A)	巻線抵抗 Resistance (Ohms)	ホールディングトルク Holding Torque (mNm)	インダクタンス Inductance (mH)	ロータイナーシャ Rotor Inertia (g·cm²)	ディテントトルク Detent Torque (mNm)	質量 Mass (g)
10PU-M002B	3.75	BI-POLAR	0.6	4.6	19	3.0	3	3.9	55
10PU-M008B	3.75	BI-POLAR	0.3	18	19	10.7	3	3.9	55
10PU-M202B	3.75	BI-POLAR	0.6	3.6	12	1.6	2	2.9	45
10PU-M208B	3.75	BI-POLAR	0.3	14	12	5.8	2	2.9	45

■ トルク・スピード特性　Torque/Speed Characteristics

Model No: 10PU-M002B,M008B
Driver: Chopper Dual
Supply Voltage: 24.0 (Volt)



Model No: 10PU-M202B,M208B
Driver: Chopper Dual
Supply Voltage: 24.0 (Volt)



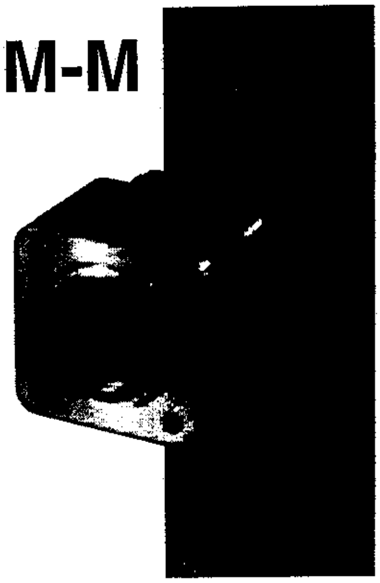
——: PULL OUT

NMB-MAT

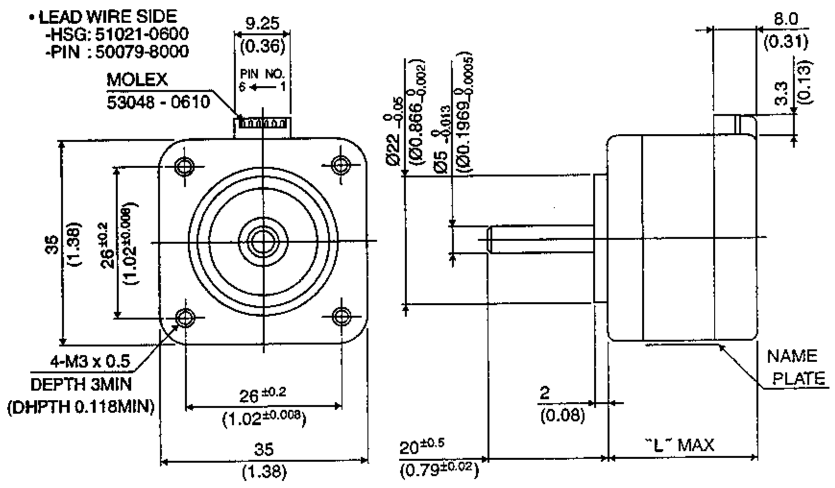
Minebea-Matsushita Motor Corporation

eMINEBEA. COM

14PM-M



■外観図　Outline



UNIT: mm/(inch)

[illegible]	[illegible]
14PM-M2	22(0.8[illegible]
14PM-M0	26(1.02)

結線表　PIN NO. VS PHASE

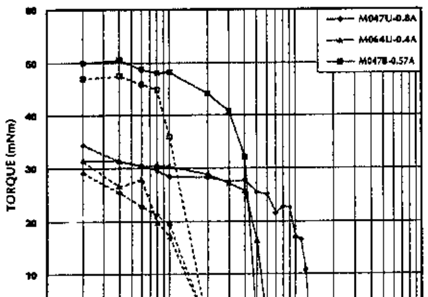
相 (PHASE)	A	A COM	[illegible]	B	B COM	[illegible]
ピン番号 (PIN NO.)	4	5	6	3	2	[illegible]

■仕様　Specifications

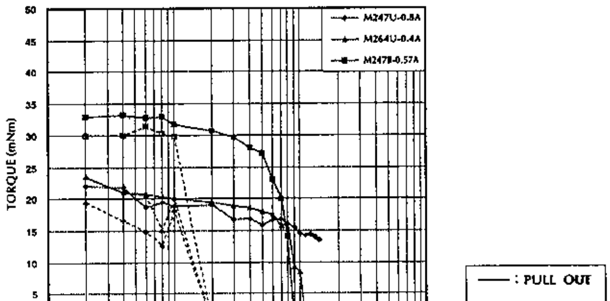
型式	ステップ角	ドライブ方式	定格電流	巻線抵抗	ホールディング トルク	インダクタンス	ロータ イナーシャ	ディテント トルク	[illegible]
Model	Step Angle	Drive Sequence	Rated Current	Resistance	Holding Torque	Inductance	Rotor Inertia	Detent Torque	Mass
	(deg)		(A)	(Ohms)	(mNm)	(mH)	(g-cm²)	(mNm)	(g)
14PM-M047U	1.8	UNI-POLAR	0.8	2.4	34	1.5	8	5.8	120
14PM-M064U	1.8	UNI-POLAR	0.4	9.0	34	5.8	8	5.8	120
14PM-M247U	1.8	UNI-POLAR	0.8	1.9	24	0.8	5	4.7	105
14PM-M264U	1.8	UNI-POLAR	0.4	7.0	24	3.1	5	4.7	105
14PM-M047B	1.8	BI-POLAR	0.6	4.8	50	6.1	8	5.8	120
14PM-M247B	1.8	BI-POLAR	0.6	3.8	34	3.2	5	4.7	105

■トルク・スピード特性　Torque/Speed Characteristics

Model No: 14PM-M047U,M064U,M047B
Driver: Chopper Dual
Supply Voltage: 24.0 (Volt)



Model No: 14PM-M247U,M264U,M247B
Driver: Chopper Dual
Supply Voltage: 24.0 (Volt)



MICROCHIP

PIC18FXX8

28/40-Pin High-Performance, Enhanced Flash Microcontrollers with CAN

High-Performance RISC CPU:

- Linear program memory addressing up to 2 Mbytes
- Linear data memory addressing to 4 Kbytes
- Up to 10 MIPS operation
- DC – 40 MHz clock input
- 4 MHz-10 MHz oscillator/clock input with PLL active
- 16-bit wide instructions, 8-bit wide data path
- Priority levels for interrupts
- 8 x 8 Single-Cycle Hardware Multiplier

Peripheral Features:

- High current sink/source 25 mA/25 mA
- Three external interrupt pins
- Timer0 module: 8-bit/16-bit timer/counter with 8-bit programmable prescaler
- Timer1 module: 16-bit timer/counter
- Timer2 module: 8-bit timer/counter with 8-bit period register (time base for PWM)
- Timer3 module: 16-bit timer/counter
- Secondary oscillator clock option – Timer1/Timer3
- Capture/Compare/PWM (CCP) modules; CCP pins can be configured as:
  - Capture input: 16-bit, max resolution 6.25 ns
  - Compare: 16-bit, max resolution 100 ns (TCY)
  - PWM output: PWM resolution is 1 to 10-bit Max. PWM freq. @:8-bit resolution = 156 kHz
    10-bit resolution = 39 kHz
- Enhanced CCP module which has all the features of the standard CCP module, but also has the following features for advanced motor control:
  - 1, 2 or 4 PWM outputs
  - Selectable PWM polarity
  - Programmable PWM dead time
- Master Synchronous Serial Port (MSSP) with two modes of operation:
  - 3-wire SPI™ (Supports all 4 SPI modes)
  - I^2C™ Master and Slave mode
- Addressable USART module:
  - Supports interrupt-on-address bit

Advanced Analog Features:

- 10-bit, up to 8-channel Analog-to-Digital Converter module (A/D) with:
  - Conversion available during Sleep
  - Up to 8 channels available
- Analog Comparator module:
  - Programmable input and output multiplexing
- Comparator Voltage Reference module
- Programmable Low-Voltage Detection (LVD) module:
  - Supports interrupt-on-Low-Voltage Detection
- Programmable Brown-out Reset (BOR)

CAN bus Module Features:

- Complies with ISO CAN Conformance Test
- Message bit rates up to 1 Mbps
- Conforms to CAN 2.0B Active Spec with:
  - 29-bit Identifier Fields
  - 8-byte message length
  - 3 Transmit Message Buffers with prioritization
  - 2 Receive Message Buffers
  - 6 full, 29-bit Acceptance Filters
  - Prioritization of Acceptance Filters
  - Multiple Receive Buffers for High Priority Messages to prevent loss due to overflow
  - Advanced Error Management Features

Special Microcontroller Features:

- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own on-chip RC oscillator
- Programmable code protection
- Power-saving Sleep mode
- Selectable oscillator options, including:
  - 4x Phase Lock Loop (PLL) of primary oscillator
  - Secondary Oscillator (32 kHz) clock input
- In-Circuit Serial Programming™ (ICSP™) via two pins

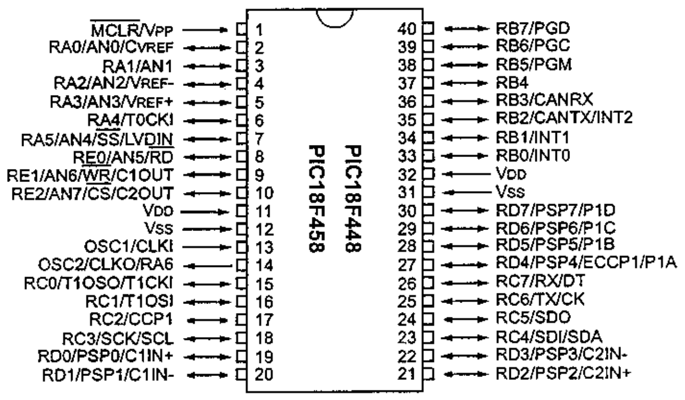
Flash Technology:

- Low-power, high-speed Enhanced Flash technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Industrial and Extended temperature ranges

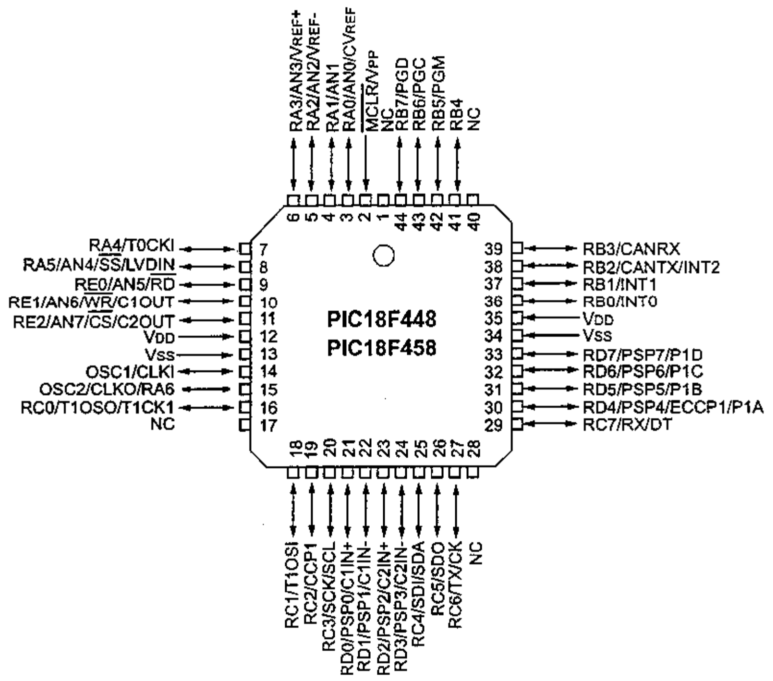
Device	Program Memory		Data Memory		I/O	10-bit A/D (ch)	Comparators	CCP/ ECCP (PWM)	MSSP		USART	Timers 8/16-bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)					SPI™	Master I²C™		
PIC18F248	16K	8192	768	256	22	5	—	1/0	Y	Y	Y	1/3
PIC18F258	32K	16384	1536	256	22	5	—	1/0	Y	Y	Y	1/3
PIC18F448	16K	8192	768	256	33	8	2	1/1	Y	Y	Y	1/3
PIC18F458	32K	16384	1536	256	33	8	2	1/1	Y	Y	Y	1/3

Pin Diagrams

PDIP

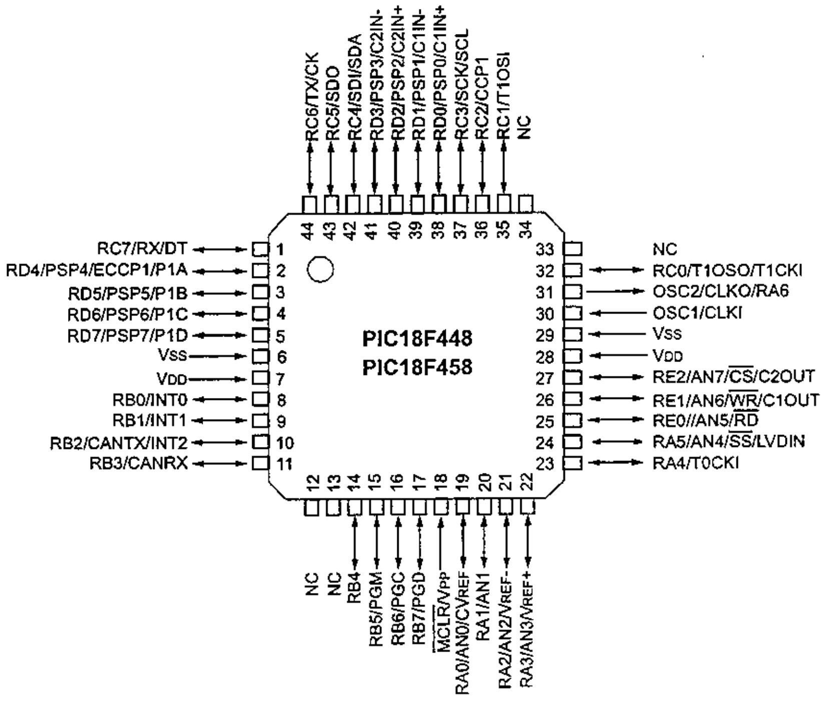


PLCC

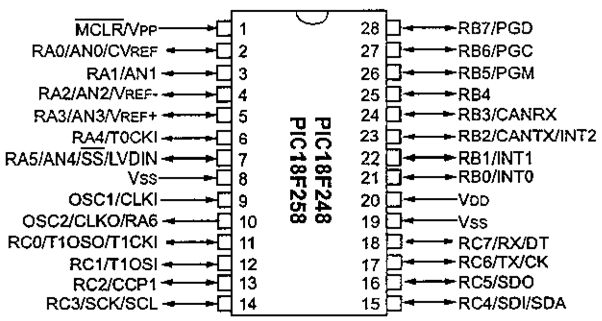


Pin Diagrams (Continued)

TQFP



SPDIP, SOIC



1.0 DEVICE OVERVIEW

This document contains dev ce specific information for the following devices:

- PIC18F248
- PIC18F258
- PIC18F448
- PIC18F458

These devices are available in 28-pin, 40-pin and 44-pin packages. They are differentiated from each other in four ways:

1. PIC18FX58 devices have twice the Flash program memory and data RAM of PIC18FX48 devices (32 Kbytes and 1536 bytes vs. 16 Kbytes and 768 bytes, respectively).
2. PIC18F2X8 devices implement 5 A/D channels, as opposed to 8 for PIC18F4X8 devices.
3. PIC18F2X8 devices implement 3 I/O ports, while PIC18F4X8 devices implement 5.
4. Only PIC18F4X8 devices implement the Enhanced CCP module, analog comparators and the Parallel Slave Port.

All other features for devices in the PIC18FXX8 family, including the serial communications modules, are identical. These are summarized in Table 1-1.

Block diagrams of the PIC18F2X8 and PIC18F4X8 devices are provided in Figure 1-1 and Figure 1-2, respectively. The pinouts for these device families are listed in Table 1-2.

TABLE 1-1: PIC18FXX8 DEVICE FEATURES

Features		PIC18F248	PIC18F258	PIC18F448	PIC18F458
Operating Frequency		DC – 40 MHz	DC – 40 MHz	DC – 40 MHz	DC – 40 MHz
Internal Program Memory	Bytes	16K	32K	16K	32K
	# of Single-Word Instructions	8192	16384	8192	16384
Data Memory (Bytes)		768	1536	768	1536
Data EEPROM Memory (Bytes)		256	256	256	256
Interrupt Sources		17	17	21	21
I/O Ports		Ports A, B, C	Ports A, B, C	Ports A, B, C, D, E	Ports A, B, C, D, E
Timers		4	4	4	4
Capture/Compare/PWM Modules		1	1	1	1
Enhanced Capture/Compare/ PWM Modules		—	—	1	1
Serial Communications		MSSP, CAN, Addressable USART	MSSP, CAN, Addressable USART	MSSP, CAN, Addressable USART	MSSP, CAN, Addressable USART
Parallel Communications (PSP)		No	No	Yes	Yes
10-bit Analog-to-Digital Converter		5 input channels	5 input channels	8 input channels	8 input channels
Analog Comparators		No	No	2	2
Analog Comparators VREF Output		N/A	N/A	Yes	Yes
Resets (and Delays)		POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)
Programmable Low-Voltage Detect		Yes	Yes	Yes	Yes
Programmable Brown-out Reset		Yes	Yes	Yes	Yes
CAN Module		Yes	Yes	Yes	Yes
In-Circuit Serial Programming (ICSP™)		Yes	Yes	Yes	Yes
Instruction Set		75 Instructions	75 Instructions	75 Instructions	75 Instructions
Packages		28-pin SPDIP 28-pin SOIC	28-pin SPDIP 28-pin SOIC	40-pin PDIP 44-pin PLCC 44-pin TQFP	40-pin PDIP 44-pin PLCC 44-pin TQFP

FIGURE 1-1: PIC18F248/258 BLOCK DIAGRAM

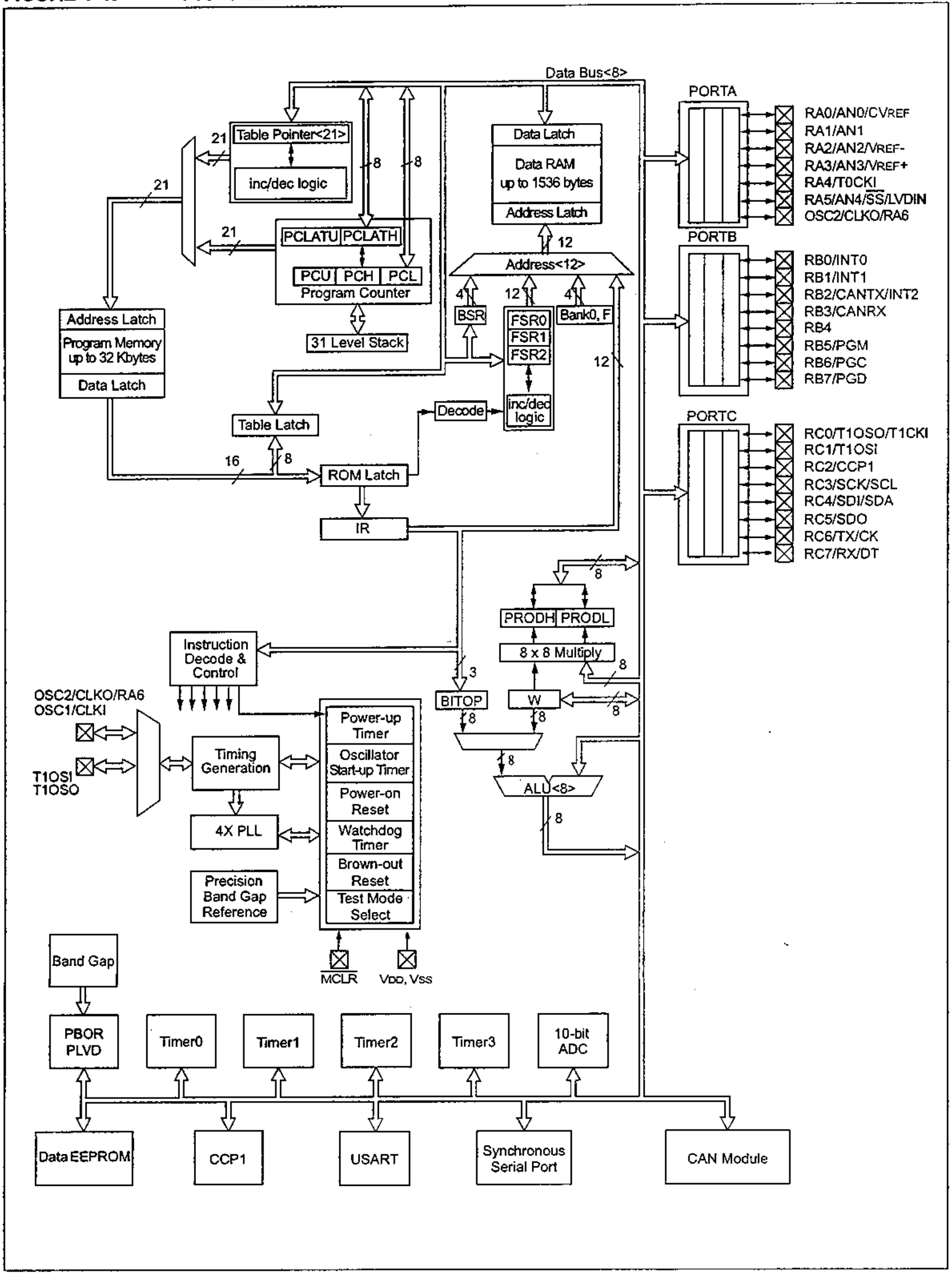


FIGURE 1-2: PIC18F448/458 BLOCK DIAGRAM

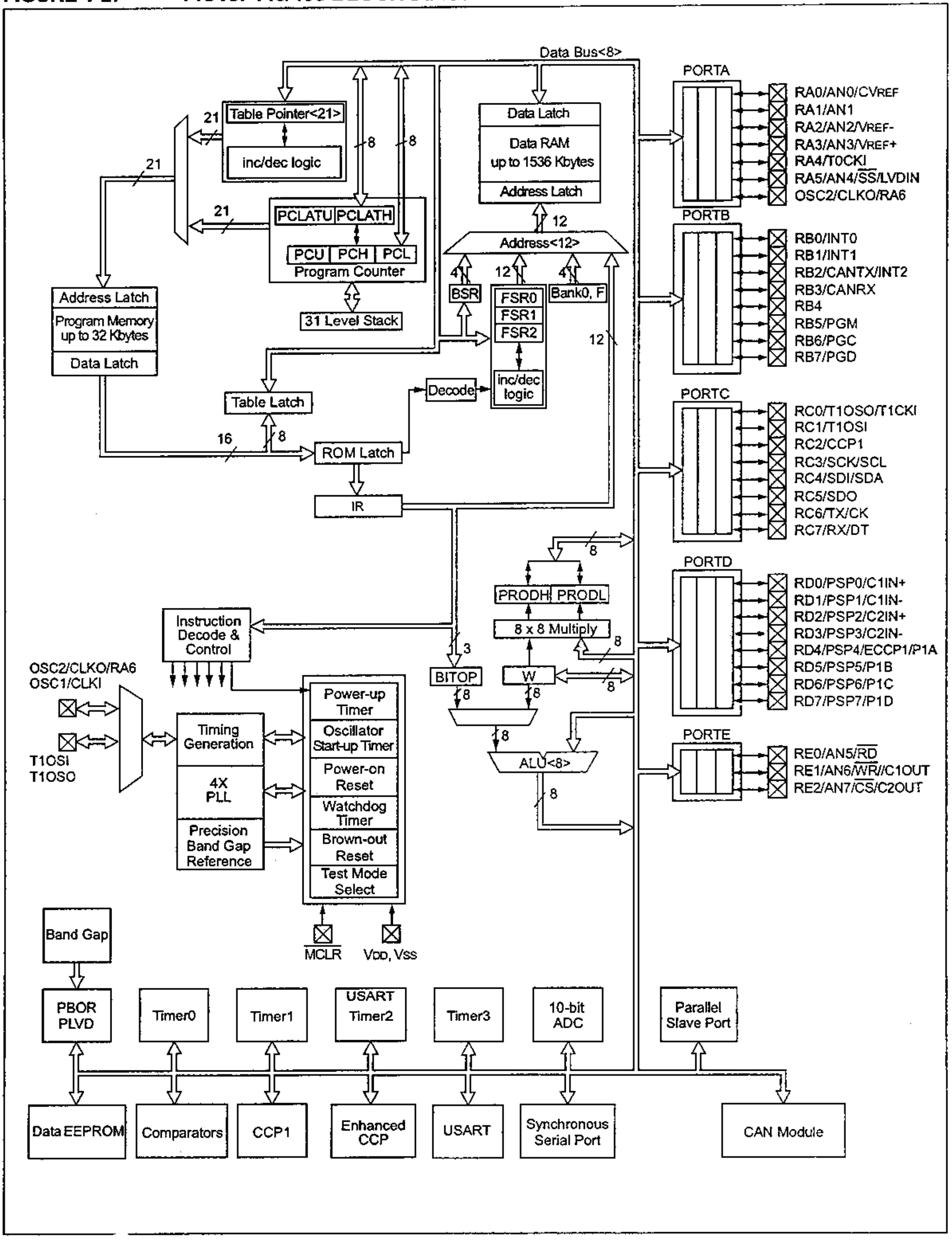


TABLE 1-2: PIC18FXX8 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number				Pin Type	Buffer Type	Description
	PIC18F248/258	PIC18F448/458					
	SPDIP, SOIC	PDIP	TQFP	PLCC			
$\overline{\text{MCLR}}$/VPP	1	1	18	2			Master Clear (input) or programming voltage (output).
$\overline{\text{MCLR}}$					I	ST	Master Clear (Reset) input. This pin is an active low Reset to the device.
VPP					P	—	Programming voltage input.
NC	—	—	12, 13, 33, 34	1, 17, 28, 40	—	—	These pins should be left unconnected.
OSC1/CLKI	9	13	30	14			Oscillator crystal or external clock input.
OSC1					I	CMOS/ST	Oscillator crystal input or external clock source input. ST buffer when configured in RC mode; otherwise, CMOS.
CLKI					I	CMOS	External clock source input. Always associated with pin function OSC1 (see OSC1/CLKI, OSC2/CLKO pins).
OSC2/CLKO/RA6	10	14	31	15			Oscillator crystal or clock output.
OSC2					O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO					O	—	In RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
RA6					I/O	TTL	General purpose I/O pin.

Legend: TTL	= TTL compatible input	CMOS	= CMOS compatible input or output
ST	= Schmitt Trigger input with CMOS levels	Analog	= Analog input
I	= Input	O	= Output
P	= Power	OD	= Open-Drain (no P diode to VDD)

TABLE 1-2: PIC18FXX8 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number				Pin Type	Buffer Type	Description
	PIC18F248/258	PIC18F448/458					
	SPDIP, SOIC	PDIP	TQFP	PLCC			
							PORTA is a bidirectional I/O port.
RA0/AN0/CVREF	2	2	19	3			
RA0					I/O	TTL	Digital I/O.
AN0					I	Analog	Analog input 0.
CVREF					O	Analog	Comparator voltage reference output.
RA1/AN1	3	3	20	4			
RA1					I/O	TTL	Digital I/O.
AN1					I	Analog	Analog input 1.
RA2/AN2/VREF-	4	4	21	5			
RA2					I/O	TTL	Digital I/O.
AN2					I	Analog	Analog input 2.
VREF-					I	Analog	A/D reference voltage (Low) input.
RA3/AN3/VREF+	5	5	22	6			
RA3					I/O	TTL	Digital I/O.
AN3					I	Analog	Analog input 3.
VREF+					I	Analog	A/D reference voltage (High) input.
RA4/T0CKI	6	6	23	7			
RA4					I/O	TTL/OD	Digital I/O – open-drain when configured as output.
T0CKI					I	ST	Timer0 external clock input.
RA5/AN4/$\overline{\text{SS}}$/LVDIN	7	7	24	8			
RA5					I/O	TTL	Digital I/O.
AN4					I	Analog	Analog input 4.
$\overline{\text{SS}}$					I	ST	SPI™ slave select input.
LVDIN					I	Analog	Low-Voltage Detect input.
RA6							See the OSC2/CLKO/RA6 pin.

Legend: TTL	= TTL compatible input	CMOS	= CMOS compatible input or output
ST	= Schmitt Trigger input with CMOS levels	Analog	= Analog input
I	= Input	O	= Output
P	= Power	OD	= Open-Drain (no P diode to VDD)

TABLE 1-2: PIC18FXX8 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number				Pin Type	Buffer Type	Description
	PIC18F248/258	PIC18F448/458					
	SPDIP, SOIC	PDIP	TQFP	PLCC			
							PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0/INT0	21	33	8	36			
RB0					I/O	TTL	Digital I/O.
INT0					I	ST	External interrupt 0.
RB1/INT1	22	34	9	37			
RB1					I/O	TTL	Digital I/O.
INT1					I	ST	External interrupt 1.
RB2/CANTX/INT2	23	35	10	38			
RB2					I/O	TTL	Digital I/O.
CANTX					O	TTL	Transmit signal for CAN bus.
INT2					I	ST	External interrupt 2.
RB3/CANRX	24	36	11	39			
RB3					I/O	TTL	Digital I/O.
CANRX					I	TTL	Receive signal for CAN bus.
RB4	25	37	14	41	I/O	TTL	Digital I/O. Interrupt-on-change pin.
RB5/PGM	26	38	15	42			
RB5					I/O	TTL	Digital I/O. Interrupt-on-change pin.
PGM					I	ST	Low-voltage ICSP™ programming enable.
RB6/PGC	27	39	16	43			
RB6					I/O	TTL	Digital I/O. In-Circuit Debugger pin. Interrupt-on-change pin.
PGC					I	ST	ICSP programming clock.
RB7/PGD	28	40	17	44			
RB7					I/O	TTL	Digital I/O. In-Circuit Debugger pin. Interrupt-on-change pin.
PGD					I/O	ST	ICSP programming data.

Legend:

TTL	= TTL compatible input	CMOS	= CMOS compatible input or output
ST	= Schmitt Trigger input with CMOS levels	Analog	= Analog input
I	= Input	O	= Output
P	= Power	OD	= Open-Drain (no P diode to V_{DD})

TABLE 1-2: PIC18FXX8 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number				Pin Type	Buffer Type	Description
	PIC18F248/258	PIC18F448/458					
	SPDIP, SOIC	PDIP	TQFP	PLCC			
							PORTC is a bidirectional I/O port.
RC0/T1OSO/T1CKI	11	15	32	16			
RC0					I/O	ST	Digital I/O.
T1OSO					O	—	Timer1 oscillator output.
T1CKI					I	ST	Timer1/Timer3 external clock input.
RC1/T1OSI	12	16	35	18			
RC1					I/O	ST	Digital I/O.
T1OSI					I	CMOS	Timer1 oscillator input.
RC2/CCP1	13	17	36	19			
RC2					I/O	ST	Digital I/O.
CCP1					I/O	ST	Capture 1 input/Compare 1 output/PWM1 output.
RC3/SCK/SCL	14	18	37	20			
RC3					I/O	ST	Digital I/O.
SCK					I/O	ST	Synchronous serial clock input/output for SPI™ mode.
SCL					I/O	ST	Synchronous serial clock input/output for I^2C™ mode.
RC4/SDI/SDA	15	23	42	25			
RC4					I/O	ST	Digital I/O.
SDI					I	ST	SPI data in.
SDA					I/O	ST	I^2C data I/O.
RC5/SDO	16	24	43	26			
RC5					I/O	ST	Digital I/O.
SDO					O	—	SPI data out.
RC6/TX/CK	17	25	44	27			
RC6					I/O	ST	Digital I/O.
TX					O	—	USART asynchronous transmit.
CK					I/O	ST	USART synchronous clock (see RX/DT).
RC7/RX/DT	18	26	1	29			
RC7					I/O	ST	Digital I/O.
RX					I	ST	USART asynchronous receive.
DT					I/O	ST	USART synchronous data (see TX/CK).

Legend:			
TTL	= TTL compatible input	CMOS	= CMOS compatible input or output
ST	= Schmitt Trigger input with CMOS levels	Analog	= Analog input
I	= Input	O	= Output
P	= Power	OD	= Open-Drain (no P diode to VDD)

TABLE 1-2: PIC18FXX8 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number				Pin Type	Buffer Type	Description
	PIC18F248/258	PIC18F448/458					
	SPDIP, SOIC	PDIP	TQFP	PLCC			
							PORTD is a bidirectional I/O port. These pins have TTL input buffers when external memory is enabled.
RD0/PSP0/C1IN+	—	19	38	21			
RD0					I/O	ST	Digital I/O.
PSP0					I/O	TTL	Parallel Slave Port data.
C1IN+					I	Analog	Comparator 1 input.
RD1/PSP1/C1IN-	—	20	39	22			
RD1					I/O	ST	Digital I/O.
PSP1					I/O	TTL	Parallel Slave Port data.
C1IN-					I	Analog	Comparator 1 input.
RD2/PSP2/C2IN+	—	21	40	23			
RD2					I/O	ST	Digital I/O.
PSP2					I/O	TTL	Parallel Slave Port data.
C2IN+					I	Analog	Comparator 2 input.
RD3/PSP3/C2IN-	—	22	41	24			
RD3					I/O	ST	Digital I/O.
PSP3					I/O	TTL	Parallel Slave Port data.
C2IN-					I	Analog	Comparator 2 input.
RD4/PSP4/ECCP1/P1A	—	27	2	30			
RD4					I/O	ST	Digital I/O.
PSP4					I/O	TTL	Parallel Slave Port data.
ECCP1					I/O	ST	ECCP1 capture/compare.
P1A					O	—	ECCP1 PWM output A.
RD5/PSP5/P1B	—	28	3	31			
RD5					I/O	ST	Digital I/O.
PSP5					I/O	TTL	Parallel Slave Port data.
P1B					O	—	ECCP1 PWM output B.
RD6/PSP6/P1C	—	29	4	32			
RD6					I/O	ST	Digital I/O.
PSP6					I/O	TTL	Parallel Slave Port data.
P1C					O	—	ECCP1 PWM output C.
RD7/PSP7/P1D	—	30	5	33			
RD7					I/O	ST	Digital I/O.
PSP7					I/O	TTL	Parallel Slave Port data.
P1D					O	—	ECCP1 PWM output D.

Legend:

TTL	= TTL compatible input	CMOS	= CMOS compatible input or output
ST	= Schmitt Trigger input with CMOS levels	Analog	= Analog input
I	= Input	O	= Output
P	= Power	OD	= Open-Drain (no P diode to V_{DD})

TABLE 1-2: PIC18FXX8 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number				Pin Type	Buffer Type	Description
	PIC18F248/258	PIC18F448/458					
	SPDIP, SOIC	PDIP	TQFP	PLCC			
							PORTE is a bidirectional I/O port.
RE0/AN5/$\overline{RD}$	—	8	25	9			
RE0					I/O	ST	Digital I/O.
AN5					I	Analog	Analog input 5.
$\overline{RD}$					I	TTL	Read control for Parallel Slave Port (see $\overline{WR}$ and $\overline{CS}$ pins).
RE1/AN6/$\overline{WR}$/C1OUT	—	9	26	10			
RE1					I/O	ST	Digital I/O.
AN6					I	Analog	Analog input 6.
$\overline{WR}$					I	TTL	Write control for Parallel Slave Port (see $\overline{CS}$ and $\overline{RD}$ pins).
C1OUT					O	Analog	Comparator 1 output.
RE2/AN7/$\overline{CS}$/C2OUT	—	10	27	11			
RE2					I/O	ST	Digital I/O.
AN7					I	Analog	Analog input 7.
$\overline{CS}$					I	TTL	Chip select control for Parallel Slave Port (see $\overline{RD}$ and $\overline{WR}$ pins).
C2OUT					O	Analog	Comparator 2 output.
V_{SS}	19, 8	12, 31	6, 29	13, 34	—	—	Ground reference for logic and I/O pins.
V_{DD}	20	11, 32	7, 28	12, 35	—	—	Positive supply for logic and I/O pins.

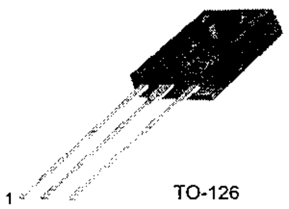
Legend:			
TTL	= TTL compatible input	CMOS	= CMOS compatible input or output
ST	= Schmitt Trigger input with CMOS levels	Analog	= Analog input
I	= Input	O	= Output
P	= Power	OD	= Open-Drain (no P diode to V_{DD})

FAIRCHILD
SEMICONDUCTOR™

BD135/137/139

Medium Power Linear and Switching Applications

- Complement to BD136, BD138 and BD140 respectively



TO-126
1. Emitter 2.Collector 3.Base

NPN Epitaxial Silicon Transistor

Absolute Maximum Ratings T_C=25°C unless otherwise noted

Symbol	Parameter	Value	Units
V_{CBO}	Collector-Base Voltage : BD135	45	V
	: BD137	60	V
	: BD139	80	V
V_{CEO}	Collector-Emitter Voltage : BD135	45	V
	: BD137	60	V
	: BD139	80	V
V_{EBO}	Emitter-Base Voltage	5	V
I_C	Collector Current (DC)	1.5	A
I_{CP}	Collector Current (Pulse)	3.0	A
I_B	Base Current	0.5	A
P_C	Collector Dissipation (T_C=25°C)	12.5	W
P_C	Collector Dissipation (T_a=25°C)	1.25	W
T_J	Junction Temperature	150	°C
T_{STG}	Storage Temperature	- 55 ~ 150	°C

Electrical Characteristics T_C=25°C unless otherwise noted

Symbol	Parameter	Test Condition	Min.	Typ.	Max.	Units
V_{CEO}(sus)	Collector-Emitter Sustaining Voltage					
	: BD135	I_C = 30mA, I_B = 0	45			V
	: BD137		60			V
	: BD139		80			V
I_{CBO}	Collector Cut-off Current	V_{CB} = 30V, I_E = 0			0.1	μA
I_{EBO}	Emitter Cut-off Current	V_{EB} = 5V, I_C = 0			10	μA
h_{FE1}	DC Current Gain : ALL DEVICE	V_{CE} = 2V, I_C = 5mA	25			
h_{FE2}	: ALL DEVICE	V_{CE} = 2V, I_C = 0.5A	25			
h_{FE3}	: BD135	V_{CE} = 2V, I_C = 150mA	40		250	
	: BD137, BD139		40		160	
V_{CE}(sat)	Collector-Emitter Saturation Voltage	I_C = 500mA, I_B = 50mA			0.5	V
V_{BE}(on)	Base-Emitter ON Voltage	V_{CE} = 2V, I_C = 0.5A			1	V

h_{FE} Classification

Classification	6	10	16
h_{FE3}	40 ~ 100	63 ~ 160	100 ~ 250

Typical Characteristics

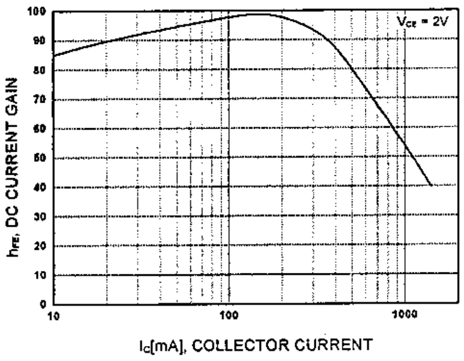


Figure 1. DC current Gain

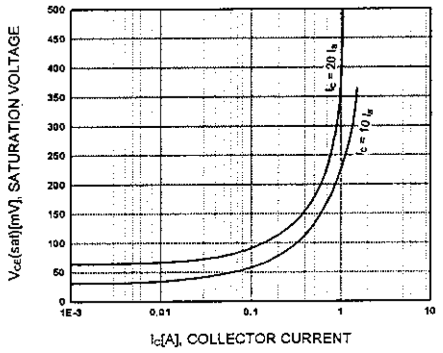


Figure 2. Collector-Emitter Saturation Voltage

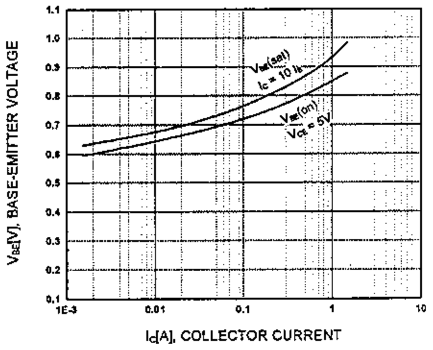


Figure 3. Base-Emitter Voltage

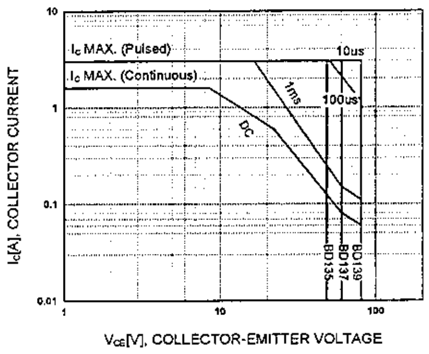


Figure 4. Safe Operating Area

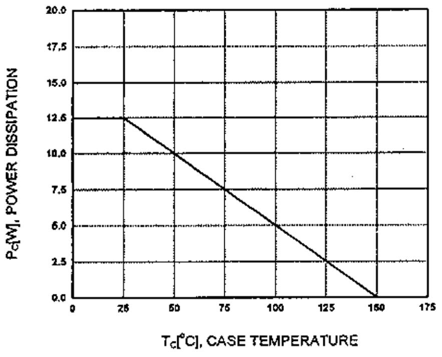


Figure 5. Power Derating

National Semiconductor

LM78XX Series Voltage Regulators

General Description

The LM78XX series of three terminal regulators is available with several fixed output voltages making them useful in a wide range of applications. One of these is local on card regulation, eliminating the distribution problems associated with single point regulation. The voltages available allow these regulators to be used in logic systems, instrumentation, HiFi, and other solid state electronic equipment. Although designed primarily as fixed voltage regulators these devices can be used with external components to obtain adjustable voltages and currents.

The LM78XX series is available in an aluminum TO-3 package which will allow over 1.0A load current if adequate heat sinking is provided. Current limiting is included to limit the peak output current to a safe value. Safe area protection for the output transistor is provided to limit internal power dissipation. If internal power dissipation becomes too high for the heat sinking provided, the thermal shutdown circuit takes over preventing the IC from overheating.

Considerable effort was expanded to make the LM78XX series of regulators easy to use and minimize the number of external components. It is not necessary to bypass the output, although this does improve transient response. Input bypassing is needed only if the regulator is located far from the filter capacitor of the power supply.

For output voltage other than 5V, 12V and 15V the LM117 series provides an output voltage range from 1.2V to 57V.

Features

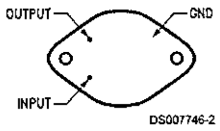
- Output current in excess of 1A
- Internal thermal overload protection
- No external components required
- Output transistor safe area protection
- Internal short circuit current limit
- Available in the aluminum TO-3 package

Voltage Range

LM7805C	5V
LM7812C	12V
LM7815C	15V

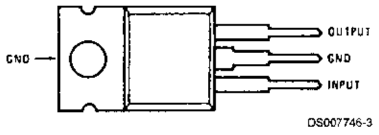
Connection Diagrams

Metal Can Package
TO-3 (K)
Aluminum



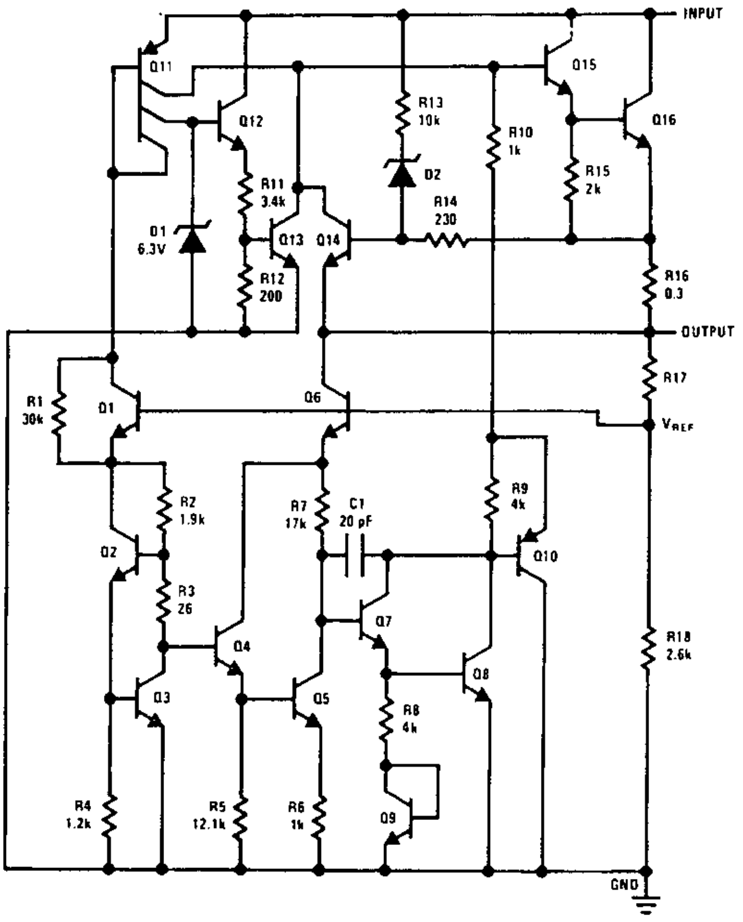
Bottom View
Order Number LM7805CK,
LM7812CK or LM7815CK
See NS Package Number KC02A

Plastic Package
TO-220 (T)



Top View
Order Number LM7805CT,
LM7812CT or LM7815CT
See NS Package Number T03B

Schematic

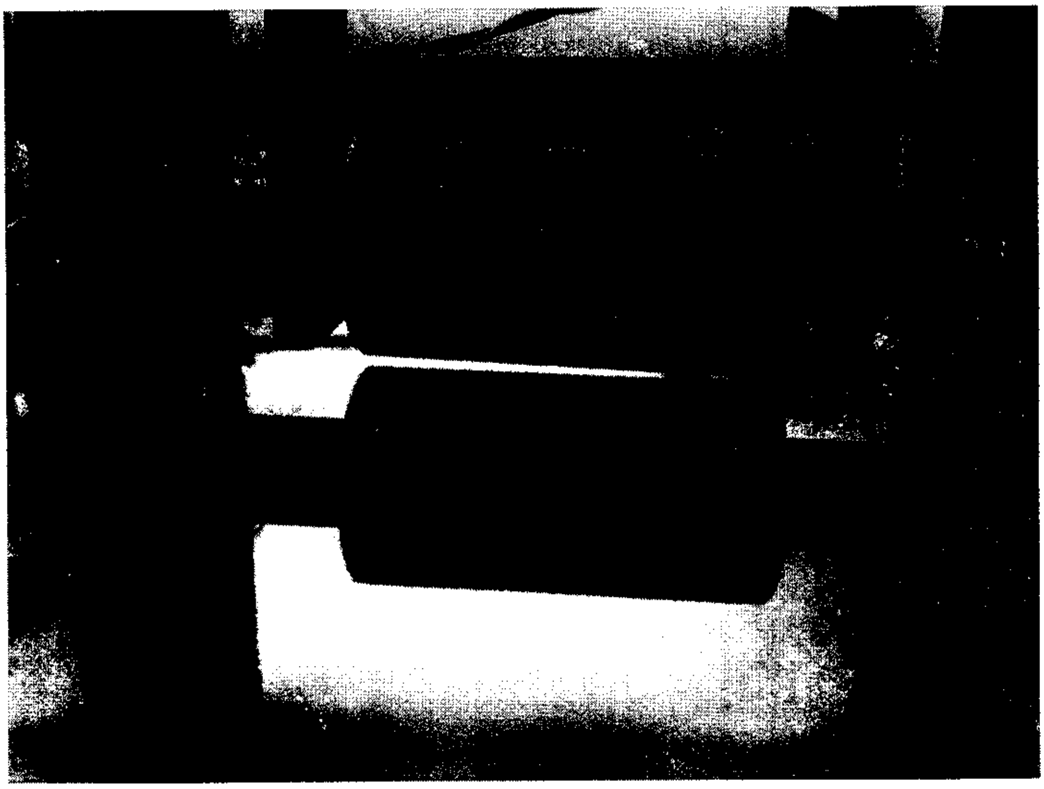


DS007746-1

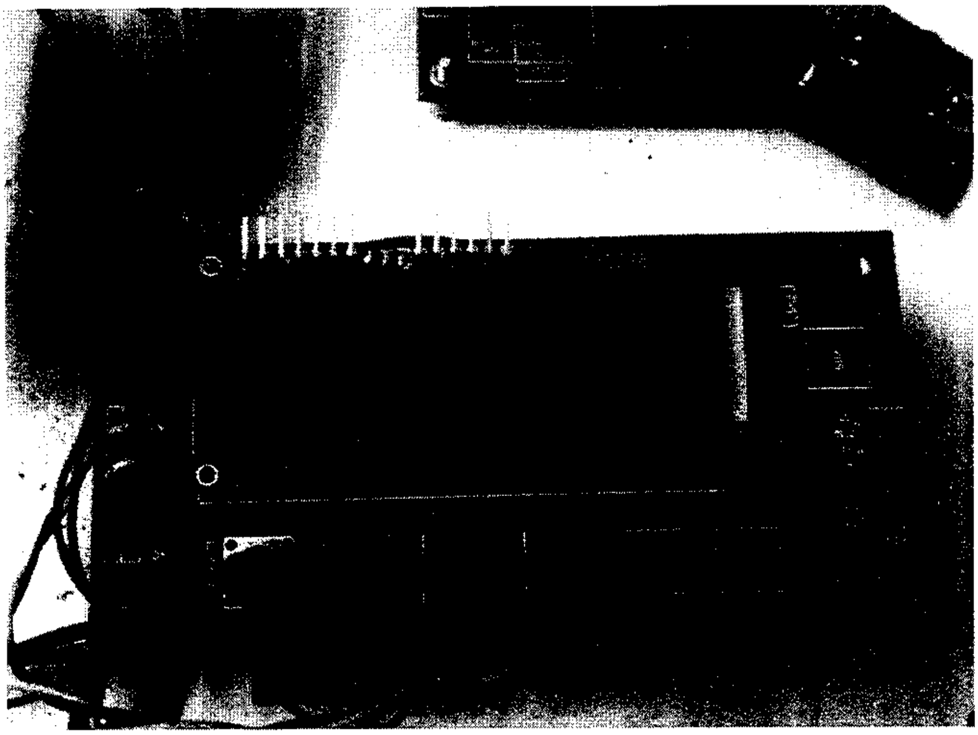
List of Photographs

List of Photographs

Rollers

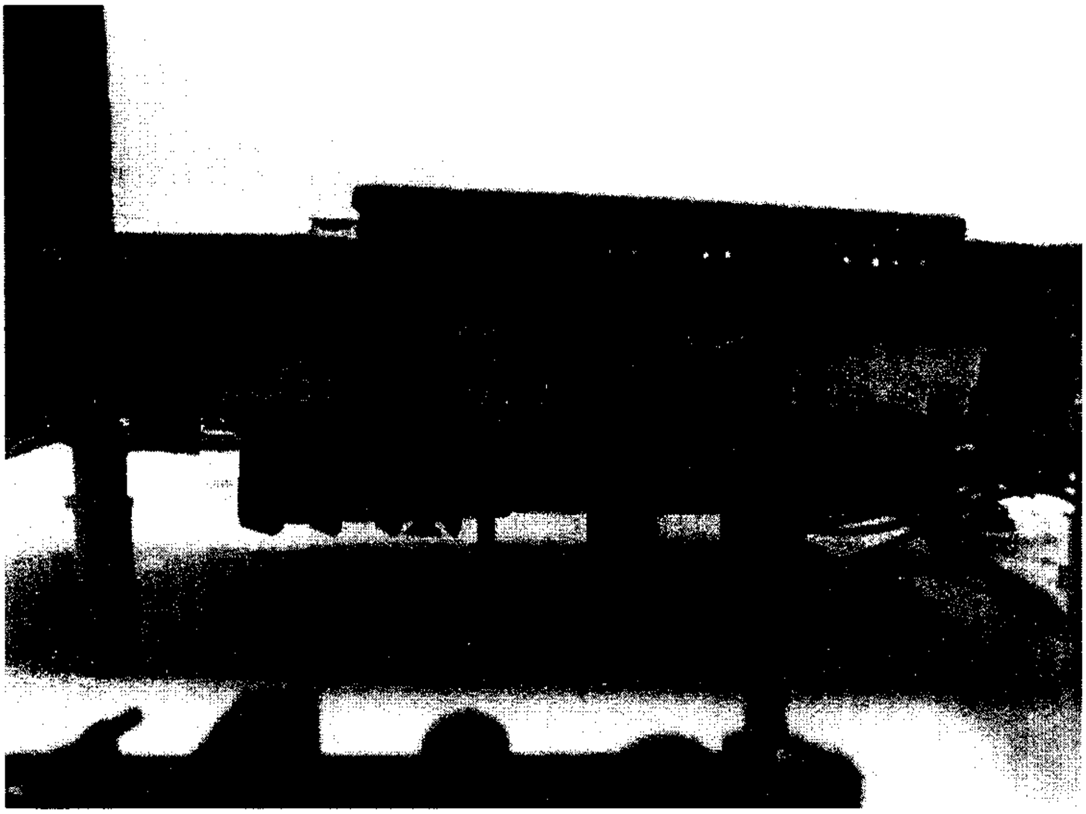


LCD

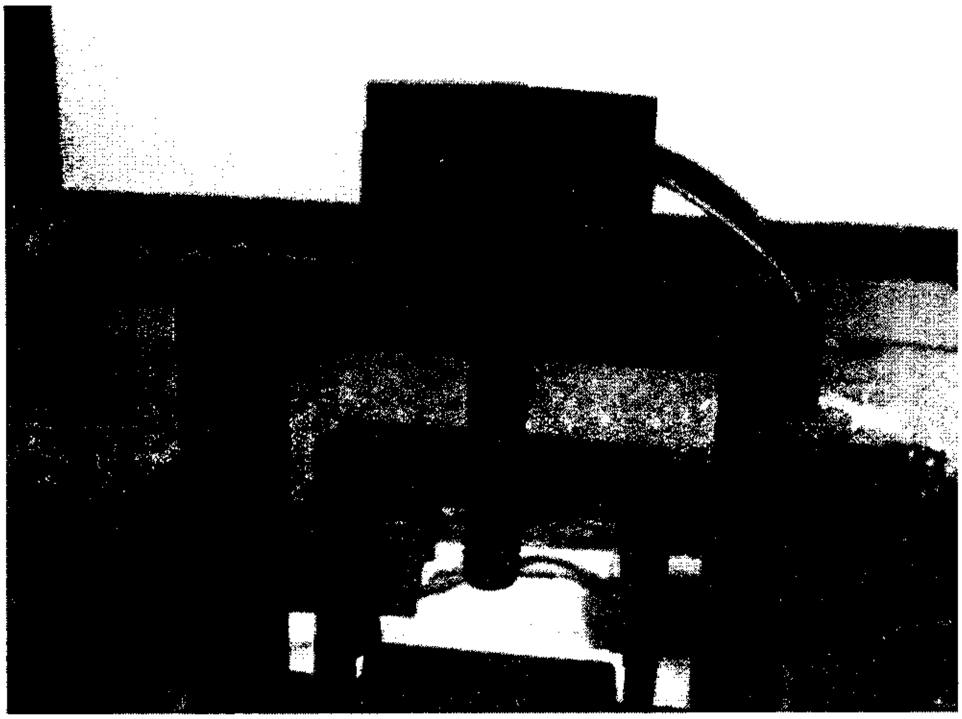


Ultrasonic Sensors

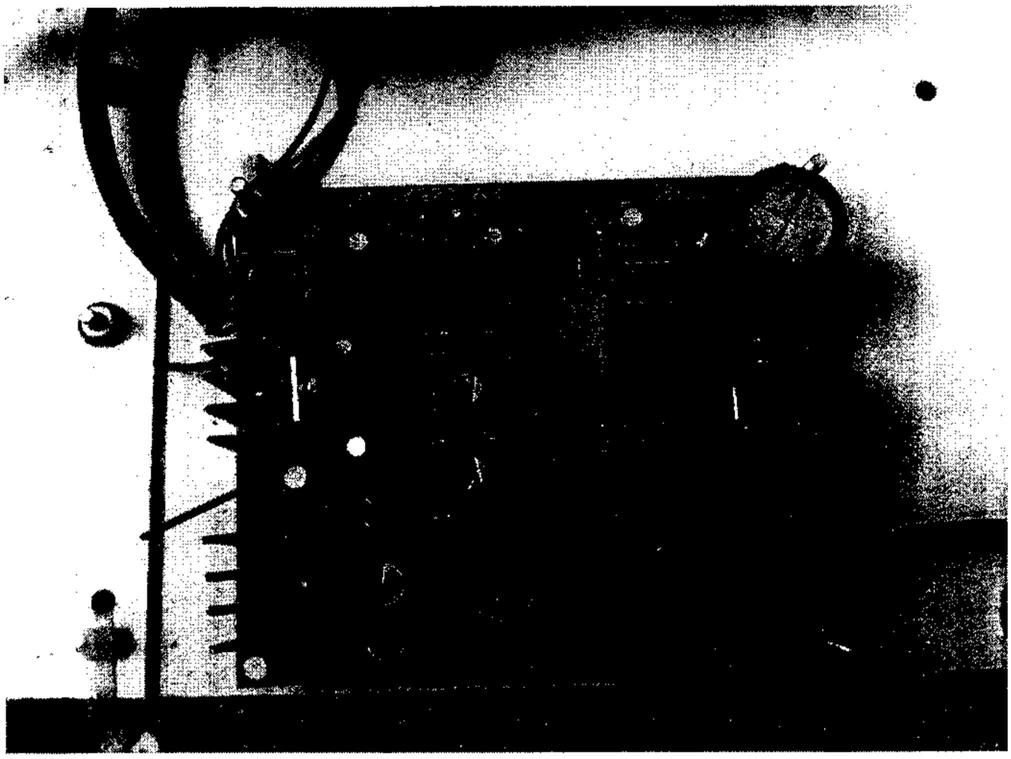
PIC Interface Board



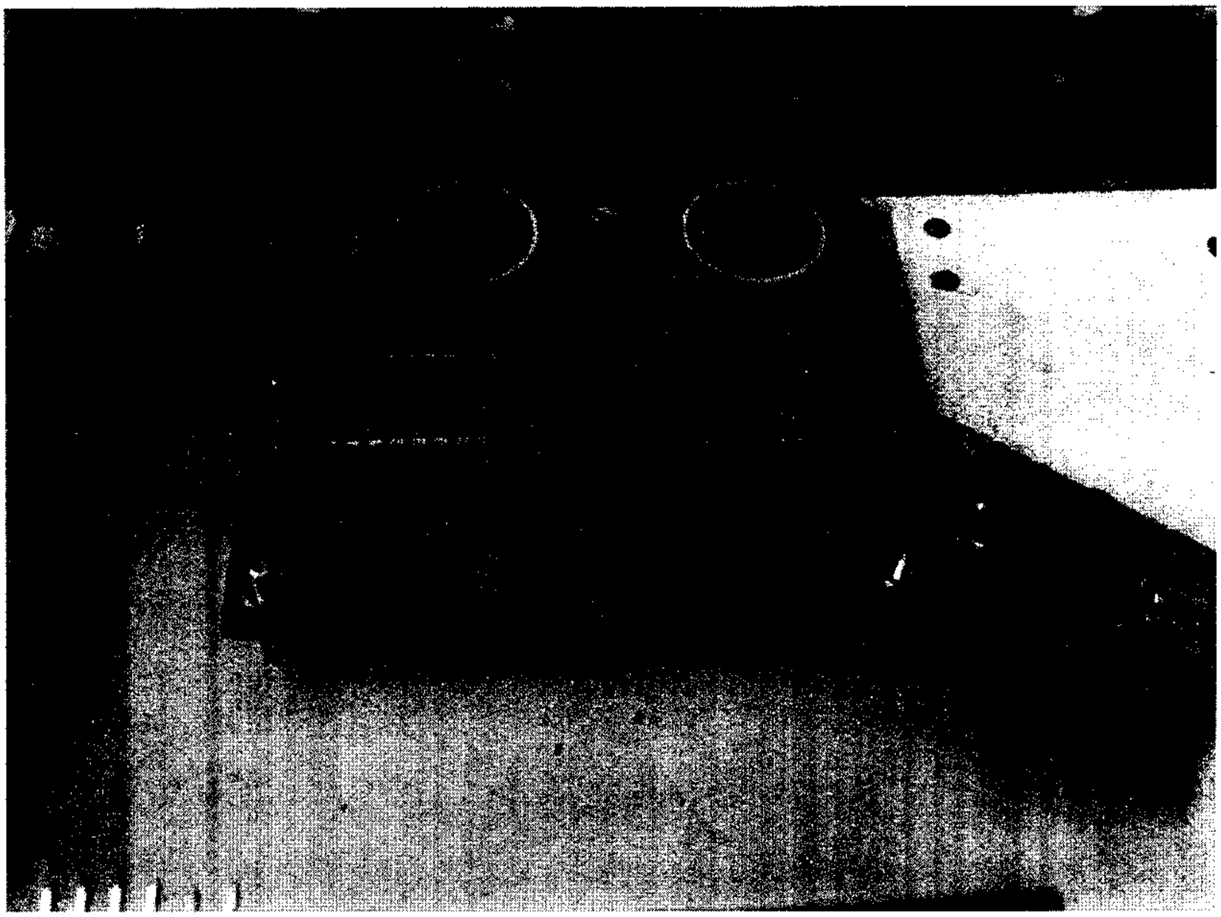
Stepper Motor



Power Supply Board 12 V

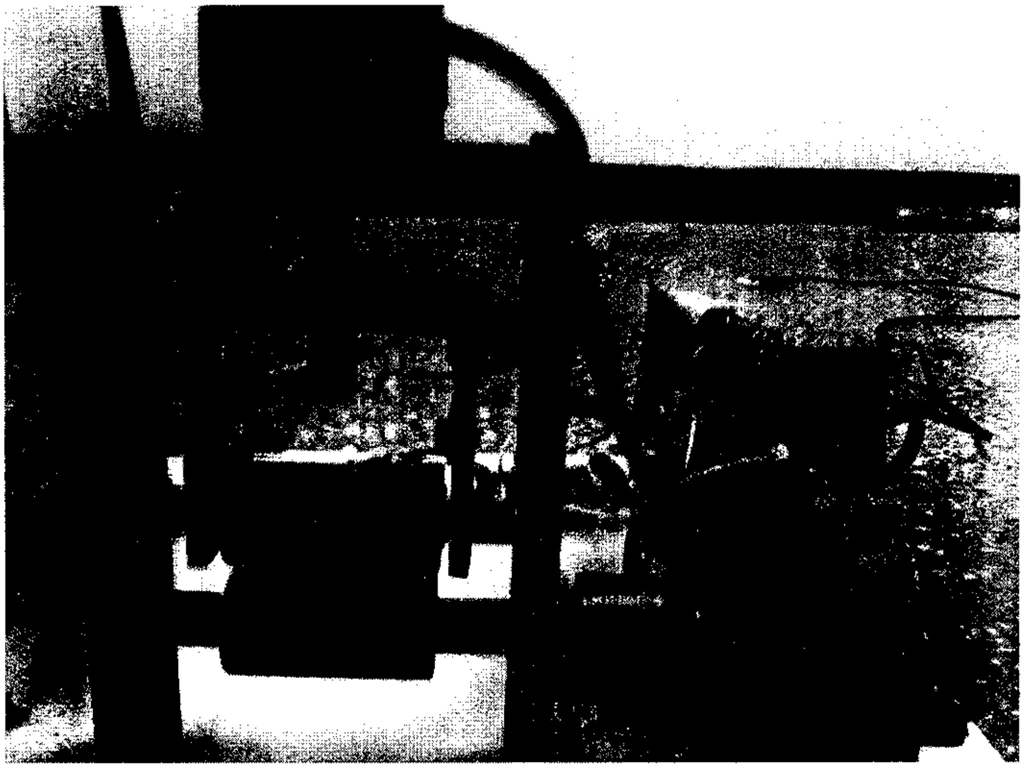


Ultrasonic Sensor Interface Board



Stepper Motor Interface and 12 V Power Supply for Motor

The Roller-Motor Mechanical Assembly



Bibliography

Bibliography

- **Handbook for Instrumentation Engineers – A.K Sawhney**
- **IEEE Paper on Sensors ISSN 1530-437X**
- **picprojects.org.uk/**
- **www.robot-electronics.co.uk/htm/srf04tech.htm**
- **SAIL process manuals**
- **www.wikipedia.com**
- **Steel industry I: Manufacturing System**

  By Tadao Kawaguchi,Kenji Sugiyama

References

References

- **SAIL Steel Plant Process Manual, Vol-6, Issue 35, Salem**
- **Other Industry Labs**