DESIGN AND IMPLEMENTATION OF AN OVER - CURRENT STATIC RELAY ON A PERSONAL COMPUTER

by

Ersam Rizali Raib

Submitted in Partial Fulfillment of the Requirements

for the degree of

Masters of Science

in

Electrical Engineering

program

08/30/1993

Advisor Date

9/8/93

Dean of the Graduate School Date

YOUNGSTOWN STATE UNIVERSITY

December, 1993

ABSTRACT

DESIGN AND IMPLEMENTATION OF AN OVER-CURRENT STATIC RELAY ON A PERSONAL COMPUTER

Ersam Rizali Raib

Master of science, Electrical Engineering
Youngstown State University, 1993

Application of the static relay in a system will improve the efficiency of the current transformer. The static relay does not use a contact mechanism. Static components are divided into two parts: 1) Passive Static and 2) Active Static. This thesis deals with the design and implementation of an over-current static relay on a personal computer.

The implementation of interfacing an over-current static relay with a modern integrated computerized system is presented. The design method significantly increases the accuracy of the current transformer. The use of the static components in this relay design is to improve the reactive speed of the relay as compared to the electromagnetic or conventional relays.

An over-current static relay is desired, and its continuation function on a personal computer is investigated. A digital computer program for simulating the system's response is developed not only for checking the correctness of the program itself, but also as a part of the over-current static relay design.

The concept of trust units with the graphical results and listing of the digital computer simulation are given in this thesis.

Acknowledgements

First, I wish to thank my thesis advisor Dr. Jalal Jalali for having given me the opportunity to work on my thesis project and for his full support, patience and hours of guidance.

I thank Dr. Salvatore R. Pansino, Chairman of Electrical Engineering Department, and Dr. Mathew Siman for reviewing this thesis and being on my committee. I must also acknowledge the assistance of Mrs. Anna Mae Serrecchio who was always willing to help me.

Finally, I dedicate this thesis to my wife, Connie, and to my parents, Jancy and Ina Raib, whose love, financial support and encouragement enabled me to complete my studies at Youngstown State University.

TABLE OF CONTENTS

Page No:

ABSTRACT ii
ACKNOWLEGDEMENTS iii
TABLE OF CONTENTS iv
LIST OF SYMBOLS vi
LIST OF FIGURES ix
LIST OF TABLES xi

Chapter One

1. INTRODUCTION 1
- **1. 1. Identification** 1
- **1. 2. Objective** 2
- **1. 3. Overview** 2

Chapter Two

2. OVER-CURRENT SYSTEM PROTECTION 3
- **2. 1. Over-current protection** 3
- **2. 2. Over-current relay** 3

Chapter Three

3. STATIC RELAYS 5
- **3. 1. Over-current static relay** 5

Chapter Four

4. DESIGN OF AN OVER-CURRENT STATIC RELAY 7
- **4. 1. Current transformer** 7
- **4. 2. Current transformer requirement** 7
- **4. 3. The circuit current reference** 9
- **4. 4. The circuit converter voltage divider** 9
- **4. 5. Operational amplifier** 13

4. 5. 1. Inverting op-amp 14
4. 5. 2. Op-amp comparator 15
4. 5. 3. Design of the op-amp detector 15
4. 6. Delay Unit 17
4. 7. Metra-Byte DAS-8 card interface 18
4. 7. 1. Installing the DAS-8 interface card 18
4. 7. 2. The realization of unit design with the DAS-8 interface 19

Chapter Five

5. ANALYSIS AND IMPLEMENTATION 20
5. 1. Design construction 20
5. 2. Gain control 24
5. 3. Comparator 26
5. 4. *RC* time delay 28
5. 5. DAS8 Module Box and card interface 29

Chapter Six

6. EXPERIMENTAL DESIGN 31
Experimental design 31
6. 1. Current reference 32
6. 2. Gain control 32
6. 3. Comparator 39
6. 4. Delay circuit 45
6. 5. Software design 45

Chapter Seven

7. CONCLUSION 48
Summary 48
APPENDIX 1 50
APPENDIX 2 57
APPENDIX 3 61
APPENDIX 4 69
APPENDIX 5 75
REFERENCES 96

LIST OF SYMBOLS

Symbol	Definition
A	Ampere
A	Internal gain of the operational amplifier
ac	Alternating current
β	Gain of the operational amplifier
C	Capacitor
CB	Circuit Breaker
CT	Current transformer
$^{\circ}C$	Degree Celsius
D	Diode
dc	Direct current
Δt	Time interval
E_i	Input voltage of the operational amplifier
E_o	Output voltage of the operational amplifier
e_o	Output voltage of the comparator
f	Frequency
I	Current
I_{ac}	Sinusoidal current
I_{CT}	Secondary current of current transformer
I_{dc}	Uni-directional current of the rectifier
I_i	Input current of the operational amplifier
I_m	Maximum current
I_{rms}	Sinusoidal current
I_L	Load current

I_{Line}	Primary current of current transformer
i_L	Load current
k	Constant value of resistor and capacitor
μ	micro (10^{-6})
m	milli (10^{-3})
Ω	Ohm (unit value of resistor)
ω	$2\pi f$
π	3.1415927
Q	Transfer charge of capacitor
R	Resistance
R_a	Internal resistance of diode
R_i	Input resistance of the gain control
R_f	Feedback resistance of the operational amplifier
R_L	Load resistance
r	Ripple factor
T	Time periodic
t	time
τ	*RC* constant value
θ	Conducting angle of the diode
V	Volt
VA	Unit of the active power
V_{ac}	Sinusoidal voltage
V_c	Voltage across the capacitor
V_{CT}	Voltage across the current transformer
V_{dc}	Uni-directional output voltage of the rectifier
V_i	Input voltage of the *RC* time-constant circuit
V_m	Maximum sinusoidal voltage
V_{out}	Output voltage of the converter voltage divider circuit
V_r	Ripple voltage
v_i	Input voltage of the operational amplifier (Appendix Three)

v_o	Output voltage of the operational amplifier (Appendix Three)
Watt	Unit of the real power
Z	Impedance
Z_{CT}	Current transformer impedance
Z_L	Load impedance

LIST OF FIGURES

Figure	Page
Chapter Two	
2-1. Block diagram of an over-current relay.	3
Chapter Four	
4-1. Voltage reference of the current transformer.	8
4-2. Converter voltage divider circuit.	13
4-3. Operational amplifier detector circuit.	15
4-4. Graphical result of the op-amp comparator with variable E_o.	16
4-5. Graphical result of the *RC* time delay circuit	17
Chapter Five	
5-1. Block diagram of the over-current static relay design.	20
5-2. Complete diagram of the over-current static relay design.	22
5-3. Flow chart of the digital computer simulation.	23
5-4. Inverting operational amplifier circuit.	25
5-5. Op-amp comparator circuit.	26
5-6. Graphic characteristics of output e_o.	27
5-7. Characteristic of the delay time and inrush effect.	28
5-8. Wiring connection between PI and DMB.	30
Chapter Six	
6-1. Complete block diagram of the design experimentation.	31
6-2. Current transformer equivalent.	33
6-3. Characteristic relations between the variation of R_f. and the delayed output of E_o.	33

6-4. Characteristic of the delayed output e_o proved by E_o. 39
6-5. Characteristic of the *RC* time-constant. 45
6-6. Final flow chart of the CLI design. 46

Chapter Seven

7-1. Characterictic of the over-current static relay design. 49

Appendix One

Figure 1. Diode. 50
Figure 2. Half-wave rectifier. 51
Figure 3. Center tap rectifier (phase inverter). 52
Figure 4. Full-wave bridge rectifier. 52
Figure 5. A capacitor filter. 54
Figure 6. Approximate analysis using a capacitor filter. 55

Appendix Two

Figure 1. Characteristic of the *RC* time-constant circuit. 59

Appendix Three

Fig 1. Basic op-amp. 61
Fig 2a. Inside op-amp. 62
Fig 2b. Op-amp equivalent circuit. 63
Fig 3. Zero input op-amp. 64
Fig 4. Inverting op-amp. 64
Fig 5. Summing inverting op-amp. 66
Fig 6. Non-inverting op-amp. 67
Fig 7. Unity gain op-amp. 67
Fig 8. Voltage to current converter. 68

LIST OF TABLES

Table **Page**

Chapter Three

Table 3-1. Comparison between the electromechanic and the static components. 6

Chapter Five

Table 5-1. Relationship between limit current of I_{CT} with variable of β. 26
Table 5-2. Correlation between level and time delay of e_o. 27

Chapter Six

Table 6-1. Gain relation between output E_o and the feedback resistor R_f. 32
Table 6-2. Delayed output E_o with variable of β. 33
Table 6-3. Delay time of the output e_o. 39
Table 6-4. Delayed output voltage of V_c. 45

1. INTRODUCTION

1.1. Identification

This thesis will introduce you to the fascinating and basically simple interface principles that are the basics of the applied over-current static relay, operational amplifier, and computers. Learning to use a current transformer and integrated circuit is like learning to play games on the computer. Pursuing the analogy further, one can't become a master of a computer game just by studying it, one must also play it. To develop skill, knowledge, and appreciation of an interface physical control of a computer, we have to combine theory and application.

A modest investment in a few essential pieces of equipment is recommended for this research. This research is also based on a formula commonly used in current transformers, rectifiers and operational amplifiers. Possibly the most common method of improving accuracy is to use a practice formula to implement current transformers through a computer. Most of these cases assume that the readers already have a profound knowledge of circuits and circuit design.

In addition, the research covers the working principles when using components and working out required values, and it describes in detail the basis for standard (current transformer, rectifier, voltage limiter, time delay, design op-amp comparator) circuits, and the necessary calculations needed to arrive at suitable component values that are used in this research.

When it comes to the computer interface card the problem is a little bit different, but most of the basic command program necessary to activate the interface is listed in Appendix 4. The application program itself is written in BASIC language and uses the

Microsoft Professional Basic Compiler to compile (produce file extension OBJ's) the program application. After compiling the program, it is then combined with the card interface driver program (using Microsoft Linker or Sourcer mixed language) to produce a stand alone program (with the extension name EXE).

In this case, the program must be specifically related to an individual card interface. This method will be adopted in describing an applied over-current static relay on a personal computer procedure.

1.2. Objective

Recently, the improvement of the personal computer has made it a necessary tool in many activities. This is not only because of its low power consumption, reasonable speed, and durability, but also because its flexibility makes the personal computer an important tool in many fields.

One of the interfaces called "DAS-8" can be used to make the personal computer applicable to a protection system. Therefore, it's important to use a personal computer to detect the over-current problems that result from power system contingencies.

1.3. Overview

This thesis is organized into four main parts. Chapter Two and Three cover the background knowledge of the basic role requirements of the over-current static relay protection system.

Chapter Four constitutes the second part which describes the pre-application of devices including current transformer requirement, rectification, *RC* time-constant, and the operational amplifier.

Development of the design and experimentation of the over-current static relay make up the third part, consisting of Chapter Five and Chapter Six. In Chapter Six, the numerical and the graphical results of the unit pre-interface of this over-current static relay system design are given.

Finally, for the fourth part, the conclusion and investigation of the graphical characteristics of the design and implementation of an over-current static relay on a personal computer are provided in Chapter Seven.

2. OVER-CURRENT SYSTEM PROTECTION

2.1. Over-current protection

When power systems were set-up, the need to add automatic protection systems was soon realized. Equipment responsive to excess currents was the obvious solution. Selectivity was soon needed and the purpose of an over-current relay had to evolve to give discriminating fault protection. Usually, an over-current protection is achieved by a correct utilization of the protective devices.

The over-current relay protection should be able to correctly recognize the need to clear the fault. As a result, settings are very important in realizing the objective of clearing the fault.

2.2. Over-current relay

The use of an over-current relay is one solution to protect an electrical line from over-current conditions. The relay must be able to detect the fault and send a command to activate the circuit breakers. Most protection relays have three units as follows:

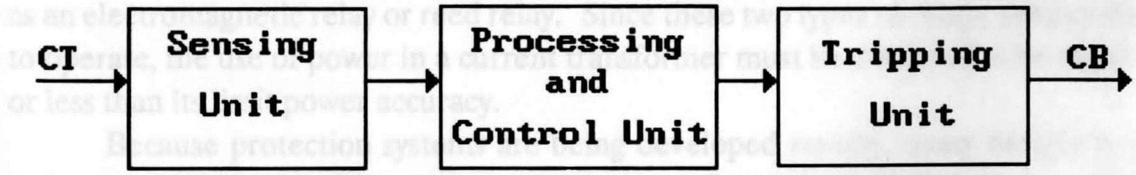


Figure 2-1. Block diagram of an over-current relay.

a. Sensing unit

This unit senses a change in the primary current of the current transformer.

b. Processing and control unit

It has the ability to process the input signal from the sensing unit and sends a command to activate the tripping unit.

c. Tripping unit

The purpose of this unit is to energize the circuit breaker.

The rating of a relay depends upon the relay reaction in response to different type of faults. Because some faults require clearance in a critical time interval, the time requirements for any relay is:

$$T_{op} = t + t_{CB}$$

T_{op} = Operation time required for any specific type of fault.

t = Length of time between the relay system receiving the input signal and sending an output command to the circuit breaker.

t_{CB} = Time for the circuit breaker to clear open.

A relay must be able to react fast enough to clear the fault from the system. The fast reaction of a relay in response to a short circuit plays an important role for the following reasons:

a. The length of time must be shorter or equal to the Critical Clearing Time (CCT).

b. Holding a short circuit on the power system for a duration of time greater than the Critical Clearing Time will damage the system.

Generally, a conventional over-current relay is operated by a mechanism such as an electromagnetic relay or reed relay. Since these two types of relays need current to operate, the use of power in a current transformer must be designed to be equal to or less than its limit power accuracy.

Because protection systems are being developed rapidly, many designers are trying to use different elements or components such as "Static Relay" to replace the electromagnetic relay and reed relay. The reason for replacing these two types is to reduce the use of power by the current transformer.

3. STATIC RELAY

3.1. Over-current static relay

Static relays are built from static components. They do not use contact mechanisms. Static components need only a small amount of power to operate, which will improve the performance accuracy of the current transformer.
Static components consist of:

1. **Active Static**
   - Tube.
   - Solid-state.
   - IC (Integrated circuit)
     * Linear.
     * TTL (Transistor-transistor logic).

2. **Passive static**
   - *R* (Resistor).
   - *L* (Inductor).
   - *C* (Capacitor).

It would be an advantage to use static components over conventional components because of their effectiveness. The comparison between electromechanic and static components is tabulated in Table 3-1.

Table 3-1. Comparison between the electromechanic and the static components.

Function	Electromagnet	Reed	Semiconductor	Thyristor
Input	1 to 3 Watts	0.1 to 3 Watts	10 *m*Watts	40 *m*Watts
Switching Capacity.	30 Watts	up to 20 Watts	50 Watts	1200 Watts
Power Gain	10 to 30	7 to 200	5000	30,000
Continuous Current Carrying	5 Amps	1 Amp	1 Amp	5 Amps
Delay	10 *m*Sec	1 or 2 *m*Sec	20 μSec	50 μSec
Contacts	up to 6	up to 6	1	1
Ambient temp^o Range	-5^o to 70^o*C*	-5^o to 55^o*C*	-20^o to 100^o*C*	-20^o to 100^o *C*
Affected by Vibration	Yes	Little	No	No
Affected by Corrosive, Atmosphere	Yes	No	No	No

4. DESIGN OF AN OVER-CURRENT STATIC RELAY

This design provides the formulability used in static components including passive static and active static. Some instrumentation devices are needed to design an over-current static relay. The main control unit design is determined. Adding a current transformer to this over-current static relay design will transform the high primary current to a low secondary working current.

In a protection system, the current transformer is used to monitor the current level of the electrical line.

4.1. Current transformer

A current transformer is an instrumentation device. Its rating depends upon the characteristics of the current transformer. Most manufacturers provide the formulas which specify the ratings for their current transformers. With modern protective-gear testing equipment, it is possible to rate the relayability of the current transformer when assessing its performance in the laboratory.

In fact, the current transformer accuracy represents the determination of its performance and also can influence its effective uses in protection systems.

4.2. Current transformer requirements

Protective current transformers provide the total burden in *VA* (Volt Ampere) at a rated current of the secondary circuit, including relay, and any other instruments. The winding load must be sufficiently below or equal to the secondary output capability of the current transformer. For example, *BS3938 classifies protective current transformers as 5P or 10P, corresponding to a maximum error of 5 percent or 10 percent, at the maximum secondary current.

Protective current transformers are specified in terms of *VA* (Volt Ampere) at a rated current, with a class and accuracy limit factor, e.g. 10 *VA*/5 P/15/CT ratio 300:5.

The impedance of burden (10 *VA* at rated current 5 Amperes) is:

$$Z_{Burden} = \frac{VA}{I^2} = \frac{10}{25} = 0.4\,\Omega$$

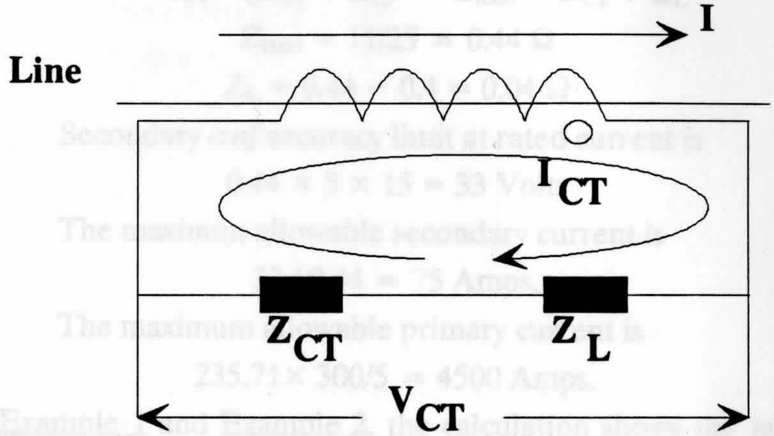


Figure 4-1. Voltage reference of the current transformer.

<u>Example 1.</u>

- Suppose that resistance of secondary winding is 0.1 Ohm.
- Therfore, total secondary impedance is 0.5 Ohm.
- Secondary *emf* accuracy limit at the rated current of the current transformer is

$$0.5 \times 5 \times 15 = 37.5 \text{ Volts.}$$

The maximum allowable secondary current is

$$37.5 / 0.5 = 75 \text{ Amps.}$$

The maximum allowable primary current is

$$75 \times \frac{300}{5} = 4500 \text{ Amps}$$

Justifying the amount of power of the current transformer is important, because power is related to the accuracy performance of the current transformer. Reducing the total power of the current transformer from 12.5 *VA* to 11*VA* at the same rated current increases the accuracy. The lowest power of 11*VA* for the maximum load is chosen to reduce the *emf* at the secondary side of the current transformer.

Example 2.

-Load impedance of the current transformer is

$$P = I_{CT}^{\,2} \cdot (Z_{CT} + Z_L) \rightarrow Z_{total} = Z_{CT} + Z_L$$

$$Z_{total} = 11/25 = 0.44\ \Omega$$

$$Z_L = 0.44 - 0.4 = 0.04\ \Omega$$

Secondary *emf* accuracy limit at rated current is

$$0.44 \times 5 \times 15 = 33 \text{ Volts}$$

The maximun allowable secondary current is

$$33 / 0.44 = 75 \text{ Amps.}$$

The maximum allowable primary current is

$$235.71 \times 300/5 = 4500 \text{ Amps.}$$

From Example 1 and Example 2, the calculation shows the incremental *emf* accuracy of the current transformer (it is increased about 113.6%).

4.3. The circuit current reference

The relay burden is quoted in *VA* (Volt Ampere) at the rated setting. The same energy must also be provided at the maximum winding load of the current transformer. This design provides the maximum power of the winding load about 1*VA*. The CT ratio is 300/5, followed by its data 10*VA*/5P/15. The rated current is 5 Amperes.

With the CT impedance (Z_{CT}) equal to 0.4 Ohm, the load impedance (Z_L) must be equal to 0.04 Ohm. The total impedance of 0.4 Ohm is chosen here to get the voltage (V_{CT}) across the current transformer, which is approximately 2.2 Volts .

Thus, to achieve a CT ratio 300/5 at the rated current of 5 Amps and with a stalled total burden of 11*VA*, this following formula is applied:

Total secondary impedance is 0.04 + 0.4 = 0.44 Ohm.

Secondary *emf* at accuracy limit current is

$$(0.4 + 0.04) \times 5 \times 15 = 33 \text{ Volts.}$$

Maximum allowable secondary current is = 33 / 0.44 = 75 Amps.

Maximum allowable primary current is = 75 × 300/5 = 4500 Amps.

Therefore, the secondary voltage at the nominal rated current (I_{CT}) is:

$$V_{CT} = (I_{CT}\, Z_L) + (I_{CT}\, Z_{CT}) = 2 + 0.2 = 2.2 \text{ Volts.}$$

4.4. The circuit converter voltage divider

This circuit is built from two units:

- **Unit rectifier.**
- **Filter unit.**

Unit rectifier

The nonlinear characteristic of a diode is used to convert alternating currents into unidirectional current. The conversion process is called rectification. A rectifier can be of two types:

Half-wave rectifier.

Full-wave rectifier.

Half-wave rectifier

The actual diode is represented by an ideal diode with a forward resistance and an internal resistance, where the input current or voltage is

$$I_{ac} = I_m \sin \omega t \quad or \quad V_{ac} = V_m \sin \omega t$$

The purpose of rectification is to obtain a unidirectional current or voltage. The dc component is the average value of the rectified current or voltage.

$$I_{dc} = \frac{1}{2\pi}\int_0^{2\pi} i\,d(\omega t) = \frac{1}{2\pi}\int_0^{\pi} \frac{V_m \sin \omega t}{(R_d + R_L)}\,d(\omega t) + 0$$

$$= \frac{V_m}{2\pi(R_d + R_L)}\left[-\cos \omega t\right]_0^{\pi} = \frac{V_m}{\pi(R_d + R_L)}$$

$$I_{dc} = \frac{I_m}{\pi}$$

or

$$V_{dc} = \frac{V_m}{\pi}$$

In practice, the dc component is approximately 30% of the maximum value.

Full-wave rectifier

The full-wave (bridge) rectifier provides a greater dc value from the same input voltage. The dc component is twice as large as in the half-wave rectifier or

$$I_{dc} = \frac{2\,I_m}{\pi}$$

$$V_{dc} = \frac{2\,V_m}{\pi}$$

where

$$V_m = \sqrt{2}\cdot V_{ac}$$

$$I_m = \sqrt{2}\cdot I_{ac}$$

Filter unit

The desired result of rectification is direct current, but the output currents of the rectifier circuits described obviously still contain large alternating components along with the dc component. As a measure of the effectiveness of rectification one defines the ripple factor (r)

where $$r = \frac{I_{ac} \text{ or } I_{rms}}{I_{dc}} = \frac{V_{ac} \text{ or } V_{rms}}{V_{dc}}$$

If the ripple factor is low, the circuit is performing the conversion from ac to dc effectively. The ripple factor can be reduced by putting a capacitor across its output. So the ripple voltage is:

$$V_r = \frac{I_{dc}}{fC}$$

where f is the frequency and C is the capacitor, or the rectified voltage maximum is:

$$V_m = V_{dc} + \frac{I_{dc}}{2fC}$$

Design of the converter voltage divider circuit

The converter voltage divider circuit will rectify the signal from the current reference circuit and reduce its ripple and limit its output to approximately 1 Volt.

This design, which has a rated current of 5 Amps with a winding load impedance of 0.4 Ohm, will produce a voltage V_{CT} = 2.2 Volts.

So that $V_m = \sqrt{2.2} \cdot V_{ct} = 1.4142 \times 2.2 = 3.11$ Volts

$$V_{dc} = \frac{2\,V_m}{\pi} = \frac{6.22}{3.14} = 1.98 \text{ Volt}$$

Suppose the design requires a dc output voltage of

$$V_{out} = 1 \text{ Volt}$$

Where

Capacitor (C) = 470 μF
Resistor (R_1) = 500 Ω
Frequency (f) = 60 Hz

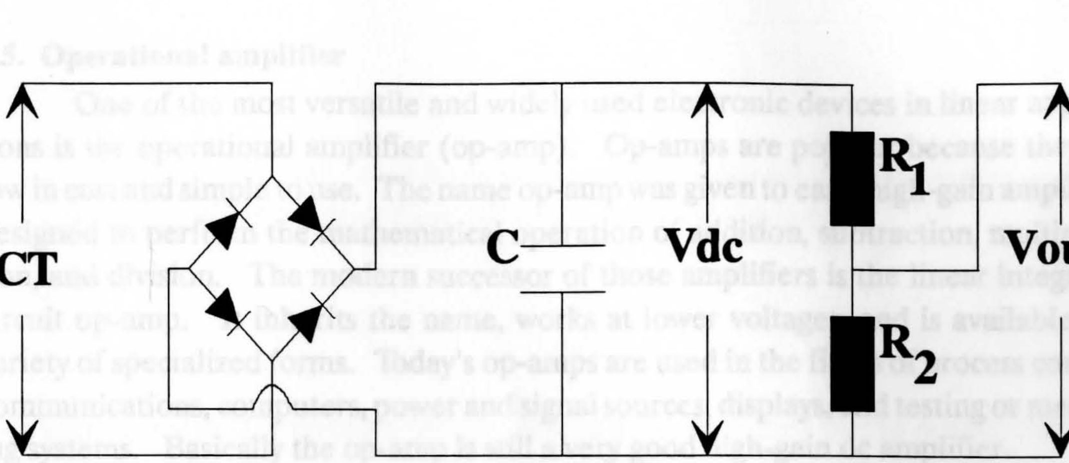


Figure 4-2. Converter voltage divider circuit.

$$V_{out} = V_{dc}\frac{R_2}{R_1 + R_2}$$

$$R_1 + R_2 = V_{dc}\,R_2 \;\rightarrow\; R_1 = V_{dc}\,R_2 - R_2$$

$$R_1 = 1.98\,R_2 - R_2$$

$$500 = 0.98\,R_2 \;\rightarrow\; R_2 = \frac{500}{0.98} = 510.2\ \Omega$$

$$I_{dc} = \frac{V_{dc}}{R_1 + R_2} = \frac{1.98}{1010.2} = 1.96\cdot 10^{-3}\ \text{Amp} \equiv 1.96\ \text{mA}$$

The ripple voltage is $V_r = \dfrac{I_{dc}}{fC} = \dfrac{1.96\cdot 10^{-3}}{60\cdot 470\cdot 10^{-6}} = 0.0695$ Volt

In practice, the effective value of ripple voltage should be between 2% to 10% of the ac value that goes along with the dc component. A benefit of reducing the ripple voltage in this design is the prevention of a spike voltage.

4.5. Operational amplifier

One of the most versatile and widely used electronic devices in linear applications is the operational amplifier (op-amp). Op-amps are popular because they are low in cost and simple to use. The name op-amp was given to early high-gain amplifiers designed to perform the mathematical operation of addition, subtraction, multiplication, and division. The modern successor of those amplifiers is the linear integrated circuit op-amp. It inherits the name, works at lower voltages, and is available in a variety of specialized forms. Today's op-amps are used in the fields of process control, communications, computers, power and signal sources, displays, and testing or measuring systems. Basically the op-amp is still a very good high-gain dc amplifier.

4.5.1. Inverting op-amp

The inverting operational amplifier is one of the most widely used in op-amp circuits. The input voltage (E_i) is applied through input resistance (R_i) to the op-amp's (-) input. Negative feedback is provided by any feedback resistor R_f to the negative op-amp input. Because one side of R_i is at E_i and the other is at 0 Volts, the voltage drop across R_i is E_i. The input current (I_i) through R_i is found from Ohm's law:

$$I_i = \frac{E_i}{R_i}$$

$$V_{R_f} = I_i R_f = \frac{E_i}{R_i} R_f$$

The current direction here is established by the input voltage forcing the right side of the feedback resistor R_f to go negative. Therefore, the output voltage (E_o) is negative when the input voltage is positive.

$$E_o = -E_i \frac{R_f}{R_i}$$

The close-loop gain from E_i to E_o of this amplifier is set by R_f and R_i. It can amplify the ac or the dc signal (Appendix 3). The gain (β) of the operational amplifier is defined:

$$\beta = \frac{E_o}{E_i} = -\frac{R_f}{R_i}$$

The load current (I_L) that flows through load resistance (R_L) is determined only by R_L and E_o and is furnished from the op-amp's output terminal.

$$I_L = \frac{E_o}{R_L}$$

The input current (I_i) through the R_f must also be furnished by the output terminal. Note, the output of I_o is set by the op-amp itself and is usually between 5 to 10 milliAmperes.

4.5.2. Op-amp comparator

The op-amp comparator compares a signal of one input with a reference signal of another input. Voltage level detectors use op-amps to solve some types of signal comparison applications.

Output e_o will occur if (negative input E_o) + (positive input V_{ref}) $\leq$ 0. The time to reach the maximum value of e_o depends upon the characteristic slew rate of the op-amp.

4.5.3. Design of the op-amp detector

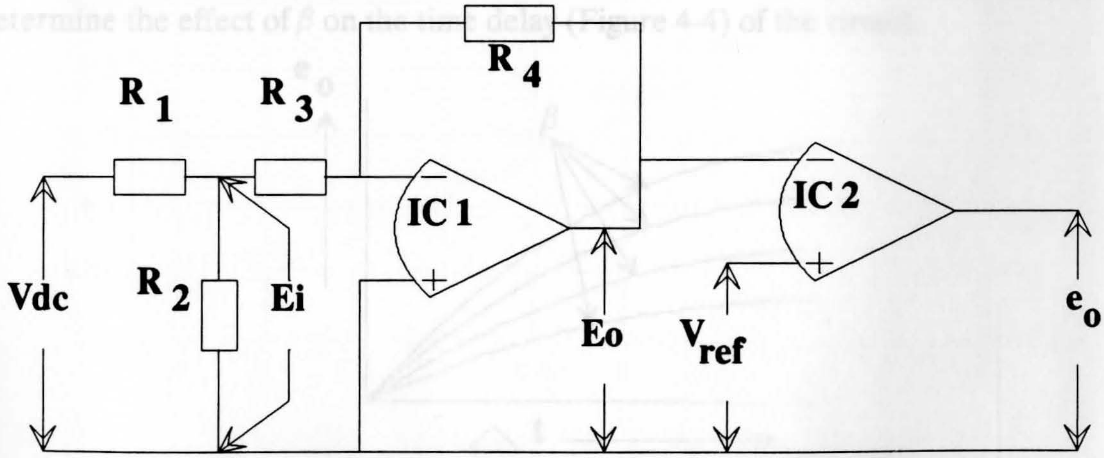


Figure 4-3. Operational amplifier detector circuit.

$$E_i = V_{dc}\left(\frac{R_2}{R_1 + R_2}\right) \quad \text{where} \quad E_o = E_i\left(-\frac{R_4}{R_3}\right)$$

$$E_o = \left(V_{dc}\frac{R_2}{R_1 + R_2}\right)\cdot\left(-\frac{R_4}{R_3}\right) \rightarrow E_o = V_{dc}\left(-\frac{R_2 R_4}{R_1 R_3 + R_2 R_3}\right)$$

Note, e_o is present if $E_o + V_{ref} \leq 0$ and IC_2 will not have an output e_o as long $E_o \leq 0$ or $E_o < (-V_{ref})$.

From the previous solution (converter voltage divider) one has established the value of V_{out} equal to 1 Volt. By substituting V_{out} for E_i , one now has:

$$E_o = V_{out}\left(-\frac{R_4}{R_3}\right)$$

where $$\beta = -\frac{R_4}{R_3} \rightarrow E_o = -V_{out}\beta$$

if $\beta = 1$ then, E_o is equal to V_{out}. The amplification of E_o depends on β.

The output E_o is compared to V_{ref}. If $E_o \leq V_{ref}$ (in an op-amp comparator), the output e_o becomes equal to the supply voltage. The output e_o represents the time delay from the minimum to the maximum value. By using the computer circuit analysis program (Micro Cap by Spectrum Advanced Circuit Analysis), one would be able to determine the effect of β on the time delay (Figure 4-4) of the circuit.

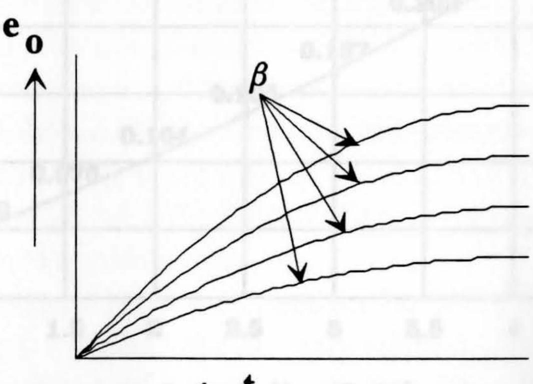


Figure 4-4. Graphical results of the op-amp comparator with variable E_o.

4.6. Delay unit

The design uses the delay unit as an option. The delay unit here is only used to delay the action of the circuit breaker, because some types of apparent faults require a time delay to prevent the system from incorrectly tripping the circuit breaker, for example, when sensing the inrush effect of an inductive load.

The delay unit in this design uses the *RC* time-constant circuit. The voltage across a capacitor increases as the charge builds up. After one time-constant the voltage becomes approximately 63% of the source voltage. The expression for the voltage across the capacitor is shown in Appendix 2.

The formula that defines the time t as a function of V_i

$$t = RC \ln \frac{V_i}{V_i - V_c}$$

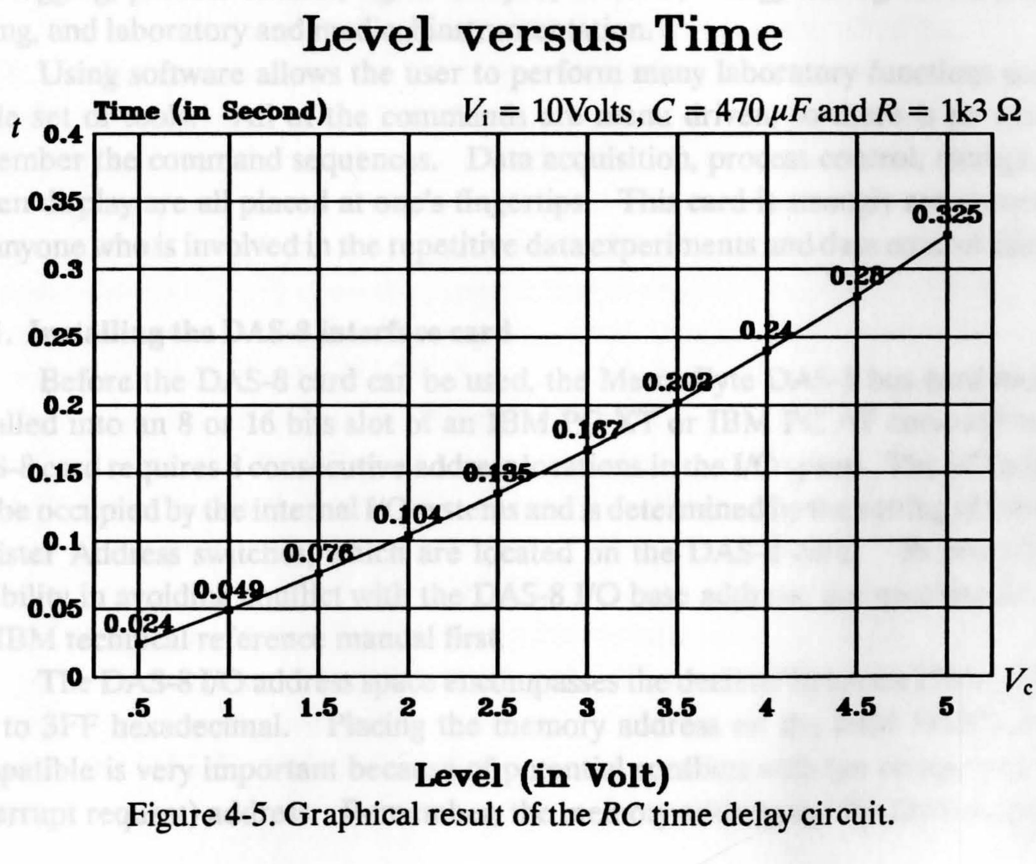


Figure 4-5. Graphical result of the *RC* time delay circuit.

4.7. Metra-Byte DAS-8 card interface

Metra-Byte's DAS-8 interface board contains 8 analog input channels, 4 digital input channels, and 3 digital output channels. This board is IBM PC bus compatible and features a high speed 12 bit successive approximation A/D (Analog / Digital) converter with a conversion time of 25 μSec (typically 35 μSec maximum), resulting in data throughput rates in excess of 30 KHz. This card also provides 7 TTL(Transistor-Transistor Logic) / CMOS compatible digital I/O (Input / Output) lines. All of the pin connections are made via a standard 37 pin D male connector, which is connected through the rear panel of the DAS-8 interface card.

DAS-8 provides a fix ± 5 Vdc input with a resolution to 0.00244 Volt, using a common ground, and it can withstand continuous voltage overloads of ± 30 Vdc with brief transients to several hundred volts.

The DAS-8 has been designed to provide power so that the interface card can provide versatile solutions for the most demanding applications. Applications include data logging, process control, signal analysis, robotics, energy management, product testing, and laboratory and medical instrumentation.

Using software allows the user to perform many laboratory functions using a single set of tools. All of the commands are menu driven, so there is no need to remember the command sequences. Data acquisition, process control, storage, and screen display are all placed at one's fingertips. This card is strongly recommended for anyone who is involved in the repetitive data experiments and data control analysis.

4.7.1. Installing the DAS-8 interface card

Before the DAS-8 card can be used, the Metra-Byte DAS-8 bus card must be installed into an 8 or 16 bits slot of an IBM PC XT or IBM PC AT compatible.The DAS-8 card requires 8 consecutive address locations in the I/O space. The I/O address will be occupied by the internal I/O systems and is determined by the setting of the Base Register Address switches, which are located on the DAS-8 card. To provide the flexibility in avoiding conflict with the DAS-8 I/O base address, the user should read the IBM technical reference manual first.

The DAS-8 I/O address space encompasses the decimal between 256 to 1023 or 100 to 3FF hexadecimal. Placing the memory address on the IBM PC/XT or AT compatible is very important because of potential conflicts with the computer's IRQ (interrupt request) address. Remember, the memory address for the DAS-8 card has to be separate from the I/O address in the computer that is already in use to avoid a conflict with any add-on memory inside the computer.

Usually, the best address for the card is either hexadecimal 300,308,310 (&H300, &H308, &H310) or decimal 768, 776, and 784. Note, if there is an IBM prototype board installed, and it uses the hexadecimal address from 300 to 31FF, the DAS-8 card should use a different address location (H330 or H340). By running the utility program called "UTIL.EXE," the base address on the DAS-8 card can be set. If this is not done, a conflict with another add-on card will result, or the computer might hang up.

4.7.2. The realization of unit design with the DAS-8 interface

The manual instructions of the DAS-8 interface card should be studied before the DAS-8 interface card is used as a part of the over-current static relay design. This interface card allows the user to use many different computer languages to communicate.

The manual of the interface card will introduce the user to all of the significant BASIC commands to activate the I/O (Input-Output) card. Two driver programs are needed when one activate the DAS-8 interface card. One driver program is named DAS8.BIN, and the other one is named DAS8.OBJ. Both of the driver programs are saved in a binary type of file. The driver file with the extension name BIN only allows the programmer to call the driver program via BASIC, and the driver file with the extension name OBJ will allow the programmer to combine the driver with any other computer languages. Usually the program that is already compiled runs faster than the program that runs under BASIC, because the compiled program is already translated into machine code language. The advantage of using the driver with the extension name OBJ is for the flexibility of the programmer to build programs using many computer languages, such as Assembly, Fortran, Pascal or C, and then to combine them with the driver program. The combining of many computer languages requires a computer program tool called "LINK.EXE," known as Microsoft Linker Software.

The program to interface an over-current static relay via DAS8 interface card to a personal computer was written using the Microsoft Quick BASIC command language together with Assembly compiler call (MASM Ver 5.1). All of the programs, including the DAS-8 interface card driver program, are combined using a Microsoft linker.

5. ANALYSIS AND IMPLEMENTATION

Block diagram of the over-current static relay is shown in Figure 5-1. The information from the Delay Circuit and PI is used as the main information for DAS-8 bus interface.

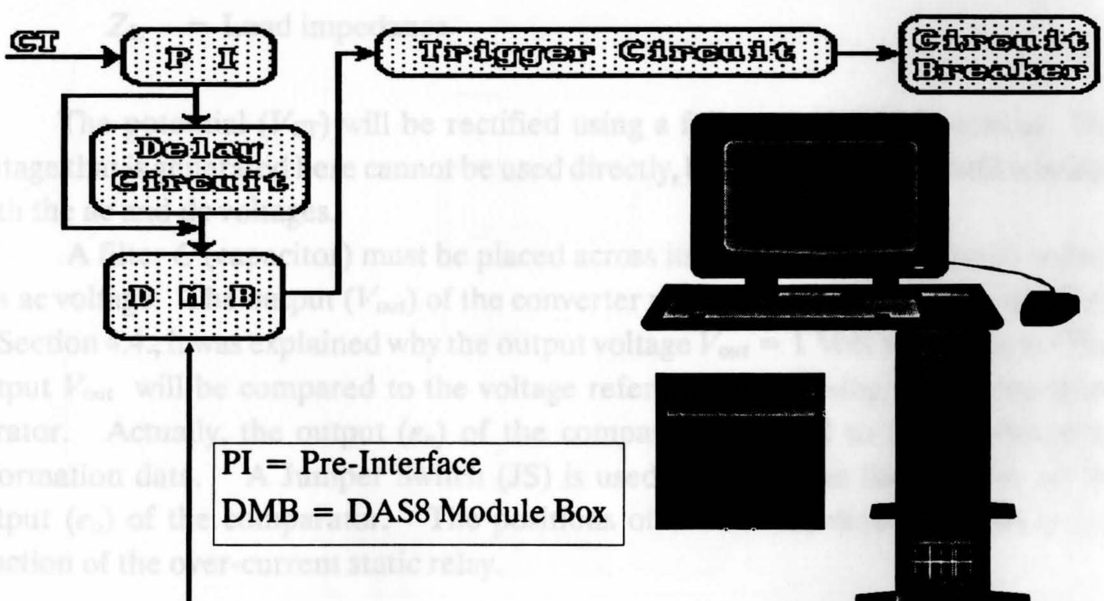


Figure 5-1. Block diagram of the over-current static relay.

5.1. Design constructions

Construction of the over-current static relay design is shown in Figure 5-2. Gain op-amp is utilized in this design to determine the allowable limit for the secondary current (I_{CT}) of the current transformer. The decision of the secondary current of the current transformer depends upon the value of the feedback resistance (R_f).

In this design, the current (I_{CT}) from the current transformer is utilized for producing the main information. By putting a resistor (Z_L) across the current transformer, one is able to get the voltage (V_{CT}). The voltage output of the current transformers depends upon the value of the resistor Z_L.

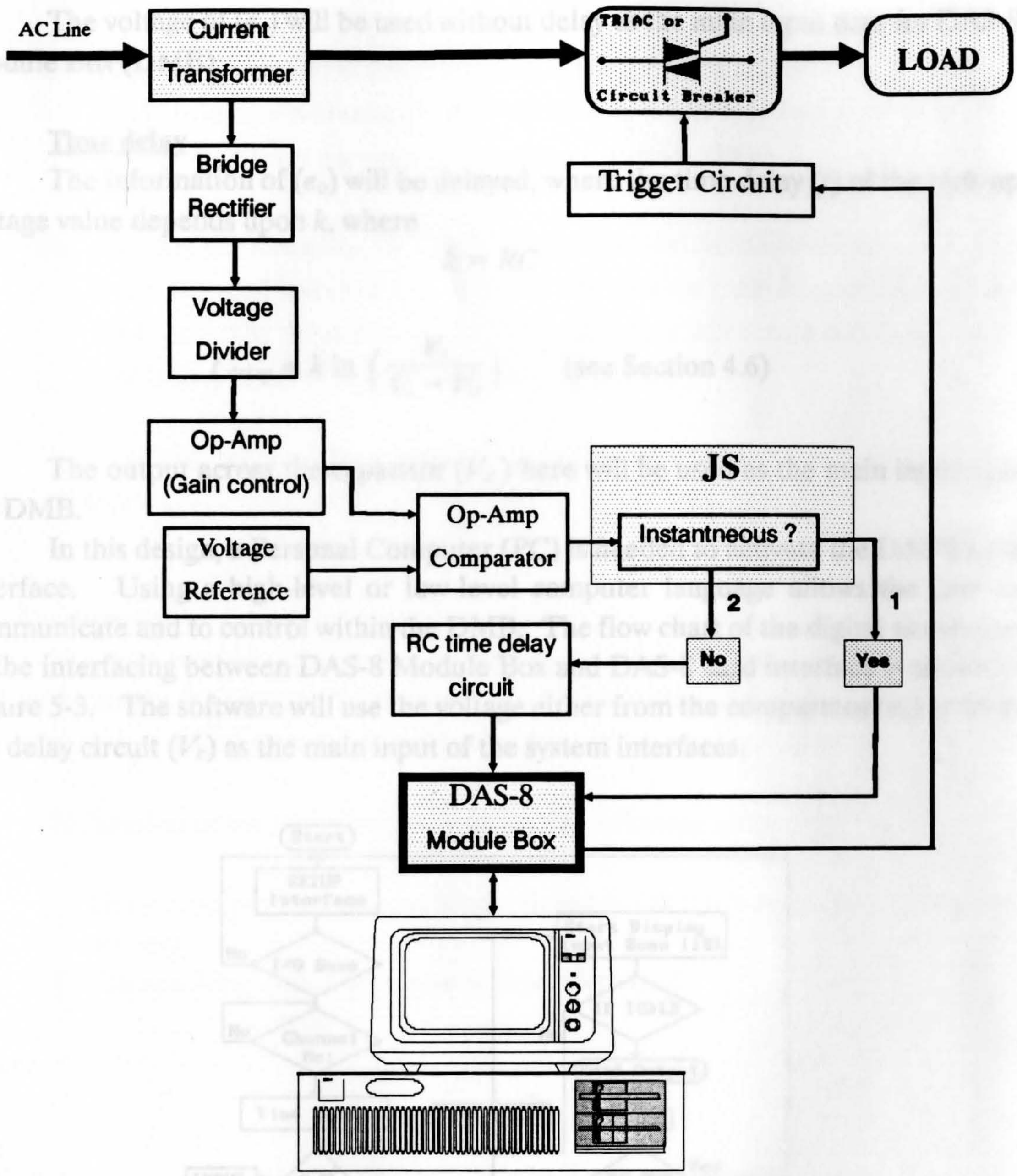
Where $$V_{CT} = I_{CT} Z_{CT} + I_{CT} Z_L = I_{CT} (Z_{CT} + Z_L)$$

V_{CT} = Voltage of the current transformer
I_{CT} = Secondary current of the current transformer
Z_{CT} = Impedance of the current transformer
Z_L = Load impedance

The potential (V_{CT}) will be rectified using a full-wave (bridge) rectifier. The voltage that is generated here cannot be used directly, because the voltage still contains both the ac and dc voltages.

A filter C (capacitor) must be placed across its output (V_{dc}) in order to reduce this ac voltage. The output (V_{out}) of the converter voltage divider is equal to one volt. In Section 4.4., it was explained why the output voltage V_{out} = 1 Volt was chosen. This output V_{out} will be compared to the voltage reference (V_{ref}) using an op-amp comparator. Actually, the output (e_o) of the comparator is used to inform the main information data. A Jumper Switch (JS) is used to determine the function of the output (e_o) of the comparator. The positions of the JS will determine the actual function of the over-current static relay.

- JS in position 1 will assume the relay function as an Instantaneous Relay
- JS in position 2 will assume the relay function as a Time Delay Relay



Personal Computer

Figure 5-2. Complete diagram of the over-current static relay design.

Instantaneous

The voltage of (e_o) will be used without delay as the main input data for DAS-8 Module Box (DMB).

Time delay

The information of (e_o) will be delayed, where the time delay (t) of the pick-up voltage value depends upon k, where

$$k = RC$$

or

$$t_{\text{delay}} = k \ln \left(\frac{V_i}{V_i - V_c} \right) \quad \text{(see Section 4.6)}$$

The output across the capacitor (V_c) here will be used as the main input data for DMB.

In this design, a Personal Computer (PC) is needed to activate the DAS-8 card interface. Using a high-level or low-level computer language allows the user to communicate and to control within the DMB. The flow chart of the digital simulation of the interfacing between DAS-8 Module Box and DAS-8 card interface is shown in Figure 5-3. The software will use the voltage either from the comparator (e_o) or from the delay circuit (V_c) as the main input of the system interfaces.

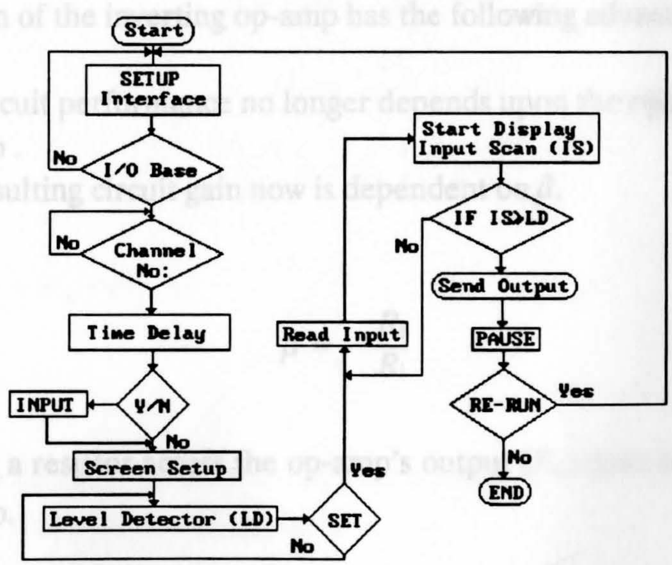


Figure 5-3. Flow chart of the digital computer simulation.

During the operation, the PC will take control of all activities including:

- Communicating and Controlling the DAS-8 Module Box (DMB).
  DAS8 interface will scan all of the data using DMB to process and to control the changed level parameter of the Pre Interface (PI).

- Automatically commanding the DMB to energize the circuit breaker.
  If the changed level parameter input information from PI goes higher than the level set by the software, the DAS8 interface will send a command to the DMB to give an output to the switching circuit.

In addition, the switching circuit is used to disconnect the load from the electrical source.

5.2. Gain control

The gain control unit uses an op-amp as an amplifier. An external feedback resistor connects the output (E_o) terminal and (-) input terminal of the op-amp. This type of circuit is called a negative feedback circuit or an inverting op-amp.

Utilization of the inverting op-amp has the following advantage:

1. The circuit performance no longer depends upon the open-loop gain of the op-amp .
2. The resulting circuit gain now is dependent on β.

where

$$\beta = -\frac{R_f}{R_i}$$

Note that adding a resistor across the op-amp's output (E_o) does not change the gain (β) of the op-amp.

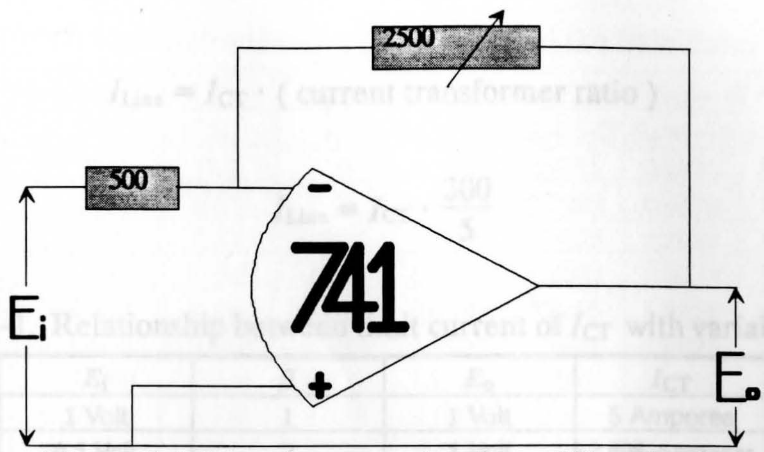


Figure 5-4. Inverting operational amplifier circuit.

This op-amp design uses a general-purpose operational amplifier type LM 741, one fixed resistor type carbon/half watt ($R_i = 0.5\,k\,\Omega$), and one variable carbon resistor type linear ($R_f = 2.5\,k\,\Omega$). Using the formula that was already described in Section 4.5.1., now the result is:

$$E_i = V_{out} = V_{dc}\frac{R_2}{R_1 + R_2}$$

where $R_1 = 500\,\Omega$, $R_2 = 510\,\Omega$

$$V_{dc} = E_i\frac{R_1 + R_2}{R_2}$$

$$V_{dc} = \frac{2\,V_m}{\pi}$$

$$V_m = \frac{V_{dc}\,\pi}{2} = \pi\,E_i\left[\frac{R_1 + R_2}{2\,R_2}\right]$$

Note, where $V_{CT} = V_m / \sqrt{2}$ and

$$I_{CT} = \frac{V_{CT}}{Z_{CT} + Z_L}$$

Assuming that the current transformer has a ratio 300/5, then the current of the primary side of the current transformer (I_{Line}) is:

$$I_{Line} = I_{CT} \cdot (\text{ current transformer ratio })$$

$$I_{Line} = I_{CT} \cdot \frac{300}{5}$$

Table 5-1. Relationship between limit current of I_{CT} with variable of β.

V_m	E_i	β	E_o	I_{CT}	I_{Line}
3.11 Volts	1 Volt	1	1 Volt	5 Amperes	300 Amperes
1.555 Volt	0.5 Volt	2	1 Volt	2.498 Amperes	149.8 Amperes
1.035 Volt	0.333 Volt	3	1 Volt	1.66 Ampere	99.6 Amperes
0.77 Volt	0.25 Volt	4	1 Volt	1.23 Ampere	73.8 Amperes
0.622 Volt	0.2 Volt	5	1 Volt	0.99 Ampere	59.4 Amperes

5.3. Comparator

Different general-purpose op-amps are available in industry for building the op-amp comparators. The most common op-amp comparator used in industry is LM/CA/μA 311. This op-amp has a low slew rate and low power consumption.

In this design, op-amp type 311 is used as a comparator. The comparator here will perform as a comparing switch, where the input voltage from the op-amp gain control (E_o) will be compared with the voltage reference (V_{ref}) to get the output e_o. To reduce the effect capacitance (slew rate) of the op-amp comparator, the voltage E_o must be increased. By using the digital computer circuit analysis program called "Micro Cap" from Spectrum, the output of the comparator (e_o) is determined (see Table 5-2).

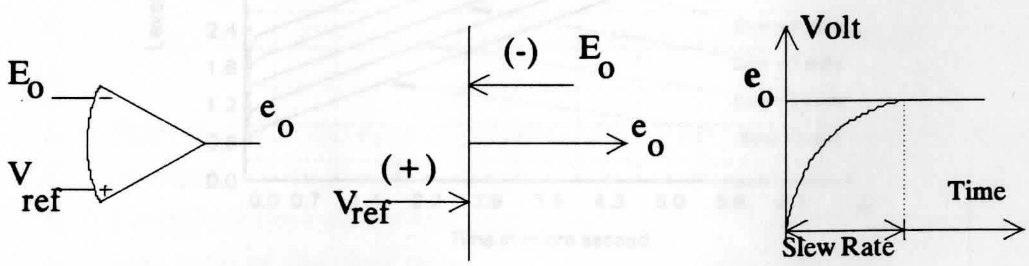


Figure 5-5. Op-amp comparator circuit.

Table 5-2. Correlation between the level and the time delay of e_o.

Time in μ second					V_{ref} = 1 and output of e_o in Volt				
t_1	t_2	t_3	t_4	t_5	$E_o = -1$	$E_o = -2$	$E_o = -3$	$E_o = -4$	$E_o = -5$
0.0	0.0	0.0	0.0	0.0	0.2887	0.433	0.5773	0.7216	0.866
0.01	0.01	0.01	0.01	0.01	0.4699	0.7039	0.9379	1.1719	1.406
0.038	0.024	0.22	0.21	0.019	0.6615	0.9403	1.2083	1.4687	1.7237
0.11	0.067	0.05	0.04	0.034	0.7814	1.0989	1.4076	1.7034	1.988
0.41	0.191	0.114	0.08	0.061	0.9489	1.2105	1.5324	1.8602	2.1797
1.306	0.742	0.372	0.207	0.133	1.3972	1.4921	1.6826	1.9722	2.2996
2.305	1.721	1.229	0.774	0.433	1.8961	1.981	2.1120	2.2613	2.4669
3.305	2.721	2.228	1.755	1.329	2.3952	2.48	2.6105	2.7512	2.915
4.305	3.721	3.228	2.755	2.329	2.8941	2.979	3.1094	3.2501	3.4139
5.305	4.721	4.228	3.755	3.329	3.393	3.4778	3.6082	3.7489	3.9127
6.305	5.721	5.228	4.755	4.329	3.8918	3.9766	4.107	4.2476	4.4114
7.305	6.721	6.228	5.755	5.329	4.3904	4.4752	4.6056	4.7463	4.91
8.305	7.721	7.228	6.755	6.329	4.889	4.973	5.1042	5.2448	5.4085
9.305	8.721	8.228	7.755	7.329	5.3875	5.4723	5.6026	5.7432	5.9069
10	9.721	9.228	8.755	8.329	5.7338	5.9707	6.101	6.2416	6.4052

Table 5-2 shows the relationship between E_o and e_o with a 1 volt reference using Micro Cap circuit analysis program.

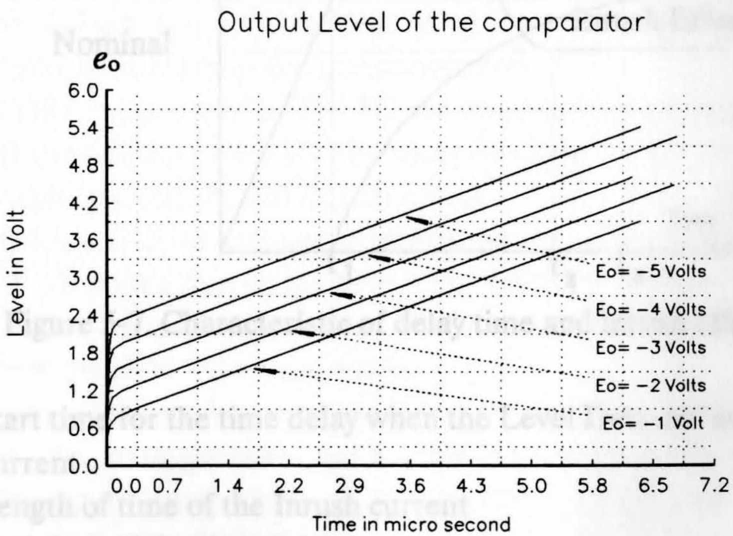


Figure 5-6. Graphical characteristics of output e_o.

The graphical result of the e_o (Figure 5-6) proves that the delay time of the comparator's output will depend upon the input E_o from the gain control output. These outputs e_o with the delay time ($t_1, t_2, \ldots t_5$) are also shown in the Table 5-2.

5.4. *RC* time delay

In this design, the *RC* time delay is used as an optional function. The *RC* time-constant circuit is used to delay the operation of the circuit breaker because most of an inductive loads automatically generate an inrush effect.

Over level during inrush current condition causes the relay to work because the relay senses only the target level that has been set for a certain system. The characteristic relationship between an inrush effect and delay time is shown in Figure 5-7.

To prevent the relay operation during inrush effect, delay the action of the relay. t_1 to t_2 is the periodic time of the relay reading the inrush effect. In t_1 the relay begins to react, but this condition is then delayed to t_3. Because the delay period is longer than the reading period of the relay, the relay will terminate the operation of the circuit breaker.

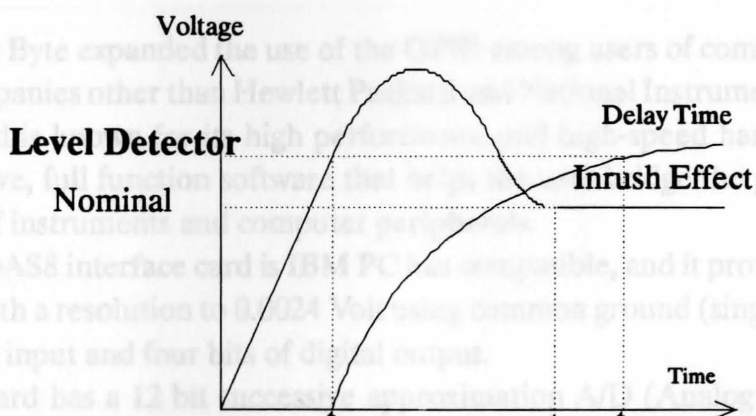


Figure 5-7. Characteristic of delay time and inrush effect.

t_1 = Start time for the time delay when the Level Detector senses the inrush current

t_2 = Length of time of the Inrush current

t_3 = End time of the time delay

5.5. DAS8 Module Box and interface card

The DAS8 Module Box (DMB) is used to connect the DAS8 interface card and the Pre-Interface (PI), including the comparator and time delay circuit. The DMB will take all the information from the PI and input it to the Personal Computer (PC) via the DAS8 interface card.

The most common interface card that can be used to interface between a Personal computer and instrumentation equipment is called "GPIB" (General Purpose Interface Bus).

In 1965 Hewlett Packard designed the Hewlett Packard Interface Bus (HP-IB), an interface bus to connect and to control their product line of programmable instruments. Because of its high data transfer rates, this interface bus quickly gained popularity in other applications, such as intercomputer communications and peripheral control.

This interface bus was later accepted by the industry as IEEE Standard 488-1978. The versatility of this bus prompted the name General Purpose Interface Bus (GPIB).

Metra Byte expanded the use of the GPIB among users of computers manufactured by companies other than Hewlett Packard and National Instruments. The DAS8 interface card is known for its high performance and high-speed hardware, and as a comprehensive, full function software that helps the user bridge the gap between the knowledge of instruments and computer peripherals.

The DAS8 interface card is IBM PC bus compatible, and it provides a fixed +/- 5 V_{dc} input with a resolution to 0.0024 Volt using common ground (single ended), three bits of digital input and four bits of digital output.

The card has a 12 bit successive approximation A/D (Analog to Digital) converter with a 25 μSeconds conversion time. The interface features a highly advanced Intel 8254 timer/counter providing 3 times 16 bits count-down register while it uses the clock cycles from the IBM PC system clock.

Using state of the art data conversion components, the DAS8 has been designed to provide powerful, analog/digital interface which provides versatile solutions for the most demanding applications, including process control, signal analysis, relay protection and laboratory testing equipment.

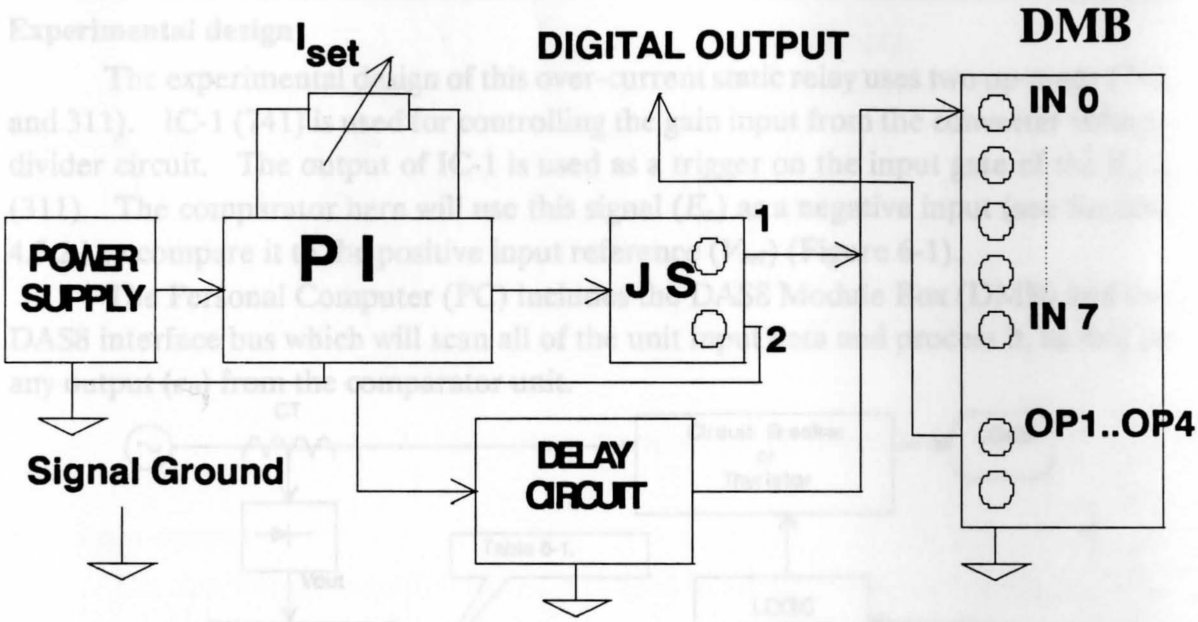


Figure 5-8. Wiring connection between PI and DMB.

6. EXPERIMENTAL DESIGN

Experimental design

The experimental design of this over-current static relay uses two op-amps (741 and 311). IC-1 (741) is used for controlling the gain input from the converter voltage divider circuit. The output of IC-1 is used as a trigger on the input gate of the IC-2 (311). The comparator here will use this signal (E_o) as a negative input (see Section 4.5.2.) to compare it to the positive input reference (V_{ref}) (Figure 6-1).

The Personal Computer (PC) includes the DAS8 Module Box (DMB) and the DAS8 interface bus which will scan all of the unit input data and process it, as well as any output (e_o) from the comparator unit.

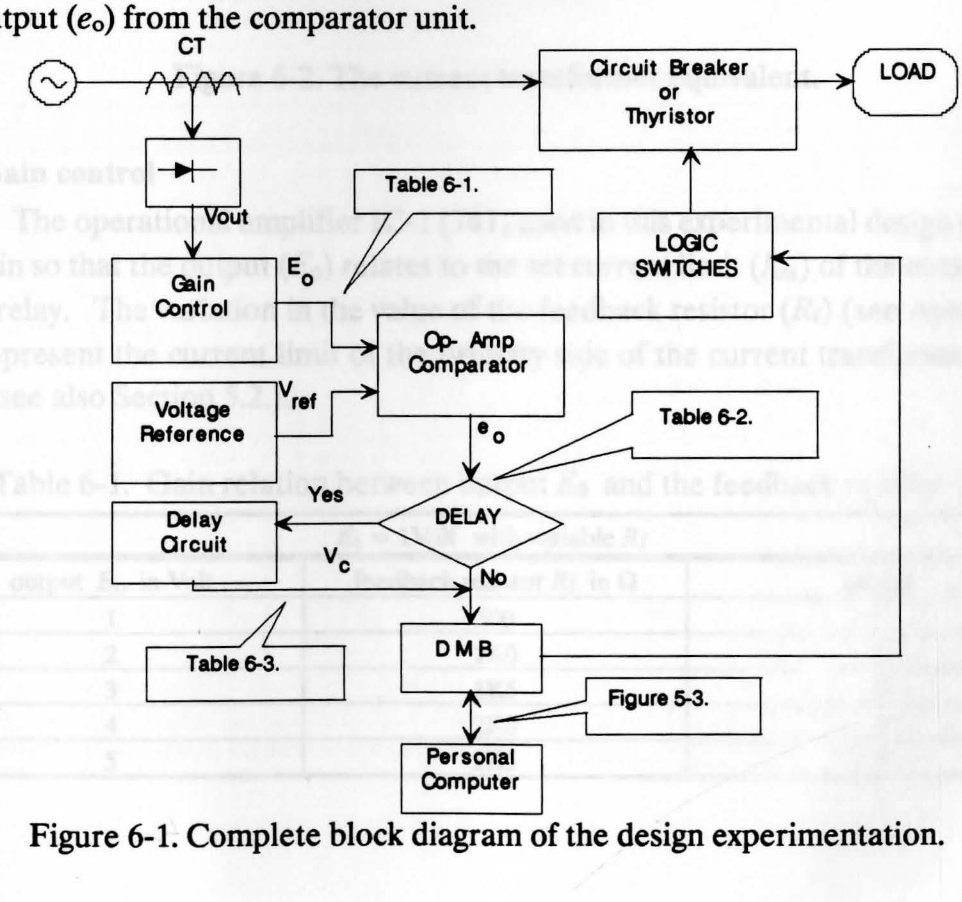


Figure 6-1. Complete block diagram of the design experimentation.

6.1. Current reference

The current transformer in this experimental design (Figure 6-2.) will transform the actual primary current to a voltage across the secondary (V_{CT}) (see Section 4.3.). The experimental design of the over-current static relay provides the equivalent of an actual current transformer as a regulated power supply that impresses a voltage across the series of resistors R_1 and R_2 (Section 4.5.3.).

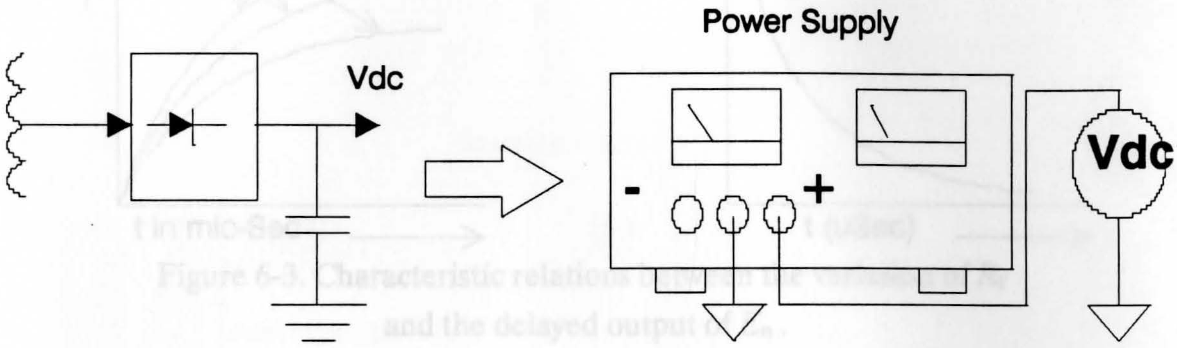


Figure 6-2. The current transformer equivalent.

6.2. Gain control

The operational amplifier IC-1 (741) used in this experimental design provides the gain so that the output (E_o) relates to the set current limit (I_{set}) of the over-current static relay. The variation in the value of the feedback resistor (R_f) (see Appendix 3) will represent the current limit of the primary side of the current transformer (Table 4-1.) (see also Section 5.2.).

Table 6-1. Gain relation between output E_0 and the feedback resistor R_f.

E_i = 1Volt with variable R_f		
output E_o in Volt	feedback resistor R_f in Ω	gain β
1	500	1
2	1K0	2
3	1K5	3
4	2K0	4
5	2K5	5

The output characteristics of the gain control circuit, which were obtained using the Micro Cap advanced circuit analysis digital computer program by SPECTRUM, are shown in Figure 6-3.

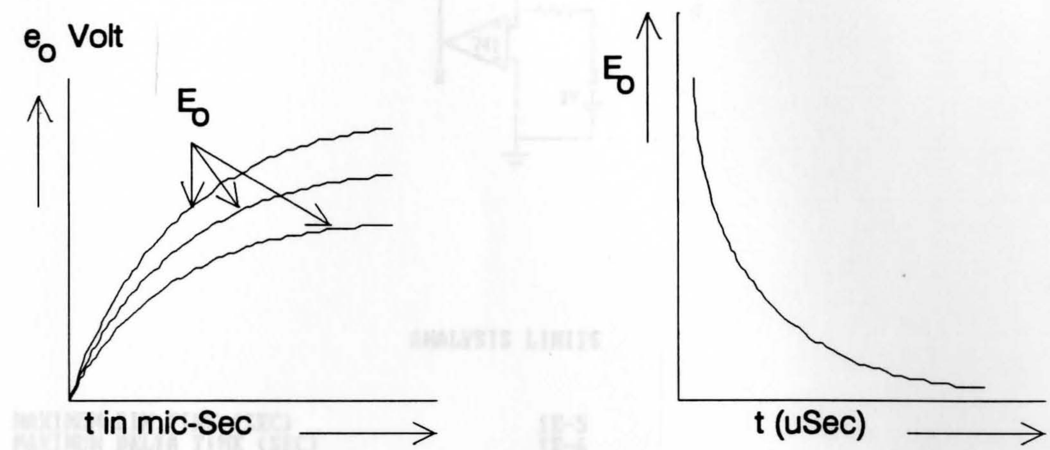
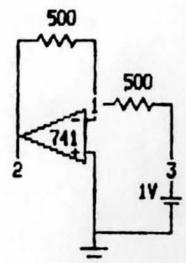


Figure 6-3. Characteristic relations between the variation of R_f and the delayed output of E_o.

Table 6-2. Delayed output E_o with variable of β.

Delayed time output of (E_o) in μSecond					Voltage reference (V_{ref}) = 1 Volt and E_o in Volt				
$\beta=1$	$\beta=2$	$\beta=3$	$\beta=4$	$\beta=5$	$E_o=1$	$E_o=2$	$E_o=3$	$E_o=4$	$E_o=5$
0	0	0	0	0	0.067	0.045	0.033	0.026	0.021
0.01	0.01	0.01	0.01	0.01	0.066	0.043	0.031	0.023	0.018
0.11	0.11	0.11	0.11	0.11	0.036	0.009	-0.005	-0.014	-0.02
0.506	0.463	0.443	0.432	0.425	-0.13	-0.144	-0.153	-0.158	-0.162
0.793	0.740	0.715	0.7	0.691	-0.259	-0.272	-0.28	-0.285	-0.288
1.061	1	0.972	0.955	0.944	-0.382	-0.394	-0.402	-0.405	-0.41
1.323	1.256	1.224	1.205	1.193	-0.503	-0.515	-0.523	-0.528	-0.531
1.583	1.509	1.474	1.454	1.44	-0.623	-0.635	-0.643	-0.649	-0.652
1.583	1.762	1.724	1.702	1.687	-0.743	-0.756	-0.763	-0.768	-0.773
1.841	2.014	1.973	1.949	1.933	-0.863	-0.876	-0.883	-0.887	-0.892
2.1	2.267	2.223	2.197	2.18	-0.938	-0.996	-1.003	-1.009	-1.013
2.358	2.519	2.472	2.444	2.426	-0.985	-1.116	-1.123	-1.128	-1.132
2.773	2.771	2.721	2.692	2.672	-1	-1.236	-1.363	-1.368	-1.252
3.773	3.024	2.970	2.939	2.918	-1	-1.356	-1.483	-1.488	-1.492
4.773	3.276	3.220	3.187	3.165	-1	-1.476	-1.603	-1.608	-1.612

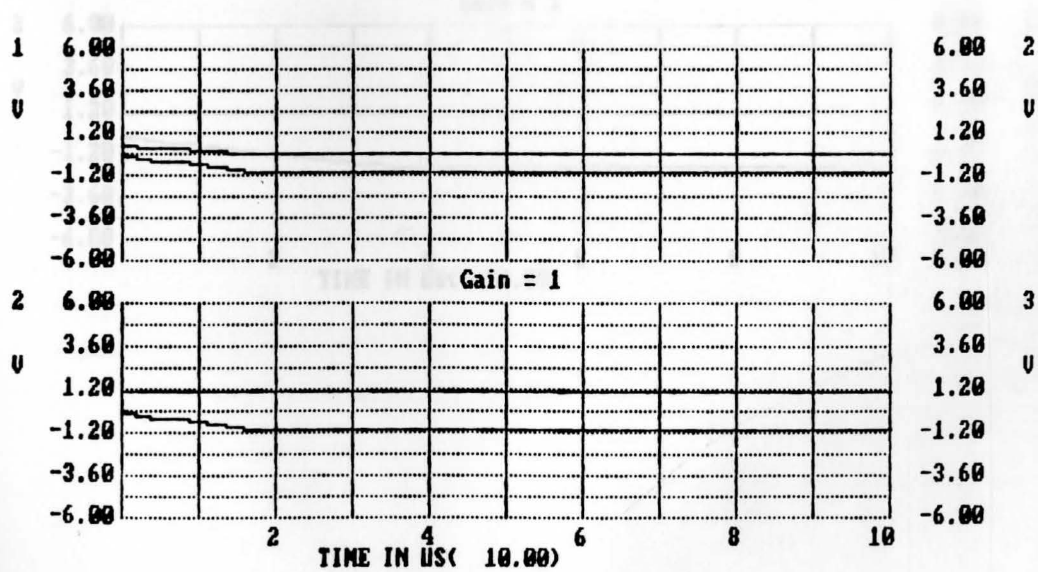
The real-time simulation of delayed output E_o is shown on pages 34 through 38.

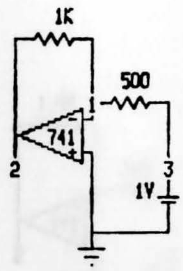


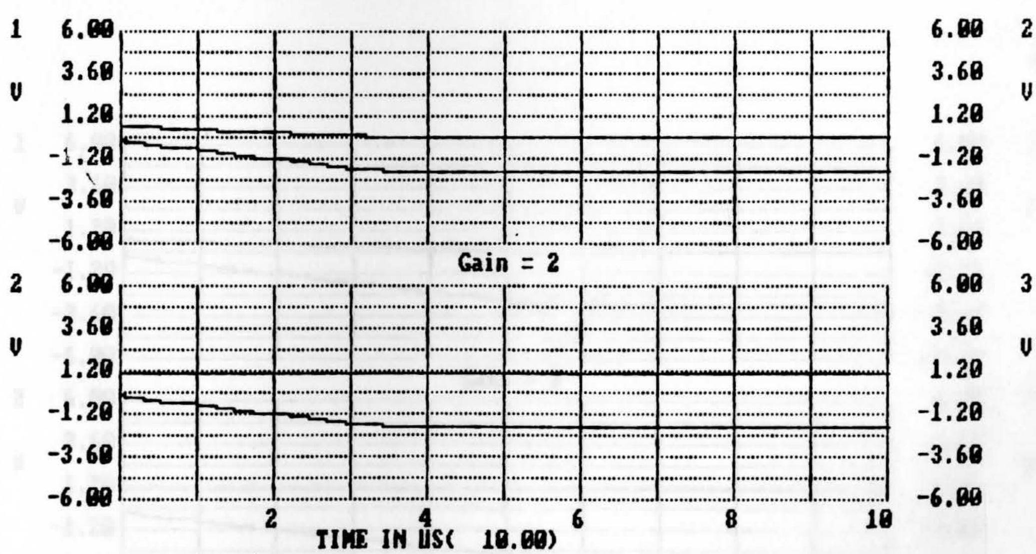
ANALYSIS LIMITS

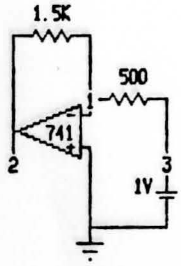
MAXIMUM SIM TIME (SEC)	1E-5
MAXIMUM DELTA TIME (SEC)	1E-6
MINIMUM ACCURACY (%)	1
UPPER TRACE A	1
UPPER TRACE B	2
UPPER TRACE RANGE(HIGH/LOW)	6/-6
LOWER TRACE A	2
LOWER TRACE B	3
LOWER TRACE RANGE(HIGH/LOW)	6/-6
ZERO INITIAL CONDITIONS (Y/N)	Y
EDIT/REVIEW INITIAL CONDITIONS (Y/N)	N
DUMP NODE WAVEFORM TO USER FILE (Y/N)	N
CRT PLOT(C) OR TABLE(T)	C
WORST CASE (Y/N)	N
TEMPERATURE (LOW/HIGH/STEP)	27
CALCULATE D.C. OPERATING POINT(Y/N)	N
SAVE(S) RETRIEVE(R) OR NORMAL(N) RUN	N

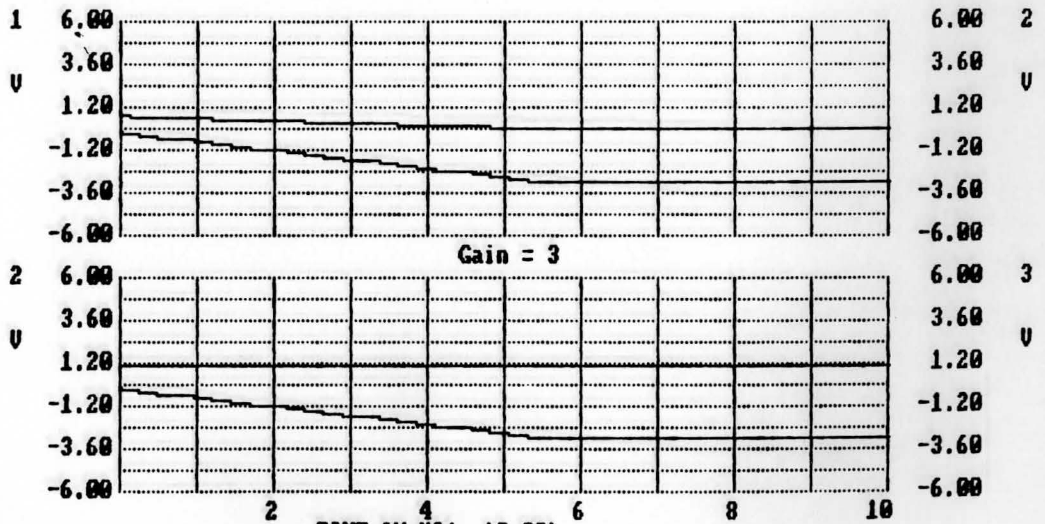
ARE THESE CORRECT (Y/N) ?

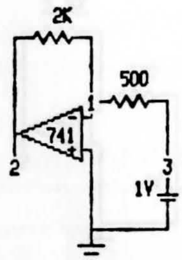


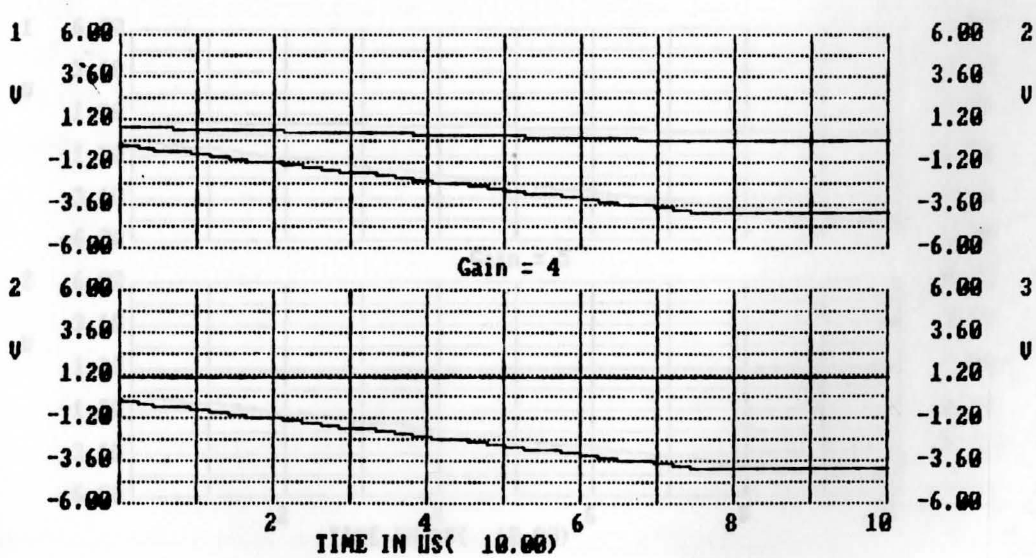

1K
500
741
1
2
3
1V

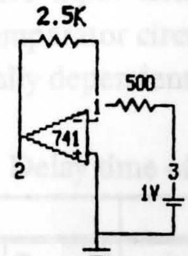

1
V
2
V
6.00
3.60
1.20
-1.20
-3.60
-6.00
Gain = 2
2
V
3
V
2
4
6
8
10
TIME IN US(10.00)

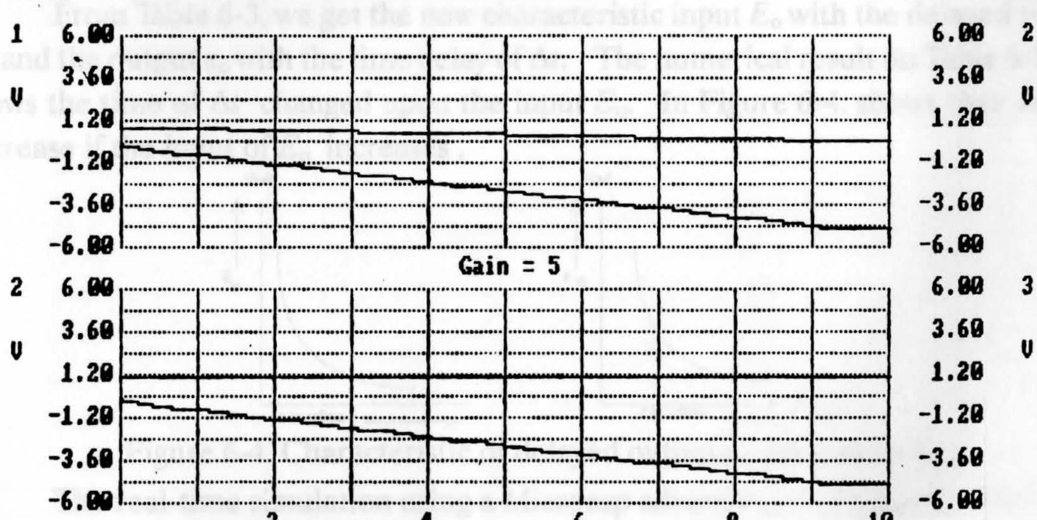

1.5K
500
741
1
2
3
1V


1
V
2
V
6.00
3.60
1.20
-1.20
-3.60
-6.00
3
Gain = 3
2
4
6
8
10
TIME IN US(10.00)


2K
500
1
741
2
3
1V


1 V
2 V
6.00
3.60
1.20
-1.20
-3.60
-6.00
Gain = 4
2 V
3 V
2
4
6
8
10
TIME IN US(10.00)


2.5K
500
1
741
2
3
1V


1
V
2
V
6.00
3.60
1.20
-1.20
-3.60
-6.00
Gain = 5
2
V
3
V
2
4
6
8
10
TIME IN US(10.00)

6.3. Comparator

Chapter 4.3. included the explanation of how to get the voltage output of e_o by comparing the voltage E_o (negative input terminal) and the voltage reference V_{ref} (positive input terminal) of the comparator circuit. Table 6-3. shows that the outputs e_o of the comparator are dynamically dependent upon the input from the gain control (E_o) with the delayed time of Δt.

Table 6-3. Delay time of the output e_o.

Level output of e_o in Volt					Time delay of Δt in μSecond				
$E_o = -1$	$E_o = -2$	$E_o = -3$	$E_o = -4$	$E_o = -5$	Δt_1	Δt_2	Δt_3	Δt_4	Δt_5
0.2887	0.433	0.5773	0.7216	0.866	0	0	0	0	0
0.4699	0.7039	0.9379	1.1719	1.406	0.01	0.01	0.01	0.01	0.01
0.6615	0.9403	1.2083	1.487	1.7232	0.038	0.024	0.022	0.021	0.019
0.7814	1.0989	1.4076	1.7034	1.988	0.11	0.067	0.05	0.04	0.034
0.9489	1.2105	1.5324	1.8602	2.1797	0.41	0.191	0.114	0.08	0.061
1.3972	1.4921	1.6826	1.9722	2.2996	1.306	0.742	0.372	0.207	0.133
1.8961	1.981	2.1120	2.2613	2.4669	2.305	1.721	1.229	0.774	0.433
2.3952	2.48	2.6105	2.7512	2.915	3.305	2.721	2.228	1.755	1.329
2.8941	2.979	3.1094	3.2501	3.4139	4.305	3.721	3.228	2.755	2.329
3.393	3.4778	3.6082	3.7489	3.9127	5.305	4.721	4.228	3.755	3.329
3.8918	3.9766	4.107	4.2476	4.4114	6.305	5.721	5.228	4.755	4.329
4.3904	4.4752	4.6056	4.7463	4.91	7.305	6.721	6.228	5.755	5.329
4.889	4.973	5.1042	5.2448	5.4085	8.305	7.721	7.228	6.755	6.329
5.3875	5.4723	5.6026	5.7432	5.9069	9.305	8.721	8.228	7.755	7.329

From Table 6-3, we get the new characteristic input E_o with the delayed time of Δt and the output e_o with the time delay of Δt. The numerical result on Table 6-3. also shows the time of Δt changed upon the input E_o. In Figure 6-4. shows that Δt will decrease if the input of E_o increases .

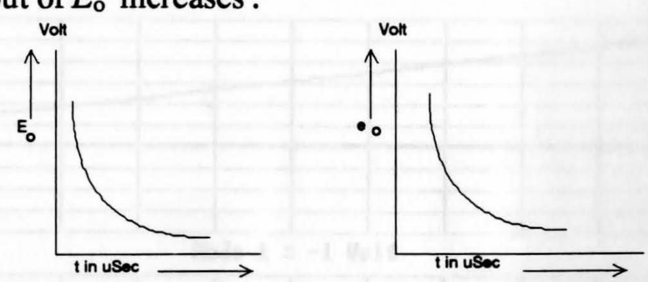
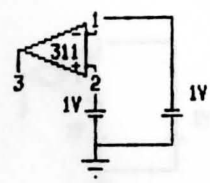


Figure 6-4. Characteristic of delayed output e_o proved by E_o.

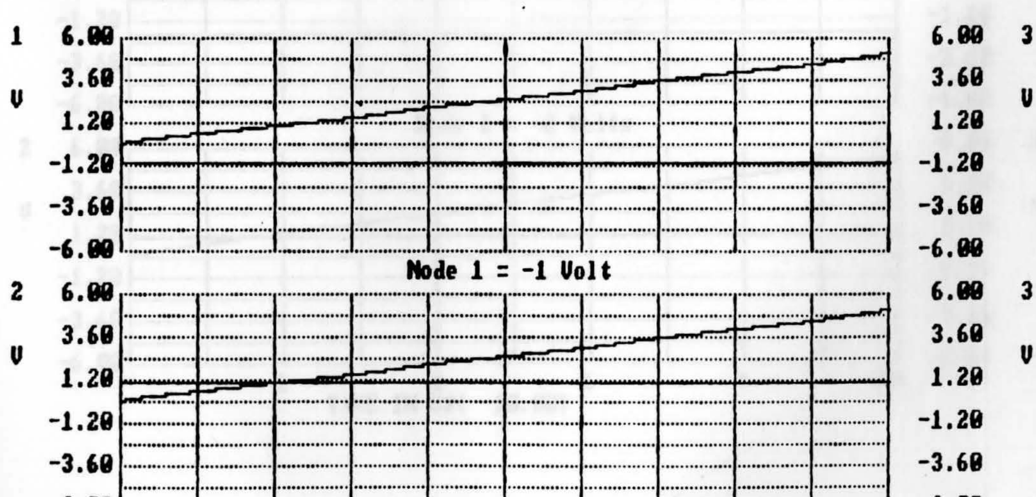
The real-time simulation using a Microcap advance circuit analysis program of this comparator circuit is shown on pages 40 through page 44.

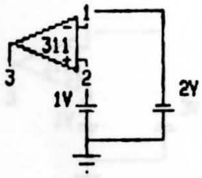


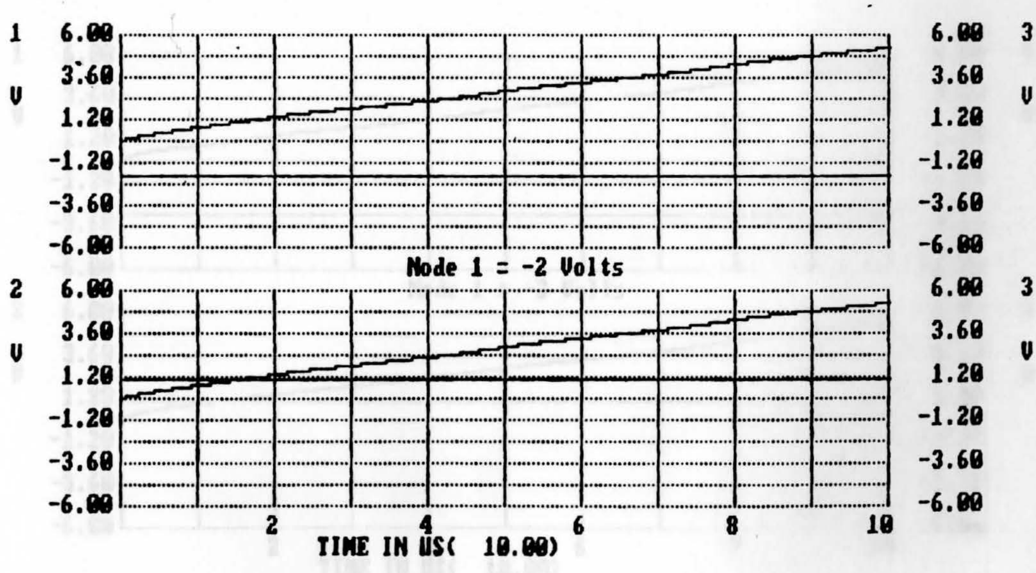
ANALYSIS LIMITS

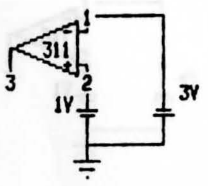
MAXIMUM SIM TIME (SEC)	1E-5
MAXIMUM DELTA TIME (SEC)	1E-6
MINIMUM ACCURACY (%)	5
UPPER TRACE A	1
UPPER TRACE B	3
UPPER TRACE RANGE(HIGH/LOW)	6/-6
LOWER TRACE A	2
LOWER TRACE B	3
LOWER TRACE RANGE(HIGH/LOW)	6/-6
ZERO INITIAL CONDITIONS (Y/N)	Y
EDIT/REVIEW INITIAL CONDITIONS (Y/N)	N
DUMP NODE WAVEFORM TO USER FILE (Y/N)	N
CRT PLOT(C) OR TABLE(T)	C
WORST CASE (Y/N)	N
TEMPERATURE (LOW/HIGH/STEP)	27
CALCULATE D.C. OPERATING POINT(Y/N)	N
SAVE(S) RETRIEVE(R) OR NORMAL(N) RUN	N

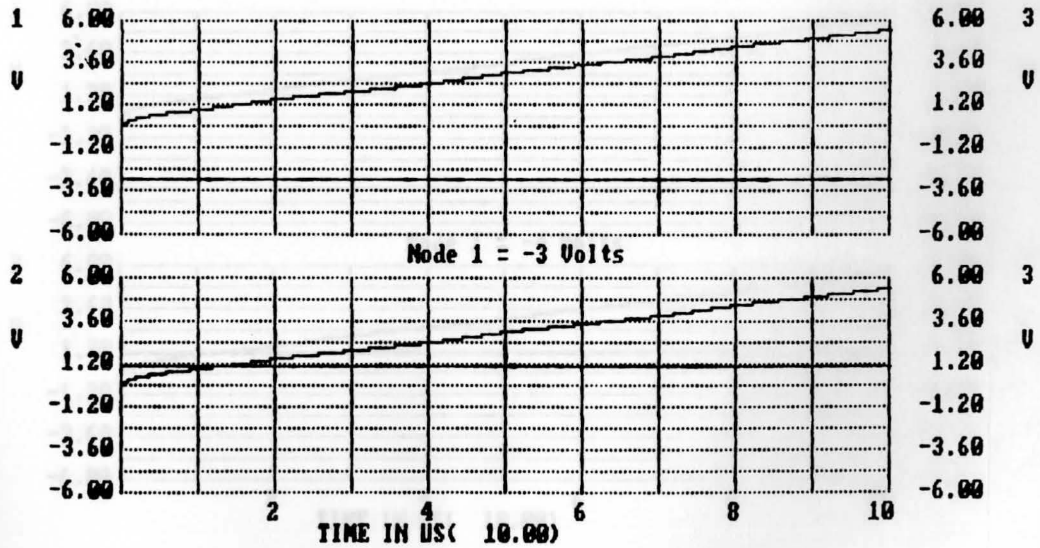
ARE THESE CORRECT (Y/N) ?

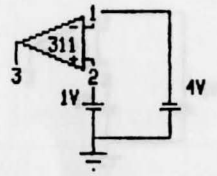


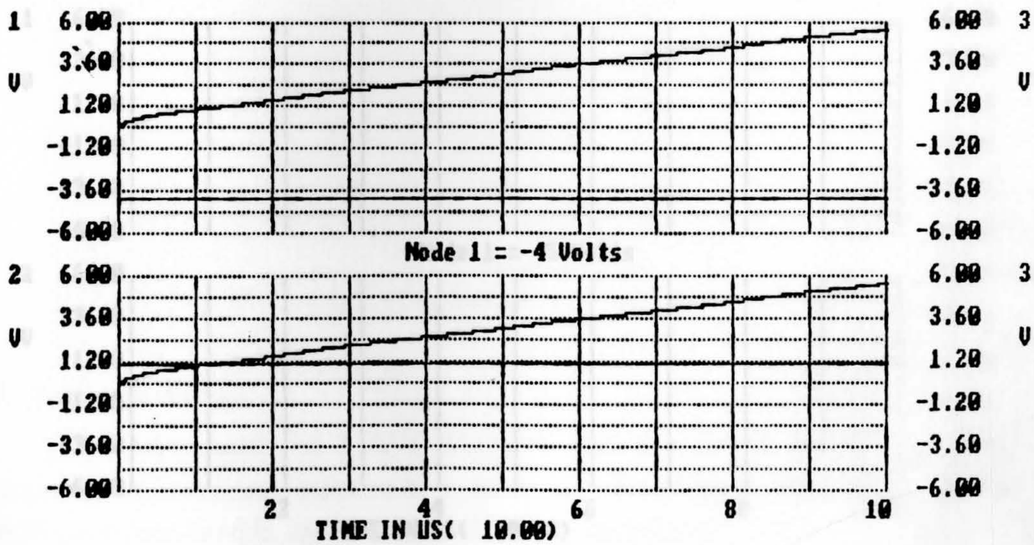

1
311
3
2
1V
2V

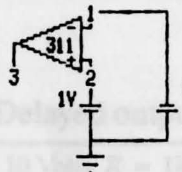

1
V
2
V
3
V
3
V
6.00
3.60
1.20
-1.20
-3.60
-6.00
Node 1 = -2 Volts
2
4
6
8
10
TIME IN US(10.00)

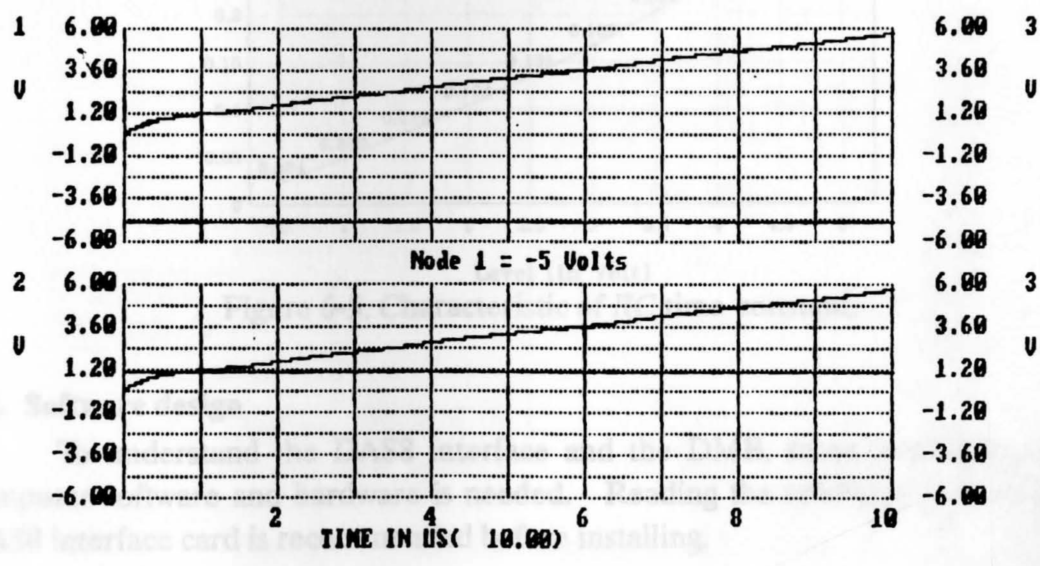

1
311
3
2
1V
3V


1
6.00
3.60
V
1.20
-1.20
-3.60
-6.00
3
Node 1 = -3 Volts
2
6.00
3.60
V
1.20
-1.20
-3.60
-6.00
3
2
4
6
8
10
TIME IN US(10.00)


1
311
3
2
1V
4V


1
V
2
V
3
V
3
V
6.00
3.60
1.20
-1.20
-3.60
-6.00
Node 1 = -4 Volts
2
4
6
8
10
TIME IN US(10.00)


1
311
3
2
1V
5V


1
V
6.00
3.60
1.20
-1.20
-3.60
-6.00
3
V
Node 1 = -5 Volts
2
V
3
2
4
6
8
10
TIME IN US(10.00)

6.4. Delay circuit

In Section 4.6., the delay circuit was discussed. This unit is used when the system protection needs to delay the operation of the circuit breaker. The purpose of this circuit is to prevent the action of the circuit breaker during inrush effect (see Section 4.4.). Table 6-4. shows the actual delay of the voltage across the capacitor (see also Appendix 2) V_c .

Table 6-4. Delayed output voltage of V_c

Tested on input voltage V_i = 10 Volts, R = 1K3 Ω and C = 470 micro-Farads	
output voltage V_c	delayed output in Second
1	0.049
2	0.104
3	0.167
4	0.240
5	0.325

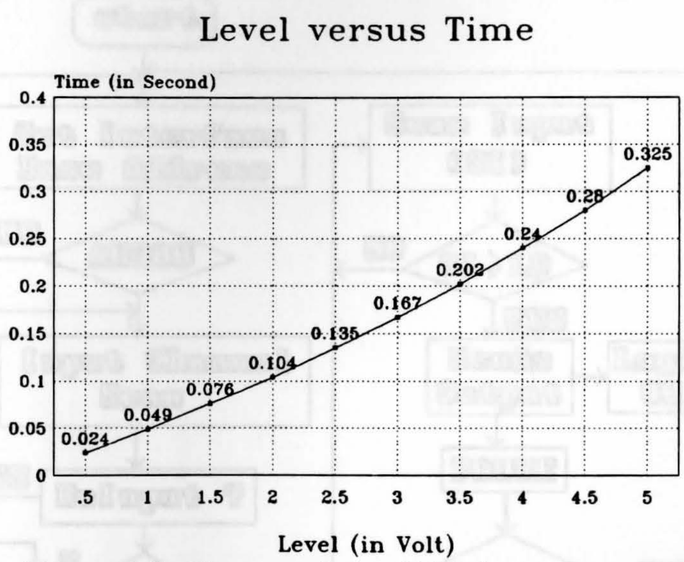


Figure 6-5. Characteristic of RC time-constant.

6.5. Software design

To understand the DAS8 interface and the DMB, some knowledge of the computer software and hardware is needed. Reading the reference manual of the DAS8 interface card is recommended before installing.

In this design, the DAS8 interface card is set to the base address &H300 hexadecimal or 768 decimal. The setup of this address is very important because if there are two card interfaces using the same address location, the computer will freeze (an internal conflict exists between those cards).

Choosing the base address of DAS8 interface bus, the user should at least be knowledgeable about the stackable memory address within the personal computer. To find out about the stackable address memory of the personal computer, the IBM Technical Reference Manual is recommended. For this purpose, the interface card uses the starting address &HC000 hexadecimal right below the address of the current fixed disk (&HC800) hexadecimal. Usually, this address location is empty for the user to place the address.

The computer languages that were used in this design were BASIC and Assembly, which is known as the Machine Language (Appendix 5). Communication between the operator and the computer is called "CLI" (Command Language Interface).

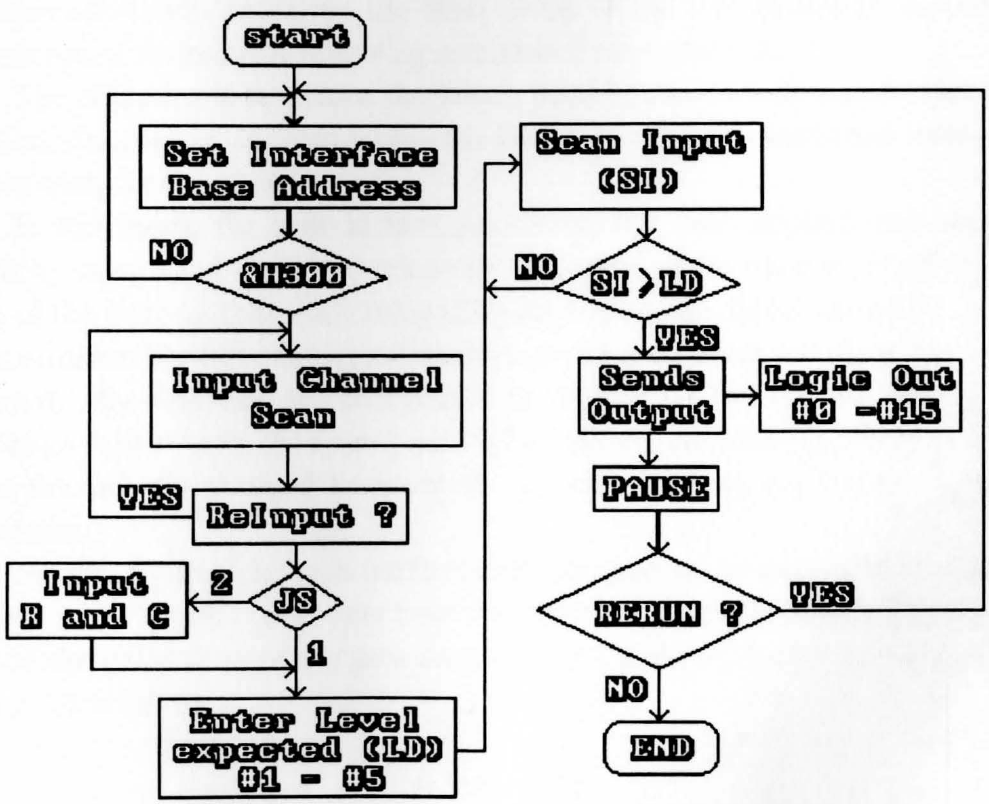


Figure 6-6. Final flow chart of the CLI design.

The DAS8 interface does not limit the user to BASIC. The flexibility of this interface card will allow the user to utilize a different CLI such as Assembly, Fortran, Pascal and C. The driver program should be loaded while the main program is executed. To make the card work properly, the driver program must be loaded into the memory of the computer and then called by the application program.

The CLI in this system design is shown in Figure 6-6. The CLI program is written in BASIC, and the driver program is written in Assembly. The BASIC program will be compiled using the Professional BASIC compiler to produce the object file with the extension OBJ's. After compiling the program, it is then linked with the driver program to produce a stand-alone, executable application program.

The purpose of using this method is speed, because a compiled program usually has a faster running time than a program running under CLI.

7. CONCLUSION

Summary

The main contribution of this thesis is the development of an over-current static relay in order to improve the accuracy and the relative speed operation of the relay. The experimentation to define the time delay of the system design supports the performance of the relay by following established relay policies.

The objective is to reduce the power used by the current transformer during operation, if necessary, and also reduce the size of the current transformer from regular size into compact size (head room).

In this thesis, the over-current static relay has been applied into protection systems by using fixed digital components or a stand-alone fixed static relay. The results of the over-current static relay using an analog and digital computer interface compared favorably with existing static relays when the different sets of the initial values were used. By observing the characteristics (Figure 7-1.) of the over-current static relay design in this thesis, the appropriate values of the time delay operation of the relay can be dynamically changed to avoid the primary jumping current of the current transformer.

Finally, Figure 7-1. shows the final characteristics of the over-current static relay designed in this thesis. To get the final characteristic time delay of the Pre-Interface, combine the delay time of the gain control t_{GC} (Table 6-2.) and the comparator t_C (Table 6-3.) which is:

$$t_1 = t_{GC} + t_C$$

Where t_1 = Summation of delayed output time of gain control and comparator.

t_{GC} = Delayed output time (E_o) of the Gain Control.

t_C = Delayed output time (e_o) of the Comparator.

The *RC* time-constant (t_2) is used in this design only if needed. Therefore, time delay for all systems is :

$$t = t_1 + t_2$$

And, finally

$$T_{op} = t + t_{CB} \; ..... \text{(see Section 2.2.)}$$

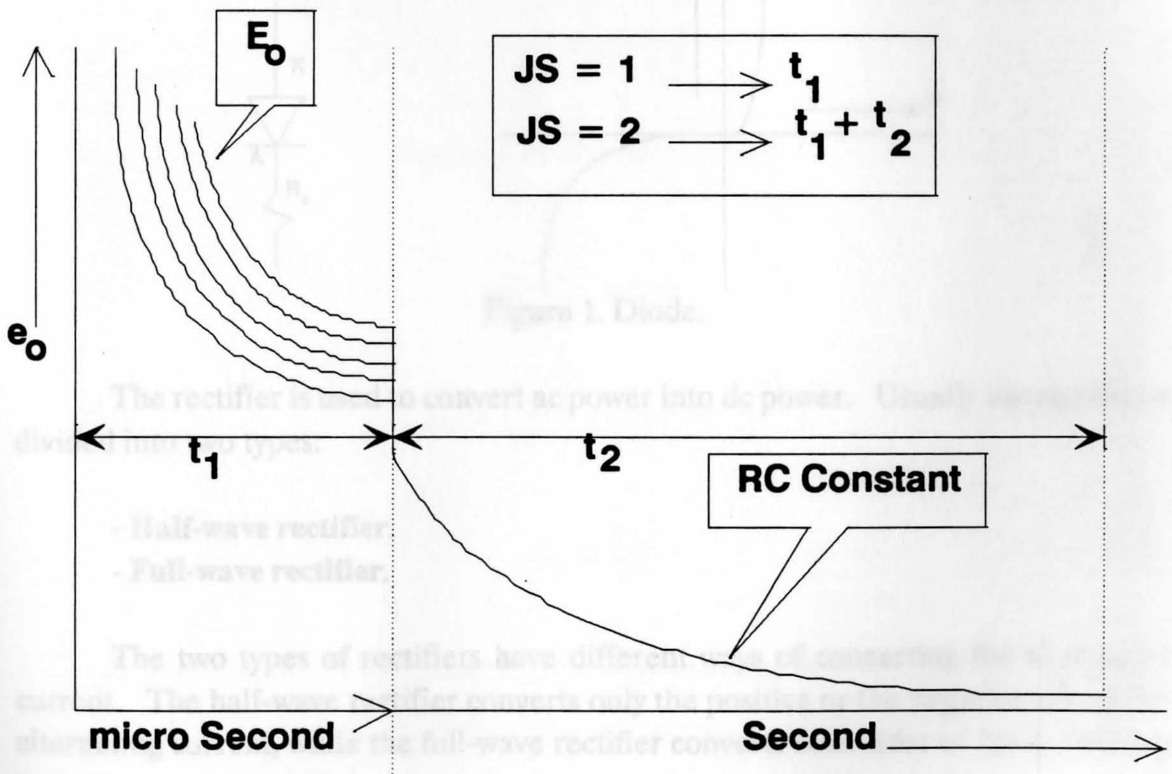


Figure 7-1. Characteristic of the over-current static relay design.

APPENDIX 1

RECTIFIER

The conversion process from alternating current (ac) to direct current (dc) is called rectification. In converting the alternating current into a direct current use a component called a "Diode." The diode is a semiconductor device that will cut the negative or the positive area of the alternating signal (Figure 1.).

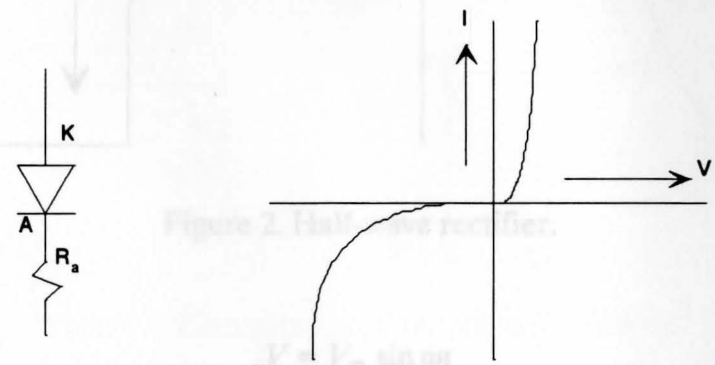


Figure 1. Diode.

The rectifier is used to convert ac power into dc power. Usually the rectifier is divided into two types:

- **Half-wave rectifier.**
- **Full-wave rectifier.**

The two types of rectifiers have different ways of converting the alternating current. The half-wave rectifier converts only the positive or the negative side of the alternating current, while the full-wave rectifier converts both sides of the alternating current.

Half-wave rectifier

In this type of rectifier, the actual diode is represented by an ideal diode with a forward resistance. The purpose of the rectification is to obtain a unidirectional current where the dc component is the average value. Figure 2. shows the conversion of the ac component into the dc component.

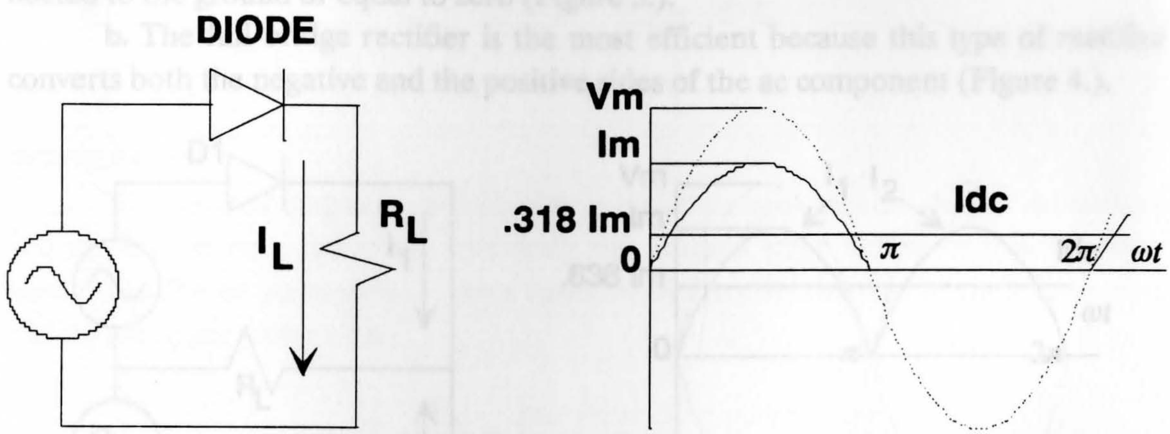


Figure 2. Half-wave rectifier.

$$V = V_m \sin \omega t$$

$$I_{dc} = \frac{1}{2\pi}\int_0^{2\pi} i\, d(\omega t) = \frac{I_m}{\pi}$$

$$V_{dc} = \frac{V_m}{\pi}$$

In practice, the dc component of the half wave rectifier is not exactly equal to $\frac{V_m \text{ or } I_m}{\pi}$. It is approximately 30% of the maximum input voltage.

Full-wave rectifier

The full-wave (bridge) rectifier provides a greater dc value than the half wave rectifier from the same voltage. Actually, the dc component of the bridge rectifier is twice as large as the half wave rectifier. The bridge rectifier is divided into two types:

a. The center-tap (phase inverter) rectifier converts only one side of the ac component, either the negative or the positive. The remaining pole is always connected to the ground or equal to zero (Figure 3.).

b. The full-bridge rectifier is the most efficient because this type of rectifier converts both the negative and the positive sides of the ac component (Figure 4.).

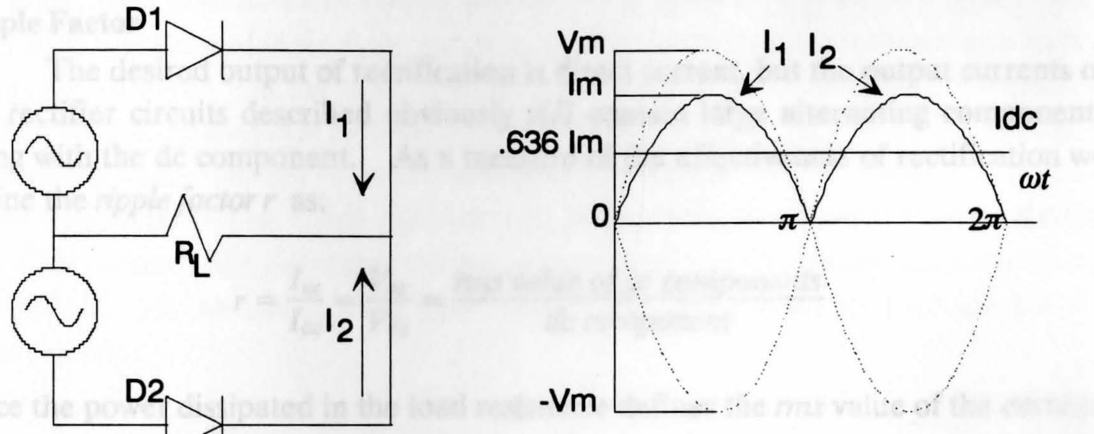


Figure 3. Center-tap rectifier (phase inverter).

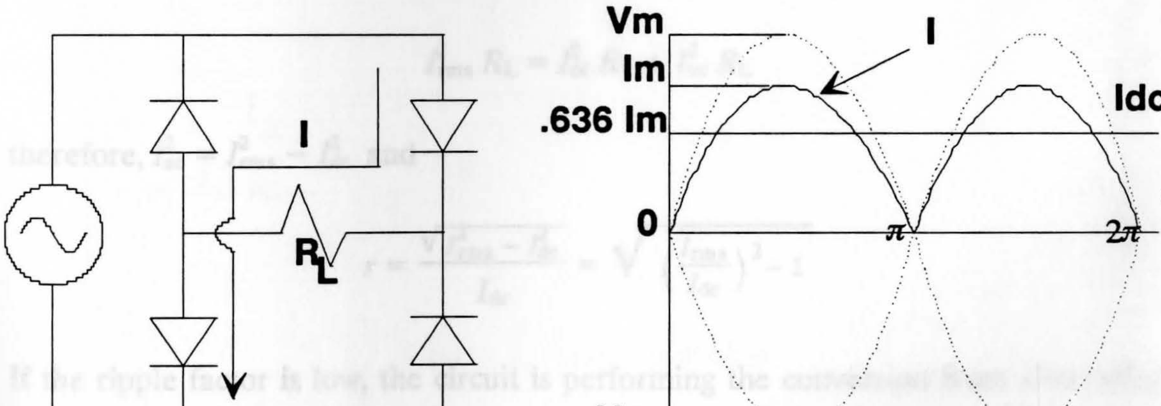


Figure 4. Full-bridge rectifier.

$$I_{dc} = \frac{2\,I_m}{\pi}$$

$$V_{dc} = \frac{2\,V_m}{\pi}$$

where

$$I_m = \sqrt{2}\cdot V_{ac} \qquad V_m = \sqrt{2}\cdot I_{ac}$$

Ripple Factor

The desired output of rectification is direct current, but the output currents of the rectifier circuits described obviously still contain large alternating components along with the dc component. As a measure of the effectiveness of rectification we define the *ripple factor r* as:

$$r = \frac{I_{ac}}{I_{dc}} = \frac{V_{ac}}{V_{dc}} = \frac{\textit{rms value of ac components}}{\textit{dc component}}$$

Since the power dissipated in the load resistance defines the *rms* value of the current, and since the total power is the summation of the power dissipated by the direct and alternating components,

$$I_{rms}^2 R_L = I_{dc}^2 R_L + I_{ac}^2 R_L$$

therefore, $I_{ac}^2 = I_{rms}^2 - I_{dc}^2$ and

$$r = \frac{\sqrt{I_{rms}^2 - I_{dc}^2}}{I_{dc}} = \sqrt{\left(\frac{I_{rms}}{I_{dc}}\right)^2 - 1}$$

If the ripple factor is low, the circuit is performing the conversion from alternating current to direct current effectively (Millman & Halkias, Electronic Devices and Circuits, McGraw-Hill Book Co., New York 1971).

FILTER

Although using full-wave instead of half-wave rectification reduces the ac component from 121% to 48% of the dc component, the output is still unsatisfactory for most electronic purposes.

Capacitor Filter

The ripple factor can be greatly decreased by using a *filter* consisting of a capacitor connected across the load resistor. The capacitor can be thought of as a low-impedance path taken by the ac components of the rectified wave. If the diode resistance is small and if the steady state has been reached, the operation is as shown in Fig. 5. At time $t = 0$, the source voltage is zero but the load voltage (v_L) = capacitor voltage (v_C) is appreciable because the previously charged capacitor is discharging through the load. At $t = t_1$, the increasing supply voltage v slightly exceeds v_L and the diode conducts. The diode current (i_D) rises abruptly to satisfy the relation $i_C = C\, dv / dt$ and then decreases to zero as expected from the natural response of the RC circuit. At the time t_3, the supply voltage again exceeds the load voltage and the cycle repeats.

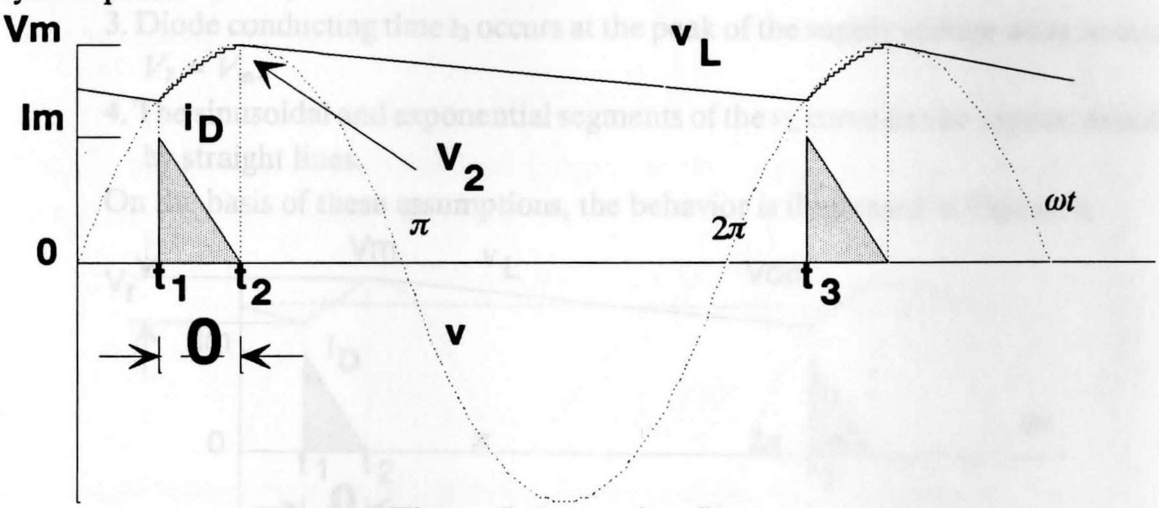


Figure. 5. A capacitor filter.

The diode conduction is off when v drops below v_L.

During the charging period $t_1 < t < t_2$,

$$v_L = V_m \sin \omega t$$

During the discharging period, $t_2 < < t < t_3$,

$$v_L = V_2 e^{-(t-t_2)/RC}$$

The load current is directly proportional to the load voltage. Because the load current will never go to zero, the average value or dc component is relatively large compared to the half-wave rectifier alone, and the ac component is correspondingly lower. The ripple factor is greatly reduced by the use of the capacitor.

Capacitor Filter - Approximate Analysis

Design charts are available in handbooks published by rectifier manufacturers (Zener Diode and Rectifier Handbooks, Motorola Inc., Phoenix, Arizona, 1987) which relate R_L, C, r, and V_L / V_m. The following approximate analysis, which gives satisfactory results for most purposes, illustrates the roles played by the various circuit parameters. Assume that:

1. The time constant $R_L C$ is large enough that the charging interval between t_1 and t_2 is small compared to the period time T for one cycle.
2. The diode current i_D, a portion of a cosine wave, can be approximated by a triangular pulse.
3. Diode conducting time t_2 occurs at the peak of the supply voltage wave so that $V_2 = V_m$.
4. The sinusoidal and exponential segments of the v_L curve can be approximated by straight lines.

On the basis of these assumptions, the behavior is illustrated in Figure. 6.

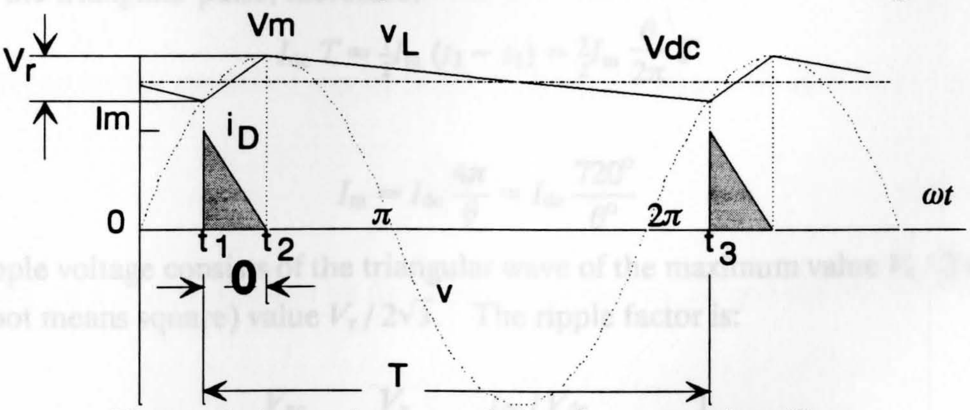


Figure. 6. Approximate analysis using a capacitor filter.

For a given Load and Capacitor, designers are interested in determining the necessary supply voltage, the diode rating, and the resulting voltage variation. If the charging interval is negligibly small, the load current is supplied by the capacitor and the charge transferred is:

$$Q = I_{dc}\, T \approx C \Delta V_C = C V_r$$

Solving, the ripple voltage is:

$$V_r \approx \frac{I_{dc} T}{C} = \frac{I_{dc}}{fC}$$

where f is the frequency, usually 60 Hertz. The necessary supply voltage is:

$$V_m = V_{dc} + \frac{V_r}{2} = V_{dc} + \frac{I_{dc}}{2fC}$$

The charging interval for the capacitor corresponds to the *conducting angle* for the diode which is:

$$\theta = \cos^{-1} \frac{V_m - V_r}{V_r}$$

and the conducting time is:

$$t_2 - t_1 = \frac{\theta}{2\pi} T$$

The charge carried through the load is equal to the charge conducted by the diode during the triangular pulse, therefore:

$$I_{dc}\, T \approx \tfrac{1}{2} I_m (t_2 - t_1) = \tfrac{1}{2} I_m \frac{\theta}{2\pi} T$$

or

$$I_m \approx I_{dc} \frac{4\pi}{\theta} = I_{dc} \frac{720^\circ}{\theta^\circ}$$

The ripple voltage consists of the triangular wave of the maximum value $V_r / 2$ and the *rms* (root means square) value $V_r / 2\sqrt{3}$. The ripple factor is:

$$r = \frac{V_{ac}}{V_{dc}} = \frac{V_r}{2\sqrt{3} fC} = \frac{I_{dc} / V_{dc}}{2\sqrt{3} fC} = \frac{1}{2\sqrt{3} fCR_L}$$

APPENDIX 2

RC time-constant

Time-varying voltages and currents

In a dc resistor circuit, the currents and voltages are constant. Even if switches are included, a switching operation causes voltage or current jumps from one constant level to another constant level. These are two values from which the voltages or currents change exponentially to their final values, but the jumps almost never reach to the final values. These voltages and currents vary with time or they are time varying. The quantities for the time-varying are distinguished from constant quantities and the numerical values of voltage and current are called "instantaneous values" because these values depend on (vary with) exact instants of time.

The specific time (T) here is not important because the charge in a resistive dc circuit flows at a steady rate. For time-varying, the value of current (i) usually changes using a very short time interval Δt. If Δq is the small charge that flows during the time interval, then the current is approximately $\Delta q / \Delta t$. For the exact current value, this quotient must be found in the limit as Δt approaches zero ($\Delta t \rightarrow 0$):

$$i = \lim_{\Delta t \rightarrow 0} \frac{\Delta q}{\Delta t} = \frac{dq}{dt}$$

The current for the capacitor can be found by substituting $q = Cv$ into $i = dq / dt$:

$$i = \frac{dq}{dt} = \frac{d}{dt}(Cv)$$

because C is a constant, it can be factored from a derivative and the result is:

$$i = C\,\frac{dv}{dt}$$

This equation specifies that the capacitor current at any time equals the product of the capacitance and the time rate of voltage change. But the current does *not* depend on the value of voltage at that time.

If a capacitor voltage is constant, then the voltage is not changing and so dv/dt is zero, making the capacitor current zero. Of course, from physical considerations, if a capacitor voltage is constant, no charge can enter or leave the capacitor, which means that the capacitor current is zero. With a voltage across it and zero current flow through it, the capacitor acts as an open circuit: *a capacitor is an open circuit to dc.* Remember, though, it is only after a capacitor voltage becomes constant that the capacitor acts as an open circuit. Capacitors are often used in electronic circuits to block dc currents and voltages.

Another important fact from $i = C\,dv\,/\,dt$ or $i \approx C\,\Delta v\,/\,\Delta t$ is that *a capacitor voltage cannot jump.* If, for example, a capacitor voltage could jump from 3V to 5V or, in other words, change by 2V in zero time, then Δv would be 2 and Δt would be 0, with the result that the capacitor current would be infinite. An infinite current is impossible because no source can deliver this current. Further, such a current flowing through a resistor would produce an infinite power loss, and there are no sources of infinite power and no resistors that can absorb such power.

Capacitor current has no similar restriction. It can jump or even change directions instantaneously. Capacitor voltage not jumping means that any capacitor voltage immediately following a switching operation is the same as it was immediately preceding the operation. This is an important fact for resistor-capacitor (*RC*) circuit analysis.

Single-capacitor dc-excited circuits

When switches open or close in a dc *RC* circuit with a single capacitor, all voltages and currents that change do so exponentially from the initial values to their final constant value, as can be shown from the circuit differential equation. The exponential terms in a voltage or current expression are called *transient terms* because they eventually become zero in practical circuits.

Figure 1. shows these exponential changes for a switching operation at $t = 0$ seconds. In Figure 1a, the initial value is greater than the final value, and in Figure

1b the final value is greater. Although both initial and final value are shown as positive, both can be negative or one can be positive and the other negative.

The voltages and currents approach their final values asymptotically, which means that they never actually reach them. As a practical matter, they are close enough to their final values to be considered to reach them.

Time-constant, with symbol τ, is the measure of the time required for certain changes in voltages and currents. For a single capacitor *RC* circuit, the time-constant of the circuit is the product of the capacitance and resistance:

$$RC \text{ time-constant} = \tau = RC$$

The expressions for the voltages and currents are:

$$V_{(t)} = v(\infty) + \left[v(0_+) - v(\infty) \right] e^{-t/\tau} \text{ Volt}$$

$$I_{(t)} = i(\infty) + \left[i(0_+) - i(\infty) \right] e^{-t/\tau} \text{ Ampere}$$

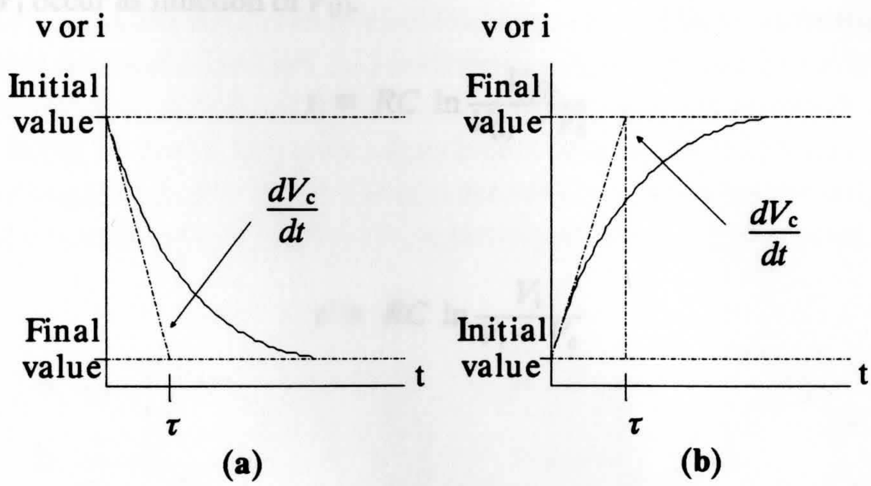


Figure 1. Characteristic of the *RC* time-constant circuit.

for all time greater than zero ($t > 0$). In these equations, $v(0+)$ and $i(0+)$ are the initial values immediately after switching; $v(\infty)$ and $i(\infty)$ are the final values; $e = 2.718$, the base of natural logarithms, and τ is the time-constant of the circuit.

To examine the variation of the voltage on the capacitor with the time-constant is

$$V_{(t)} = RI \left(1 - e^{-t - t_o / \tau} \right)$$

and if $t_o = 0$ then

$$V_{(t)} = RI \left(1 - e^{-t / \tau} \right)$$

therefore

$$\frac{dV_c}{dt} = \frac{V_{(t)}}{RC} \, e^{-t/RC}$$

By letting $t = \tau = RC$ the equation becomes:

$$V_c = V_{(t)} \left(1 - \frac{1}{e} \right) \approx 0.632 \, V_{(t)}$$

The above equation shows that the charge on the capacitor builds up the voltage approximately 63.2% of the source voltage (V_i) and it also defines the time t when the values of V_i occur as function of $V_{(t)}$.

$$t \equiv RC \ln \frac{V_{(t)}}{V_{(t)} - V_c}$$

or

$$t \equiv RC \ln \frac{V_i}{V_i - V_c}$$

APPENDIX 3

OPERATIONAL AMPLIFIER CIRCUITS

Introduction

Operational amplifiers, usually called *op-amps*, are important components of electronic circuits. Basically, an op-amp is a very high-gain voltage amplifier, having a voltage gain of 100,000 or more. Although an op-amp may consist of more than two dozen transistors, one dozen resistors, and perhaps one capacitor, it may be as small as an individual resistor. Because of its small size and relatively simple external operation, for purposes of analysis or design an op-amp can often be considered as a single circuit element.

Figure 1.a shows the circuit symbol for an op-amp. The three terminals are an *inverting input terminal a* (marked -), a *noninverting input terminal b* (marked +), and an output terminal c. A physical operational amplifier has more terminals. The extra two shown in Fig 1.b are for dc power supply inputs, which offer +15V and -15V. Both positive and negative power supply voltages are required to enable the output voltage on terminal *c* to vary both positively and negatively with respect to ground.

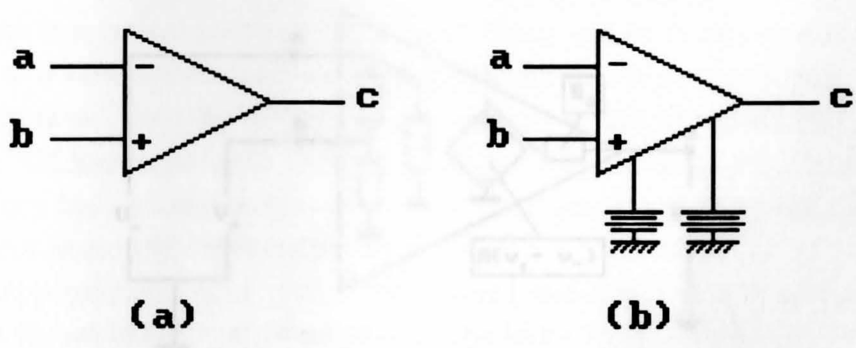


Fig 1. Basic op-amp.

Op-amp operation

The circuit of Fig 2a. is a model for an op-amp. It illustrates how an op-amp operates as a voltage amplifier. As indicated by the dependent voltage source, for an open-circuit load, the op-amp provides an output voltage of $v_o = A\ (v_+ - v_-)$, which is A times the difference in input voltages. This A is often referred to as the *open-loop voltage gain*. From $A\ (v_+ - v_-)$, observe that a positive voltage v_+ applied to the noninverting input terminal *b* tends to make the output voltage positive, and a positive voltage v_- applied to the inverting input terminal *a* tends to make the output voltage negative.

The open-loop voltage gain A is typically so large (100,000 or more) that it can often be approximated by infinity (∞), as is shown in the simpler model of Fig 2b. Note that Fig 2b. does not show the sources or circuits that provide the input voltage v_+ and v_- with respect to ground. Instead, just the voltages v_+ and v_- are shown. This simplifies the circuit diagrams without any loss of information.

In Fig 2a., the resistors shown at the input terminals have such large resistances (megohms) as compared to other resistances (usually kilohms) in a typical op-amp circuit, that they can be considered to be open circuits, as is shown in Fig 2b. As a consequence, the input currents to an op-amp are almost always negligibly small and assumed to be zero. This approximation is important to remember.

The output resistance R_a may be as large as 75Ω or more, and so may not be negligibly small. However, when an op-amp is used with negative-feedback components (as will be explained), the effect of R_a is negligible, and so R_a can be replaced by a short circuit, as shown in Fig 2b. Except for a few special op-amp circuits, negative feedback is always used.

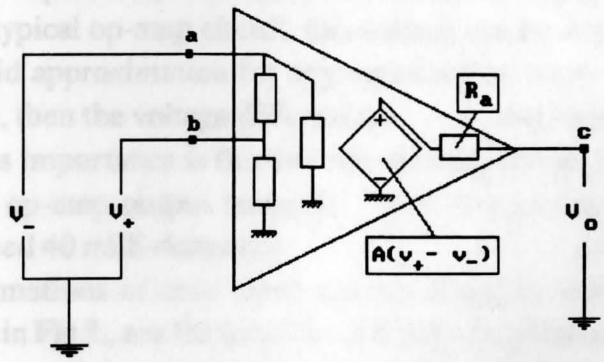


Fig 2a. Inside op-amp.

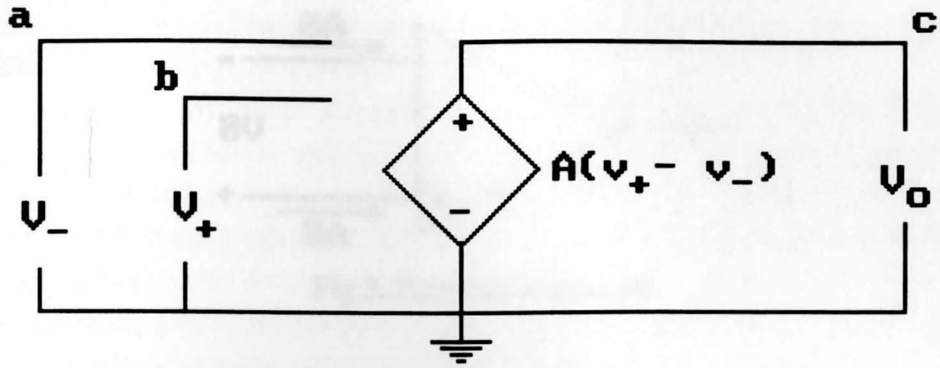


Fig 2b. Op-amp equivalent circuit.

The simple model of Fig 2b. is adequate for many practical applications. Although not indicated, there is a limit to the output voltage. It cannot be greater than the positive supply voltage or less than the negative supply voltage. In fact, it may be several volts less in magnitude than the magnitude of the supply voltages, with the exact magnitude depending upon the current drawn from the output terminal. When the output voltage is at either extreme, the op-amp is said to be *saturated*, or to be *in saturation*. An op-amp that is not saturated is said to be operating *linearly*.

Since the open-loop voltage gain A is so large and the output voltage is limited in magnitude, the voltage $v_+ - v_-$ across the input terminals must be very small in magnitude for an op-amp to operate linearly. Specifically, it must be less than 100 μV in a typical op-amp application. (This small voltage is obtained with *negative feedback*, as will be explained.) Because this voltage is negligible compared to the other voltages in a typical op-amp circuit, this voltage can be considered to be zero.

This is a valid approximation for any op-amp that is not saturated. But if an op-amp is saturated, then the voltage difference $v_+ - v_-$ can be significantly large, and typically is. Of less importance is the limit on the magnitude of the current that can be drawn from the op-amp output terminal. For one popular op-amp this output current cannot exceed 40 milli-Amperes.

The approximations of zero input current and zero voltage across the input terminals, as shown in Fig 3., are the bases for the following analysis of popular op-amp circuits. In addition, nodal analysis will be used almost exclusively.

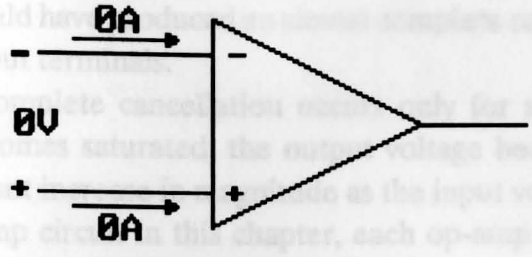


Fig 3. Zero input op-amp.

Popular op-amp circuits

Fig 4. shows the *inverting amplifier*, or simply the *inverter*. The input voltage is v_i and the output voltage is v_o. As will be shown, $v_o = \beta v_i$ in which β is a *negative* constant. The output voltage v_o is similar to the input voltage v_i, but it is amplified and inverted.

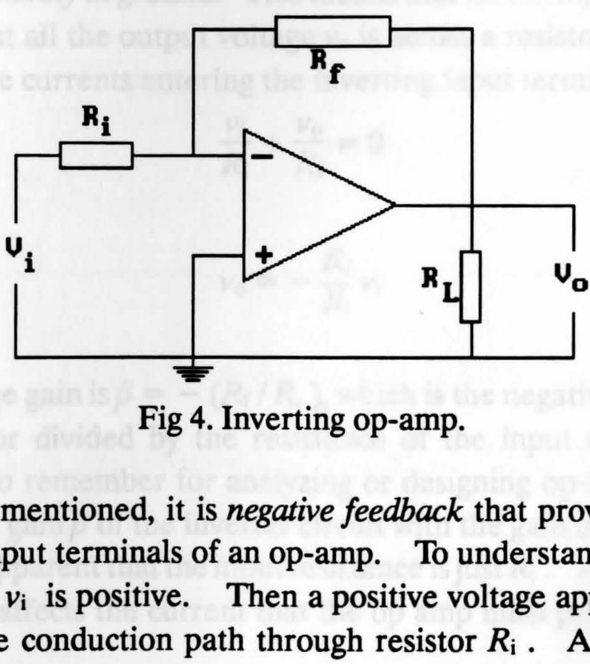


Fig 4. Inverting op-amp.

As has been mentioned, it is *negative feedback* that provides the almost zero voltage across the input terminals of an op-amp. To understand this, assume that in the circuit of Fig 4. v_i is positive. Then a positive voltage appears at the inverting input because of the conduction path through resistor R_i . As a result, the output voltage v_o becomes negative. Because of the conduction path through the resistor R_f, this negative voltage also affects the voltage at the inverting input terminal and causes an almost complete cancellation of the positive voltage there. If the input voltage v_i had been negative instead, then the voltage feedback would have been positive and again would have produced an almost complete cancellation of the voltage across the op-amp input terminals.

This almost complete cancellation occurs only for a nonsaturated op-amp. Once an op-amp becomes saturated, the output voltage becomes constant and the voltage feedback cannot increase in magnitude as the input voltage does.

In every op-amp circuit in this chapter, each op-amp has a feedback resistor connected between the output terminal and the inverting input terminal. Consequently, in the *absence of saturation*, all the op-amps in these circuits can be considered to have zero volts across the input terminals. They can also be considered to have zero currents into the input terminals because of the large input resistance.

The best way to obtain the voltage gain of the inverter of Fig 4. is to apply KCL (Kirchhoff Current Law) at the inverting input terminal. Before doing this, though, consider the following: Since the voltage across the op-amp input terminals is zero, and since the noninverting input terminal is grounded, it follows that the inverting input terminal is also effectively at ground. This means that all the input voltage v_i is across a resistor R_i, and that all the output voltage v_o is across a resistor R_f. Consequently, the summation of the currents entering the inverting input terminal is :

$$\frac{v_i}{R_i} + \frac{v_o}{R_f} = 0$$

and therefore

$$v_o = -\frac{R_f}{R_i} v_i$$

So, the voltage gain is $\beta = -(R_f / R_i)$, which is the negative of the resistance of the feedback resistor divided by the resistance of the input resistor. This is an important formula to remember for analyzing or designing op-amp inverter circuits. (Do not confuse this gain β of the inverter circuit with the gain of the op-amp itself.)

It should be apparent that the input resistance is just R_i. Additionally, although the load resistor R_L affects the current that the op amp must provide, it has no effect on the voltage gain.

The *summing amplifier*, or *summer*, is shown in Fig 5. Basically, a summer is an inverter circuit with more than one input. By convention, the sources for providing the input voltages v_a, v_b, and v_c are not shown. If this circuit is analyzed with the same approach as used for the inverter, the result is :

$$v_o = -\left(\frac{R_f}{R_a}v_a + \frac{R_f}{R_b}v_b + \frac{R_f}{R_c}v_c\right)$$

For the special case of all the resistances being the same, this formula simplifies to :

$$v_o = -(v_a + v_b + v_c)$$

There is no special significance to the inputs being three in number. There can be two, four, or more inputs.

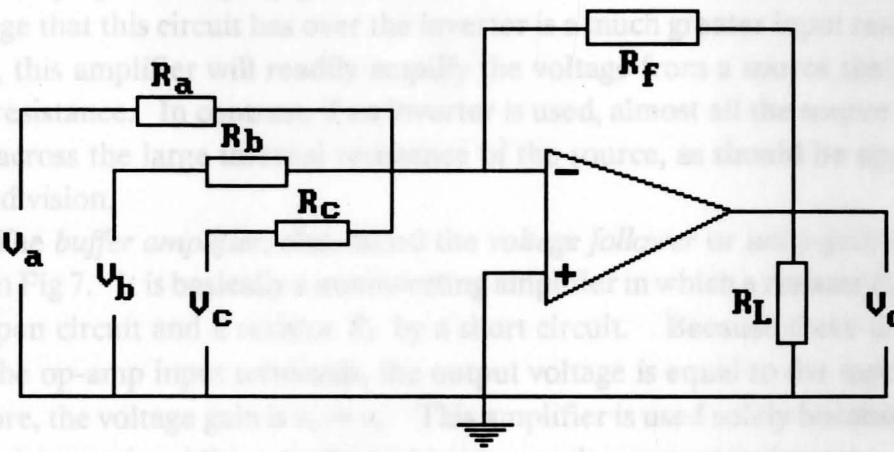


Fig 5. Summing inverter op-amp.

Fig 6. shows the ***noninverting voltage amplifier***. Observe that the input voltage v_i is applied at the noninverting input terminal. Because of the almost zero voltage across the input terminals, v_i is also effectively at the inverting input terminal. Consequently, the KCL equation at the inverting input terminal is :

$$\frac{v_i}{R_a} + \frac{v_i - v_o}{R_f} = 0 \text{ which results in } v_o = \left(1 + \frac{R_f}{R_a}\right) v_i$$

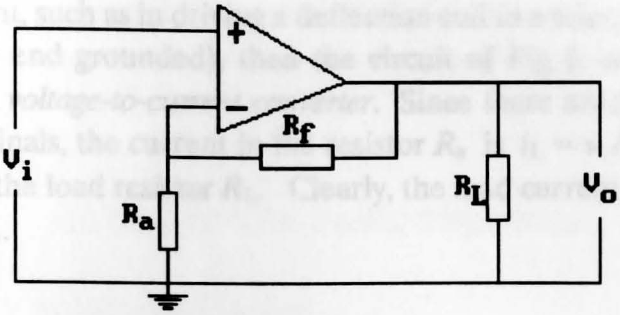


Fig 6. Non-inverting op-amp.

Since the voltage gain of $\beta = (1+R_f / R_a)$ does not have a negative sign, there is no inversion with this type of amplifier. Also, for the same resistances, the magnitude of the voltage gain is slightly greater than that of the inverter. The most significant advantage that this circuit has over the inverter is a much greater input resistance. As a result, this amplifier will readily amplify the voltage from a source that has a large output resistance. In contrast, if an inverter is used, almost all the source voltage will be lost across the large internal resistance of the source, as should be apparent from voltage division.

The *buffer amplifier*, also called the *voltage follower* or *unity-gain* amplifier, is shown in Fig 7. It is basically a noninverting amplifier in which a resistor R_a is replaced by an open circuit and a resistor R_f by a short circuit. Because there are zero volts across the op-amp input terminals, the output voltage is equal to the input voltage v_i. Therefore, the voltage gain is $v_o = v_i$. This amplifier is used solely because of its large input resistance, in addition to the typical op-amp low output resistance.

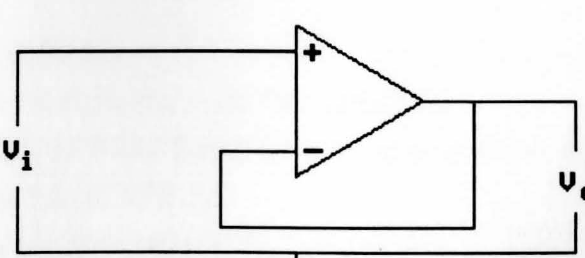


Fig 7. Unity gain op-amp.

There are applications in which a voltage signal is to be converted to a proportional output current, such as in driving a deflection coil in a television set. If the load is floating (neither end grounded), then the circuit of Fig 8. can be used. This is sometimes called a *voltage-to-current converter*. Since there are zero volts across the op-amp input terminals, the current in the resistor R_a is $i_L = v_i / R_a$, and this current also flows through the load resistor R_L. Clearly, the load current i_L is proportional to the signal voltage v_i.

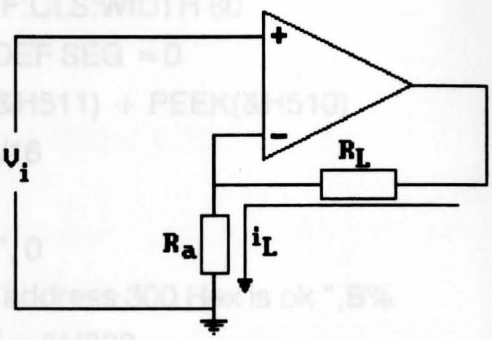


Fig 8. Voltage to current converter.

The circuit of Fig 8. can also be used for applications in which the load resistance R_L varies but the load current i_L must be constant. v_i is made a constant voltage and v_i and R_a are selected so that v_i / R_a is the desired current i_L. Consequently, when R_L varies, the load current i_L does not change. Of course, the load current cannot exceed the maximum allowable op-amp output current, and the load voltage plus the source voltage cannot exceed the maximum obtainable output voltage.

APPENDIX 4

LISTING OF THE BASIC PROGRAM

```
 10 SCREEN 0:KEY OFF:CLS:WIDTH 80
 20 CLEAR, 48*1024 : DEF SEG =0
 30 SG = 256 * PEEK(&H511) + PEEK(&H510)
 40 SG =SG + 49152!/16
 50 DEF SEG =SG
 60 BLOAD "DAS8.BIN", 0
 70 INPUT "Enter Base address 300 Hex is ok ",B%
 75 IF B%=0 THEN B%=&H300
 80 DIM DIO%(16)
 90 MD% = 0 : FLAG% = 0 : NCHAN%=8
 95 D%(0) = B%
100 CALL DAS8 (MD%, D%(0), FLAG%
105 IF FLAG%=3 OR FLAG%=10 THEN PRINT"DAS8 not installed, or I/O
    address out of range.":GOTO 495
110 IF FLAG%< >0 THEN PRINT"Error # ";FLAG%;" in DAS8 initialization."
    :GOTO 495
120 GOSUB 515
130 DIM CH%(16),YL%(16)
135 FOR I%=0 TO 16:YL%(I%)=-32768!:NEXT I%
140 CLS:LOCATE 25,1:PRINT"DAS8 Electronic strip chart Requires color
    graphics adapter";:LOCATE 1,1
145 PRINT"DAS8 is set for ";NCHAN%;" channels"
150 PRINT
```

```
155 INPUT "Which channels do you want plotted (0 - 7): ",X$
156 IF X$ = "" THEN GOTO 150
160 X$ = "-" + X$
165 L% = LEN(X$)
166 PRINT:INPUT"Enter desired grid calibration (1/10/20): ",A$
167 IF A$ = "1" THEN A = 1:GOTO 171
168 IF A$ = "10" THEN A = 10:GOTO 171
169 IF A$ = "20" THEN A = 20:GOTO 171
170 BEEP:SOUND 400,1:GOTO 166
171 FOR I% = 1 TO L%
175 IF MID$(X$,I%,1) = " " THEN MID$(X$,I%,1) = "-"
180 NEXT I%
185 FOR I% = 0 TO NCHAN%-1:CH%(I%) = 0:NEXT I%
190 CR% = ASC(LEFT$(X$,1))
195 IF ((CR% > = 48 AND CR% < = 55) AND (VAL(X$) < = NCHAN%-1
  AND VAL(X$) > = 0)) THEN CH%(VAL(X$)) = 1:L% = LEN(X$)
   :X$ = RIGHT$(X$,L%-(1 + INT(VAL(X$)/10)))
200 IF VAL(X$) > NCHAN%-1 THEN PRINT "One or more entries are incompatible
   with the configuration. Please re-enter":PRINT"Valid channel numbers range
   from 0 to ;NCHAN%-1:PRINT:GOTO 145
205 IF I% < NCHAN%-1 THEN N% = ASC(MID$(X$,I + 1,1))
210 L% = LEN(X$): IF L% > = 1 THEN X$ = RIGHT$(X$,L%-1):GOTO 190
215 IF U% > = 48 AND U% < = 55 AND N% > = 48 AND N% < 55 THEN
   CH%(10*(U%-48) + N%-48) = 1:I = I + 1
220 SCREEN 2: CLS
221 INPUT "Place your Level Detector ( 1 - 10 ): ";E%
222 IF E% < 1 OR E% > 10 THEN BEEP:CLS:GOTO 220
223 CLS:PRINT:PRINT"Do you want to interface as:"
224 PRINT:PRINT"   1. As an Instantaneous.":PRINT:PRINT"   2. As a time delay."
225 PRINT:INPUT "Choose one from above ( 1 or 2 ): ";Q%
```

```
226 IF Q%<1 OR Q%>2 THEN BEEP:BEEP:GOTO 223
227 CLS:ON Q% GOTO 2000,2050
228 CLS:ON E% GOSUB 1000,1010,1020,1030,1040,1050,1060,1070,1080,1090
230 LOCATE 25,1:PRINT"Press +/- to change speed, to pause - stop/start,
  to exit"
235 X%=30:U%=1:C%=1:LOCATE 23,15:PRINT"T(Delay) ="M" Sec."
  :LOCATE 23,49 :PRINT"Grid in 1 :":LOCATE 23,60:PRINT A:LOCATE 23,63
  :PRINT" second intervals";
240 DATA 5,4.5, 4,3.5, 3,2.5, 2,1.5, 1,0.5, 0
250 FOR I%=1 TO 11:READ A$:LOCATE I%*2-1,1:PRINT A$;:NEXT I%
255 VIEW SCREEN (30,0)-(610,170)
260 IF X%>= 610 THEN LINE (X%,0) - (X%,169),0:X%=30
  :LINE (X%-A,0)-(X%-A,169),0
265 LINE (X%+A,0) - (X%+A,169)
270 LINE (X%,0) - (X%,169),0
275 LINE (0,170)-(620,170)
285 MD%=1:D%(0)=0:D%(1)=NCHAN%-1
290 CALL DAS8 (MD%, D%(0), FLAG%)
295 MD%= 4
300 FOR Z%=0 TO NCHAN%-1
305 CALL DAS8 (MD%,D%(0),FLAG%)
310 IF VL=0 THEN DIO%(Z%)=D%(0)
315 NEXT Z%
320 FOR Z%=0 TO NCHAN%-1
325 IF CH%(Z%)=0 THEN GOTO 355
330 IF X%<30 THEN X%=30
335 Y%=DIO%(Z%)
340 IF YL%(Z%)=-32768! THEN GOTO 350
341 AA=164-YL%(Z%)*160!/2048!
342 BB=164-Y%*160!/2048!
```

```
345 LINE (X%,AA)-(X%,BB)
350 YL%(Z%) = Y%
355 NEXT Z%
360 GOSUB 450: IF Q + C% > T THEN GOTO 375
365 FOR I% = 1 TO 11 :PSET (X%,I%*16-12):NEXT I%
370 Q = T
375 X% = X% + A
380 FOR I% = 0 TO U%
385 A$ = INKEY$:IF A$ = "" GOTO 440
390 I% = U%
395 IF A$ = CHR$(27) THEN LOCATE 1,1:GOTO 685
400 IF A$ = " + " THEN U% = U%/2:IF U% = 1 THEN GOSUB 465
405 IF A$ = "-" AND U% < 16000 THEN U% = U%*2:IF U% > 16000
   THEN GOSUB 465
410 IF A$ = "-" AND U% > 16000 THEN GOSUB 465
415 IF U% < = 200 THEN C% = 1 :LOCATE 23,1:PRINT SPC(79):LOCATE 23,15
   :PRINT "T(Delay) = "M"Sec.":LOCATE 23,49:PRINT"Grid in 1 :"
   :LOCATE 23,60:PRINT A: LOCATE 23,63: PRINT" second intervals";
420 IF U% > 2000 THEN C% = 60: LOCATE 23,1:PRINT SPC(79)
   :LOCATE 23,15:PRINT"T (Delay) = "M" Sec.":LOCATE 23,51:PRINT"Grid
   in 1 :":LOCATE 23,62:PRINT A: LOCATE 23,65:PRINT" min intervals";
   :GOTO 430
425 IF U% > 200 THEN C% = 10: LOCATE 23,1:PRINT SPC(79):LOCATE 23,15
   :PRINT "T(Dealy) = "M" Sec.":LOCATE 23,51:PRINT"Grid in 10 :"
   :LOCATE 23,63 : PRINT A:LOCATE 23,66:PRINT" sec intervals";
430 IF A$ = " " THEN GOTO 435 ELSE GOTO 440
435 IF INKEY$ = "" GOTO 435
440 NEXT I%
445 GOTO 260
450 T$ = TIME$
```

```
455 T=3600!*VAL(LEFT$(T$,2))+60!*VAL(MID$(T$,4,2))+VAL(RIGHT$(T$,2))
460 RETURN
465 IF U%=1 THEN LOCATE 23,1:PRINT"MAX SPEED";
470 IF U%>10000 THEN LOCATE 23,1:PRINT"MIN SPEED";
475 SOUND 500,3:SOUND 400,3
480 LOCATE 23,1:PRINT"          ";
485 RETURN
495 PRINT:COLOR 0,7:PRINT" - Hit any key to return to Menu - ";:COLOR 7,0
500 IF INKEY$="" GOTO 500
505 CLS:GOTO 685
515 VL=0
520 MD%=20:CALL DAS8( MD%,D%(0),FLAG% ):IF D%(3)=-1 THEN RETURN
685 SCREEN 0:CLS:PRINT:PRINT"1. Do you want to rerun !"
690 PRINT:PRINT"2. Do you want to go back to Menu."
695 PRINT:INPUT"Enter number : ";Z%
700 IF Z%< OR Z%>2 THEN BEEP:BEEP:CLS:GOTO 685
710 ON Z% GOTO 3000,3010
800 MD%=14
805 OP%=0
810 FLAG%=0
815 CALL DAS8 (MD%,OP%,FLAG%)
820 IF FLAG%<>0 THEN CLS:PRINT"Error writing digital outputs !:GOTO 720
825 LOCATE 23,40:PRINT "TP=";OP%
830 RETURN
900 IF INKEY$="" THEN GOTO 900 ELSE RETURN
1000 LINE (30,156)-(610,156):RETURN
1010 LINE (30,140)-(610,140):RETURN
1020 LINE (30,124)-(610,124):RETURN
1030 LINE (30,108)-(610,108):RETURN
1040 LINE (30,92)-(610,92):RETURN
```

```
1050 LINE (30,76)-(610,76):RETURN
1060 LINE (30,60)-(610,60):RETURN
1070 LINE (30,44)-(610,44):RETURN
1080 LINE (30,28)-(610,28):RETURN
1090 LINE (30,12)-(610,12):RETURN
2000 CLS:PRINT:PRINT"Please follow this direction:"
2005 PRINT:PRINT"1. Disconnect the wire between Op-Amp comparator and
     RC time constant circuit.":PRINT"2. Exchange the polarity when you connect
     into the box interface.":PRINT"3. Connect that wire directly into the channel
     interface that you prefere
2010 PRINT:PRINT"Strike any key to continue.....";
2015 IF INKEY$ = "" GOTO 2015
2020 CLS:GOTO 228
2050 CLS:PRINT:INPUT"Enter your Capacitor (in uF) : ",Q
2055 PRINT:INPUT"Enter your Resistor (in Ohm) : ",R
2060 PRINT:INPUT"Enter input Voltage (in Volt): ",V
2065 D = E%*.5
2070 M = R*Q*10 ^ (-6)*LOG(V/(V-D))
2075 PRINT:PRINT:PRINT"The delay time is ":PRINT:PRINT M" Second."
2080 PRINT:INPUT"Is above time okay.......(Y/N): ";ZZ$
2085 IF ZZ$ = "Y" OR ZZ$ = "y" THEN CLS:GOTO 228
2090 GOTO 2050
3000 RUN "RELAY"
3010 END
```

APPENDIX 5

```
;               DAS8. ASM
;               (C) 1991 ADCdriver Vers. V1.0

movseg          macro reg16, unused, Imm16 ; Fixup for Assembler
                ifidn           reg16, bx
                db              0BBh
                endif
                ifidn           reg16, cx
                db              0B9h
                endif
                ifidn           reg16, dx
                db              0BAh
                endif
                ifidn           reg16, si
                db              0BEh
                endif
                ifidn           reg16, di
                db              0BFh
                endif
                ifidn           reg16, bp
                db              0BDh
                endif
                ifidn           reg16, sp
                db              0BCh
                endif
                ifidn           reg16, BX
                db              0BBH
                endif
                ifidn           reg16, CX
                db              0B9H
                endif
                ifidn           reg16, DX
                db              0BAH
                endif
                ifidn           reg16, SI
                db              0BEH
                endif
                ifidn           reg16, DI
                db              0BFH
                endif
                ifidn           reg16, BP
                db              0BDH
                endif
                ifidn           reg16, SP
                db              0BCH
                endif
                dw              seg Imm16
endm

D_9EF2_0085_E           EQU 85H     ; (=0)
D_9EF2_0089_E           EQU 89H     ; (=0)
D_9EF2_031B_E           EQU 31BH    ; (=3B06H)

;               SEG A
;               SEGMENT BYTE PUBLIC
```

```
ASSUME CS:SEG_A , DS:SEG_A

;               PROGRAM ENTRY POINT

DAS8            PROC            FAR

START:
                NOP
                JMP             L_00BD
COPYRIGHT DB                    '(c) 1991 ADCdriver Vers. V1.0'
D_0021          DB 0, 0         ; xref 9F02:0181, 03B2, 0646, 069A
                                ; 06A7, 0702, 078B, 0AC4
D_0023          DW 9EF2H        ; xref 9F02:00C4, 00F5, 013D, 015C
                                ; 0443, 05D7, 089C, 0B95
D_0025          DW 0            ; xref 9F02:0620, 067C, 06E0, 07B7
D_0027          DW0             ; xref 9F02:062A, 0689, 06ED, 07AD
D_0029          DW0             ; xref 9F02:0634, 0691, 06F5, 0784, 082D
D_002B          DW 0            ; xref 9F02:00EA, 0141, 0169, 01F2
                                ; 022A, 0251, 0262, 043E
                                ; 04B4, 0588, 05BB, 05F0, 0659, 06A3, 06C6
                                ; 070E, 0738, 0746, 07C2, 086A, 0878, 0897
                                ; 0AAC, 0AB4, 0ABC, 0ACC, 0B01, 0B0D
                                ; 0B6B, 0E2D, 0E3E, 0E6D
D_002D          DW 0            ; xref 9F02:01FB, 0452, 04C9, 04D1
                                ; 0574, 05BF, 05F4, 06B1, 071D, 0811, 08AB
                                ; 0B1D, 0B8C, 0C14, 0C1F, 0C54, 0C96, 0DC5
D_002F          DW 0            ; xref 9F02:05C9, 08CA, 0B2E, 0B9E
                                ; 0C27, 0CC6, 0DD5
D_0031          DW 0            ; xref 9F02:08C5, 0B32, 0B49, 0BFD
                                ; 0C23, 0D36, 0DF0
D_0033          DW 0            ; xref 9F02:08C0, 0C1B, 0DF8, 0DFE, 0E13
D_0035          DW 0            ; xref 9F02:08BB, 0C44, 0E21
D_0037          DW 0            ; xref 9F02:018D, 01B0, 027C, 03E5
                                ; 0471, 0768, 08F7, 0BB4, 0F3D, 0FB9
D_0039          DW 0            ; xref 9F02:0412, 087C, 0888, 09D6
                                ; 09F1, 0A8A, 0BE0, 0E59, 0FB1
D_003B          DW 0            ; xref 9F02:0108, 011E, 012E, 0155
                                ; 016E, 01E1, 01EB, 0219, 0223, 023A, 0255, 0266
                                ; 0490, 04B8, 04E4, 055C, 0566, 05AA, 05B4, 05E6
                                ; 05F9, 060A, 064F, 065F, 06BC, 06CA, 0721, 073C
                                ; 074A, 0777, 0815, 086E, 088D, 0913, 0934, 0946
                                ; 09B9, 09C6, 0A98, 0AA7, 0AD1, 0B4D, 0B65
                                ; 0B75, 0C0B, 0C71, 0D71, 0D79
                                ; 0DB9, 0DEA, 0E27, 0E84
D_003D          DW 0            ; xref 9F02:0433
D_003F          DW 0            ; xref 9F02:045E, 04A0, 08B7, 0923
D_0041          DW 0            ; xref 9F02:0117, 01DA
D_0043          DW 0            ; xref 9F02:0512, 0593, 0DDF, 0F54
D_0045          DW 0            ; xref 9F02:00D3, 010E
D_0047          DW 0            ; xref 9F02:0212, 0233, 024D, 0273
                                ; 03C8, 03DC, 046D, 0554, 076C, 08F3, 09DB
                                ; 0A03, 0A85, 0B2A, 0E50, 0EFF
                                ; 0F06, 0F69, 0F76
D_0049          DW 0            ; xref 9F02:019A, 01D0, 0797, 0A9F, 0B39
D_004B          DW 0            ; xref 9F02:0191, 01B4, 0277, 03E0
                                ; 043A, 04F9, 0521, 0540
                                ; 05A0, 0771, 0AFD, 0DA1
                                ; 0E8A, 0E90, 0F9C, 0FA8
```

```
D_004D   DW 0       ; xref 9F02:02D9, 02F7
D_004F   DW 0       ; xref 9F02:02B8, 02BD, 02E4
D_0051   DW 0       ; xref 9F02:02B3
D_0053   DW 0       ; xref 9F02:02AD, 02C5, 02E0, 02EC
D_0055   DW 0       ; xref 9F02:04D6, 0519, 0DA7, 0E38
                    ; 0E46, 0F5A
D_0057   DW 0       ; xref 9F02:0209, 03D6, 054F, 09E5
                    ; 0A0B, 0A7B, 0BAC, 0BBC
                    ; 0BED, 0E63, 0F73
D_0059   DW 7       ; xref 9F02:020D, 03CF, 09E0, 0A07
                    ; 0A80, 0BB9, 0E4A, 0F6D
D_005B   DW 0       ; xref 9F02:0188, 0244, 026E, 0462, 047D, 072B
                    ; 075B, 07D2, 081F, 0842, 08EE, 0B21
                    ; 0C4B, 0C66, 0E5F, 0F83; 0FAC
D_005D   DW 100H    ; xref 9F02:03BB
D_005F   DW 0       ; xref 9F02:0407, 094F, 09D1, 09EA
                    ; 0A8F, 0AD8, 0AF3, 0BD3, 0E53, 0FB5
D_0061   DW 0       ; xref 9F02:08CF, 093B, 09FF, 0C2B
D_0063   DW 0       ; xref 9F02:09F8, 0A28, 0A39, 0A4E, 0A5F, 0A74
D_0065   DW 0       ; xref 9F02:08D4, 094B, 095F, 0975
                    ; 0987, 0998, 09AA, 0C2F
D_0067   DW 0       ; xref 9F02:0992, 09A6, 0A30, 0A45
                    ; 0A56, 0A6B
D_0069   DW 0       ; xref 9F02:0983, 09B5, 0A3E, 0A64
D_006B   DW 0       ; xref 9F02:08DE, 096C, 09BE, 0A1F, 0C33, 0C3B
D_006D   DW 0       ; xref 9F02:08D9, 0963, 0979, 098B
                    ; 099C, 09AE, 0C37
D_006F   DW 0       ; xref 9F02:00DF, 00ED, 0144
         DB 0, 0
D_0073   DW 0       ; xref 9F02:00E6, 0150
         DB 0, 0
D_0077   DW 0, 0    ; xref 9F02:044D, 0459, 0495, 04A7
                    ; 08A6, 08B2, 0918, 092A
D_007B   DW 0A0000000H      ; xref 9F02:0504, 050D, 0539, 056D
                    ; 057E, 0B12, 0DBF, 0DCD, 0F47, 0F50, 0F65
D_007F   DW 0       ; xref 9F02:0534, 0583, 0B16, 0DD1, 0F61
         DB 00H,0A0H
D_0083   DW 0       ; xref 9F02:0BC2
D_0085   DB 0, 0, 0, 0  ; xref 9F02:0BA1, 0CA3, 0CBC, 0CD1, 0D39
D_0089   DW 0       ; xref 9F02:0B8F
D_008B   DW 0       ; xref 9F02:0B98
D_008D   DW OFFSET L_0169   ; Data table (indexed access)
                    ; xref 9F02:0134
D_008F   DW OFFSET L_01EB   ; xref 9F02:0134
D_0091   DW OFFSET L_0223   ; xref 9F02:0134
D_0093   DW OFFSET L_0244   ; xref 9F02:0134
D_0095   DW OFFSET L_025F   ; xref 9F02:0134
D_0097   DW OFFSET L_0431   ; xref 9F02:0134
D_0099   DW OFFSET L_04B1   ; xref 9F02:0134
D_009B   DW OFFSET L_0540   ; xref 9F02:0134
D_009D   DW OFFSET L_0566   ; xref 9F02:0134
D_009F   DW OFFSET L_05B4   ; xref 9F02:0134
D_00A1   DW OFFSET L_05F0   ; xref 9F02:0134
D_00A3   DW OFFSET L_0659   ; xref 9F02:0134
D_00A5   DW OFFSET L_06C6   ; xref 9F02:0134
D_00A7   DW OFFSET L_072B   ; xref 9F02:0134
D_00A9   DW OFFSET L_0746   ; xref 9F02:0134
D_00AB   DW OFFSET L_0781   ; xref 9F02:0134
D_00AD   DW OFFSET L_081F   ; xref 9F02:0134
D_00AF   DW OFFSET L_0878   ; xref 9F02:0134
D_00B1   DW OFFSET L_0897   ; xref 9F02:0134
D_00B3   DW OFFSET L_0A98   ; xref 9F02:0134
D_00B5   DW OFFSET L_0AFD   ; xref 9F02:0134
D_00B7   DW OFFSET L_0B5F   ; xref 9F02:0134
D_00B9   DW OFFSET L_0D9A   ; xref 9F02:0134
D_00BB   DW 9F02H   ; xref 9F02:00C9, 00D7, 00F1, 0101
                    ; 0138, 0B61

L_00BD:             ; xref 9F02:0001
```

```
         PUSH    BP
         MOV     BP,SP
         PUSH    ES
         PUSH    BX
         PUSH    SI
         PUSH    DI
         MOV     CS:D_0023,DS           ; (=9EF2H)
         MOV     CS:D_00BB,CS           ; (=9F02H)
         MOV     DI,[BP+0AH]
         MOV     AX,[DI]
         MOV     CS:D_0045,AX           ; (=0)
         MOV     DS,CS:D_00BB           ; (=9F02H)
         MOV     CX,[BP+8]
         MOV     D_006F,CX              ; (=0)
         MOV     CX,[BP+6]
         MOV     D_0073,CX              ; (=0)
         MOV     DI,OFFSET D_002B       ; (=0)
         MOV     SI,D_006F              ; (=0)
         MOV     ES,D_00BB              ; (=9F02H)
         MOV     DS,D_0023              ; (=9EF2H)
         PUSHF                          ; Push flags
         CLD                            ; Clear direction
         MOV     CX,6
         REP     MOVSW     ; Rep when cx 0 Mov [si] to es:[di]
         POPF                           ; Pop flags
         MOV     DS,CS:D_00BB           ; (=9F02H)
         PUSH    CS
         POP     DS
         MOV     D_003B,2               ; (=0)
         MOV     SI,D_0045              ; (=0)
         CMP     SI,16H
         JA      L_0129                 ; Jump if above
         CMP     D_0041,1               ; (=0)
         JE      L_012C                 ; Jump if equal
         MOV     D_003B,1               ; (=0)
         CMP     SI,0
         JE      L_012C                 ; Jump if equal
L_0129:                                 ; xref 9F02:0115
         JMP     SHORT L_0138
         DB 90H
L_012C:                        ; xref 9F02:011C, 0127
         SHL     SI,1                   ; Shift w/zeros fill
         MOV     D_003B,0               ; (=0)
         JMP     WORD PTR D_008D[SI]    ;*(=169H) 23 entries
L_0138:                        ; xref 9F02:0129, 01E8, 0220, 0241
                               ; 025C, 026B, 04AE, 04EE
                               ; 0563, 05B1, 05ED, 0656
                               ; 06C3, 0728, 0743, 077E
                               ; 081C, 0875, 0894, 08E8
                               ; 0931, 0AFA, 0B54, 0B81, 0E95
         MOV     DS,CS:D_00BB           ; (=9F02H)
         MOV     ES,D_0023              ; (=9EF2H)
         MOV     SI,OFFSET D_002B       ; (=0)
         MOV     DI,D_006F              ; (=0)
         PUSHF                          ; Push flags
         CLD                            ; Clear direction
         MOV     CX,6
         REP     MOVSW     ; Rep when cx 0 Mov [si] to es:[di]
         POPF                           ; Pop flags
         MOV     DI,CS:D_0073           ; (=0)
         MOV     AX,CS:D_003B           ; (=0)
         MOV     ES:[DI],AX
         MOV     DS,CS:D_0023           ; (=9EF2H)
         POP     DI
         POP     SI
         POP     BX
         POP     ES
         POP     BP
```

```
        RETF    6                       ; Return far

;       Indexed Entry Point

L_0169:                                 ; xref 9F02:008D, 0134
        MOV     DX,CS:D_002B            ; (=0)
        MOV     CS:D_003B,3             ; (=0)
        CMP     DX,100H
        JL      L_01E8                  ; Jump if
        CMP     DX,3F8H
        JG      L_01E8                  ; Jump if
        MOV     WORD PTR CS:D_0021,DX ; (=0)
        INC     DX
        INC     DX
        MOV     CS:D_005B,DX            ; (=0)
        MOV     AX,CS:D_0037            ; (=0)
        OR      AX,CS:D_004B            ; (=0)
        OR      AX,5
        OUT     DX,AL           ; port 2, DMA-1 bas&add ch 1
        MOV     CS:D_0049,0             ; (=0)
        INC     DX
        JMP     SHORT $+3               ; delay for I/O
        NOP
        IN      AL,DX           ; port 3, DMA-1 bas&cnt ch 1
        MOV     CL,4
        SHR     AL,CL                   ; Shift w/zeros fill
        AND     AL,7
        CMP     AL,5
        JNE     L_01D7                  ; Jump if not equal
        MOV     AX,CS:D_0037            ; (=0)
        OR      AX,CS:D_004B            ; (=0)
        OR      AX,2
        DEC     DX
        JMP     SHORT $+3               ; delay for I/O
        NOP
        OUT     DX,AL           ; port 2, DMA-1 bas&add ch 1
        INC     DX
        JMP     SHORT $+3               ; delay for I/O
        NOP
        IN      AL,DX           ; port 3, DMA-1 bas&cnt ch 1
        MOV     CL,4
        SHR     AL,CL                   ; Shift w/zeros fill
        AND     AL,7
        CMP     AL,2
        JNE     L_01D7                  ; Jump if not equal
        MOV     CS:D_0049,1             ; (=0)
L_01D7:                         ; xref 9F02:01AE, 01CE
        CALL    S_026E
        MOV     CS:D_0041,1             ; (=0)
        MOV     CS:D_003B,0             ; (=0)
L_01E8:                         ; xref 9F02:0179, 017F
        JMP     L_0138

;       Indexed Entry Point

L_01EB:                         ; xref 9F02:008F, 0134
        MOV     CS:D_003B,4             ; (=0)
        MOV     AX,CS:D_002B            ; (=0)
        CMP     AX,7
        JA      L_0220                  ; Jump if above
        MOV     CX,CS:D_002D            ; (=0)
        CMP     CX,AX
        JB      L_0220                  ; Jump if below
        CMP     CX,7
        JA      L_0220                  ; Jump if above
        MOV     CS:D_0057,AX            ; (=0)
        MOV     CS:D_0059,CX            ; (=7)
        MOV     CS:D_0047,AX            ; (=0)
        CALL    S_026E
        MOV     CS:D_003B,0             ; (=0)
L_0220:                         ; xref 9F02:01F9, 0202, 0207
        JMP     L_0138

;       Indexed Entry Point

L_0223:                         ; xref 9F02:0091, 0134
        MOV     CS:D_003B,5             ; (=0)
        MOV     AX,CS:D_002B            ; (=0)
        CMP     AX,7
        JA      L_0241                  ; Jump if above
        MOV     CS:D_0047,AX            ; (=0)
        CALL    S_026E
        MOV     CS:D_003B,0             ; (=0)
L_0241:                                 ; xref 9F02:0231
        JMP     L_0138

;       Indexed Entry Point

L_0244:                         ; xref 9F02:0093, 0134
        MOV     DX,CS:D_005B            ; (=0)
        IN      AL,DX           ; port 0, DMA-1 bas&add ch 0
        AND     AX,7
        MOV     CS:D_0047,AX            ; (=0)
        MOV     CS:D_002B,AX            ; (=0)
        MOV     CS:D_003B,0             ; (=0)
        JMP     L_0138

;       Indexed Entry Point

L_025F:                         ; xref 9F02:0095, 0134
        CALL    S_03B2
        MOV     CS:D_002B,AX            ; (=0)
        MOV     CS:D_003B,BX            ; (=0)
        JMP     L_0138
DAS8    ENDP
;
;       SUBROUTINE
;
; Called from: 9F02:01D7, 0216, 0237, 0559, 05A7, 0A0F,
;       0A94
;
S_026E  PROC    NEAR
        MOV     DX,CS:D_005B            ; (=0)
        MOV     AX,CS:D_0047            ; (=0)
        OR      AX,CS:D_004B            ; (=0)
        OR      AX,CS:D_0037            ; (=0)
        OUT     DX,AL           ; port 0, DMA-1 bas&add ch 0
        RETN
S_026E  ENDP
;
;       SUBROUTINE
;
;
;       Called from: 9F02:02C2, 02E9
;
S_0283  PROC    NEAR
        PUSH    AX
        PUSH    DX
        PUSH    DI
        PUSH    DS
        SHL     AX,1                    ; Shift w/zeros fill
        SHL     AX,1                    ; Shift w/zeros fill
        MOV     DI,AX
        XOR     AX,AX                   ; Zero register
        MOV     DS,AX
        MOV     AX,ES:[BX]
```

```
            MOV     DX,ES:[BX+2]
            PUSHF                           ; Push flags
            CLI                             ; Disable interrupts
            XCHG    AX,[DI]
            XCHG    DX,[DI+2]
            POPF                            ; Pop flags
            MOV     ES:[BX],AX
            MOV     ES:[BX+2],DX
            POP     DS
            POP     DI
            POP     DX
            POP     AX
            RETN
S_0283      ENDP
;
;           SUBROUTINE
;
;           Called from: 9F02:04E1, 0D68, 0E79
;
S_02AC      PROC    NEAR
            PUSH    ES
            MOV     CS:D_0053,AX            ; (=0)
            PUSH    BX
            PUSH    CS
            POP     CS:D_0051               ; (=0)
            POP     CS:D_004F               ; (=0)
            MOV     BX,OFFSET D_004F        ; (=0)
            PUSH    CS
            POP     ES
            CALL    S_0283
            MOV     AX,CS:D_0053            ; (=0)
            CMP     AX,0AH
            JB      L_02DD                  ; Jump if below
            CMP     AX,0FH
            JA      L_02DD                  ; Jump if above
            SUB     AX,8
            CALL    S_02FF
            MOV     CS:D_004D,AX            ; (=0)
L_02DD:                             ; xref 9F02:02CC, 02D1
            POP     ES
            RETN
S_02AC      ENDP
;
;           SUBROUTINE
;
;           Called from: 9F02:054C, 0D8E, 0DAD
;
S_02DF      PROC    NEAR
            PUSH    ES
            MOV     AX,CS:D_0053            ; (=0)
            MOV     BX,OFFSET D_004F        ; (=0)
            PUSH    CS
            POP     ES
            CALL    S_0283
            MOV     AX,CS:D_0053            ; (=0)
            JC      L_02FD                  ; Jump if carry Set
            CMP     AX,0FH
            JA      L_02FD                  ; Jump if above
            MOV     AX,CS:D_004D            ; (=0)
            OUT     21H,AL          ; port 21H, 8259-1 int comands
L_02FD:                             ; xref 9F02:02F0, 02F5
            POP     ES
            RETN
S_02DF      ENDP
;
;           SUBROUTINE
;
```

```
;           Called from: 9F02:02D6
;S_02FF     PROC    NEAR
            PUSH    BX
            PUSH    CX
            MOV     CX,AX
            MOV     BX,1
            SHL     BL,CL                   ; Shift w/zeros fill
            XOR     BX,0FFH
            IN      AL,21H          ; port 21H, 8259-1 int IMR
            MOV     CX,AX
            AND     AL,BL
            JMP     SHORT $+2               ; delay for I/O
            OUT     21H,AL          ; port 21H, 8259-1 int comands
                                    ; al = 0F8H, inhibit IRQ3-7
            MOV     AX,CX
            POP     CX
            POP     BX
            RETN
S_02FF      ENDP

D_031B      DW 0    ; xref 9F02:0C5E, 0D5C, 0D88
            DB      50H, 53H, 51H, 52H, 57H, 06H
            DB      1EH, 2EH, 8EH, 1EH,0BBH, 00H
            DB      83H, 3EH, 1BH, 03H, 00H, 75H
            DB      16H, 8BH, 16H, 21H, 00H, 8BH
            DB      0EH, 83H, 00H,0C4H, 3EH, 89H
            DB      00H, 8BH, 1EH, 85H, 00H,0F7H
            DB      0C3H,0FFH,0FFH, 75H, 03H,0EBH
            DB      5EH, 90H,0FFH, 06H, 1BH, 03H
            DB      42H, 42H, 91H,0EEH, 91H,0F6H
            DB      0C5H, 07H, 74H, 02H,0FEH,0C1H
L_0359:
            DEC     DX
            DEC     DX
            INC     DX
            INC     DX
L_035D:                                     ; xref 9F02:0360
            IN      AL,DX           ; port 0, DMA-1 bas&add ch 0
            TEST    AL,10H
            JNZ     L_035D                  ; Jump if not zero
            DEC     DX
            DEC     DX
            INC     DX
            OUT     DX,AL ; ??IO  NON-STANDARD I/O PORT.
            JMP     SHORT L_0372
L_0368:                                     ; xref 9F02:0392
            INC     DX
            OUT     DX,AL           ; port 0, DMA-1 bas&add ch 0
            MOV     ES:[DI],AX
            INC     DI
            INC     DI
            DEC     BX
            JZ      L_039A                  ; Jump if zero
L_0372:                                     ; xref 9F02:0366
            INC     DX
            TEST    CH,7
            JZ      L_0380                  ; Jump if zero
            XCHG    AX,CX
            OUT     DX,AL           ; port 1, DMA-1 bas&cnt ch 0
            INC     AL
            XCHG    AX,CX
            JMP     SHORT L_0380
            DB      90H
L_0380:                             ; xref 9F02:0376, 037D, 0383
            IN      AL,DX           ; port 1, DMA-1 bas&cnt ch 0
            TEST    AL,80H
            JNZ     L_0380                  ; Jump if not zero
```

```
        DEC     DX
        IN      AL,DX       ; port 0, DMA-1 bas&add ch 0
        MOV     AH,AL
        DEC     DX
        IN      AL,DX       ; ??IO  NON-STANDARD I/O PORT.
        TEST    CH,7
        JZ      L_0394                  ; Jump if zero
        DEC     CH
        JMP     SHORT L_0368
L_0394:                                 ; xref 9F02:038E
        MOV     ES:[DI],AX
        INC     DI
        INC     DI
        DEC     BX
L_039A:                                 ; xref 9F02:0370
        MOV     DS:D_9EF2_0089_E,DI     ; (=0)
        MOV     DS:D_9EF2_0085_E,BX     ; (=0)
        DEC     WORD PTR DS:D_9EF2_031B_E ; (=3B06H)
L_03A6:
        POP     DS
        POP     ES
        POP     DI
        POP     DX
        POP     CX
        POP     BX
        MOV     AL,20H                  ; ' '
        OUT     20H,AL      ; port 20H, 8259-1 int command
                            ; al = 20H, end of interrupt
        POP     AX
        IRET                            ; Interrupt return
;
;       SUBROUTINE
;
; Called from:  9F02:025F, 0488, 0501, 090B, 0A1C, 0F03
;
S_03B2  PROC    NEAR
        MOV     DX,WORD PTR CS:D_0021 ; (=0)
        INC     DX
        PUSHF                           ; Push flags
        CLI                             ; Disable interrupts
        OUT     DX,AL       ; port 1, DMA-1 bas&cnt ch 0
        MOV     CX,CS:D_005D            ; (=100H)
        INC     DX
        IN      AL,DX       ; port 2, DMA-1 bas&add ch 1
        POPF                            ; Pop flags
        AND     AX,80H
        JZ      L_042A                  ; Jump if zero
        MOV     BX,CS:D_0047            ; (=0)
        MOV     AX,BX
        CMP     AX,CS:D_0059            ; (=7)
        JB      L_03DB                  ; Jump if below
        MOV     AX,CS:D_0057            ; (=0)
        DEC     AX
L_03DB:                                 ; xref 9F02:03D4
        INC     AX
        MOV     CS:D_0047,AX            ; (=0)
        OR      AX,CS:D_004B            ; (=0)
        OR      AX,CS:D_0037            ; (=0)
        OUT     DX,AL       ; port 2, DMA-1 bas&add ch 1
        NOP
        NOP
        JMP     SHORT L_03F0
        DB      90H
L_03F0:                     ; xref 9F02:03ED, 03F7
        IN      AL,DX       ; port 2, DMA-1 bas&add ch 1
        DEC     CX
        JZ      L_042A                  ; Jump if zero
        AND     AX,80H
        JNZ     L_03F0                  ; Jump if not zero
        DEC     DX
        IN      AL,DX       ; port 1, DMA-1 bas&cnt ch 0
        MOV     AH,AL
        DEC     DX
        IN      AL,DX       ; port 0, DMA-1 bas&add ch 0
        SHR     AX,1                    ; Shift w/zeros fill
        SHR     AX,1                    ; Shift w/zeros fill
        SHR     AX,1                    ; Shift w/zeros fill
        SHR     AX,1                    ; Shift w/zeros fill
        CMP     CS:D_005F,1             ; (=0)
        JE      L_0412                  ; Jump if equal
        SUB     AX,800H
L_0412:                                 ; xref 9F02:040D
        CMP     CS:D_0039,1             ; (=0)
        JNE     L_0427                  ; Jump if not equal
        SHL     BL,1                    ; Shift w/zeros fill
        SHL     BL,1                    ; Shift w/zeros fill
        SHL     BL,1                    ; Shift w/zeros fill
        SHL     BL,1                    ; Shift w/zeros fill
        AND     AH,8FH
        OR      AH,BL
L_0427:                                 ; xref 9F02:0418
        XOR     BX,BX                   ; Zero register
        RETN
L_042A:                     ; xref 9F02:03C6, 03F2
        MOV     AX,7FFFH
        MOV     BX,6
        RETN
S_03B2  ENDP

;       Indexed Entry Point

L_0431:                     ; xref 9F02:0097, 0134
        XOR     AX,AX                   ; Zero register
        MOV     CS:D_003D,AX            ; (=0)
        MOV     AX,8
        MOV     CS:D_004B,AX            ; (=0)
        PUSH    CS:D_002B               ; (=0)
        PUSH    CS:D_0023               ; (=9EF2H)
        POP     WORD PTR CS:D_0077+2; (=0)
        POP     CS:D_0077               ; (=0)
        MOV     AX,CS:D_002D            ; (=0)
        DEC     AX
        SHL     AX,1                    ; Shift w/zeros fill
        ADD     AX,CS:D_0077            ; (=0)
        MOV     CS:D_003F,AX            ; (=0)
L_0462:                                 ; xref 9F02:046B
        MOV     DX,CS:D_005B            ; (=0)
        IN      AL,DX       ; port 0, DMA-1 bas&add ch 0
        AND     AX,10H
        JZ      L_0462                  ; Jump if zero
        MOV     AX,CS:D_0047            ; (=0)
        OR      AX,CS:D_0037            ; (=0)
        OR      AX,8
        OUT     DX,AL       ; port 0, DMA-1 bas&add ch 0
        JMP     SHORT L_047D
        DB      90H
L_047D:                     ; xref 9F02:047A, 0486, 04AC
        MOV     DX,CS:D_005B            ; (=0)
        IN      AL,DX       ; port 0, DMA-1 bas&add ch 0
        AND     AX,8
        JZ      L_047D                  ; Jump if zero
        CALL    S_03B2
        CMP     BX,0
        JE      L_0495                  ; Jump if equal
        MOV     CS:D_003B,BX            ; (=0)
L_0495:                                 ; xref 9F02:048E
```

```
        LDS     DI,DWORD PTR CS:D_0077; (=0) Load 32 bit ptr
        MOV     [DI],AX
        INC     DI
        INC     DI
        PUSH    CS
        POP     DS
        CMP     DI,CS:D_003F            ; (=0)
        JA      L_04AE                  ; Jump if above
        MOV     CS:D_0077,DI            ; (=0)
        JMP     SHORT L_047D
L_04AE:                                 ; xref 9F02:04A5
        JMP     L_0138

;       Indexed Entry Point

L_04B1:                 ; xref 9F02:0099, 0134
        PUSH    AX
        PUSH    BX
        PUSH    ES
        MOV     AX,CS:D_002B            ; (=0)
        MOV     CS:D_003B,7             ; (=0)
        CMP     AX,2
        JB      L_04EB                  ; Jump if below
        CMP     AX,7
        JA      L_04EB                  ; Jump if above
        CMP     CS:D_002D,1             ; (=0)
        JA      L_04EB                  ; Jump if above
        PUSH    CS:D_002D               ; (=0)
        POP     CS:D_0055               ; (=0)
        ADD     AX,8
        MOV     BX,4F1H
        CALL    S_02AC
        MOV     CS:D_003B,0             ; (=0)
L_04EB:                 ; xref 9F02:04C2, 04C7, 04CF
        POP     ES
        POP     BX
        POP     AX
        JMP     L_0138
        PUSH    AX
        PUSH    BX
        PUSH    CX
        PUSH    DX
        PUSH    DS
        PUSH    DI
        PUSH    CS
        POP     DS
        CMP     CS:D_004B,8             ; (=0)
        JNE     L_0528                  ; Jump if not equal
        CALL    S_03B2
        LDS     DI,CS:D_007B ; (=0) Load 32 bit ptr
        MOV     [DI],AX
        INC     DI
        INC     DI
        MOV     WORD PTR CS:D_007B,DI ; (=0)
        CMP     DI,CS:D_0043            ; (=0)
        JB      L_0528                  ; Jump if below
        CMP     CS:D_0055,0             ; (=0)
        JNE     L_0534                  ; Jump if not equal
        MOV     CS:D_004B,0             ; (=0)
L_0528:                 ; xref 9F02:04FF, 0517, 053E
        POP     DI
        POP     DS
        POP     DX
        POP     CX
        POP     BX
        MOV     AL,20H          ;''
        OUT     20H,AL          ; port 20H, 8259-1 int command
                                ; al = 20H, end of interrupt
        POP     AX
        STI                                     ; Enable interrupts
        IRET                                    ; Interrupt return
L_0534:                                         ; xref 9F02:051F
        PUSH    CS:D_007F                       ; (=0)
        POP     WORD PTR CS:D_007B              ; (=0)
        JMP     SHORT L_0528

;       Indexed Entry Point

L_0540:                 ; xref 9F02:009B, 0134
        MOV     CS:D_004B,0             ; (=0)
        MOV     CX,800H
L_054A:                                 ; xref 9F02:054A
        LOOP    L_054A                  ; Loop if cx 0
        CALL    S_02DF
        PUSH    CS:D_0057               ; (=0)
        POP     CS:D_0047               ; (=0)
        CALL    S_026E
        MOV     CS:D_003B,0             ; (=0)
        JMP     L_0138

;       Indexed Entry Point

L_0566:                 ; xref 9F02:009D, 0134
        MOV     CS:D_003B,8             ; (=0)
        MOV     WORD PTR CS:D_007B,0 ; (=0)
        PUSH    CS:D_002D               ; (=0)
        POP     WORD PTR CS:D_007B+2 ; (=0A000H)
        PUSH    WORD PTR CS:D_007B      ; (=0)
        POP     CS:D_007F               ; (=0)
        MOV     AX,CS:D_002B            ; (=0)
        CMP     AX,1
        JL      L_05B1                  ; Jump if
        SHL     AX,1                    ; Shift w/zeros fill
        MOV     CS:D_0043,AX            ; (=0)
        MOV     AX,WORD PTR CS:D_007B+2 ; (=0A000H)
        CMP     AX,0
        JL      L_05B1                  ; Jump if
        MOV     CS:D_004B,8             ; (=0)
        CALL    S_026E
        MOV     CS:D_003B,0             ; (=0)
L_05B1:                 ; xref 9F02:058F, 059E
        JMP     L_0138

;       Indexed Entry Point

L_05B4:                 ; xref 9F02:009F, 0134
        MOV     CS:D_003B,9             ; (=0)
        MOV     AX,CS:D_002B            ; (=0)
        MOV     CX,CS:D_002D            ; (=0)
        CMP     CX,1
        JL      L_05ED                  ; Jump if
        MOV     SI,CS:D_002F            ; (=0)
        CMP     SI,0
        JL      L_05ED                  ; Jump if
        SHL     SI,1                    ; Shift w/zeros fill
        MOV     DI,AX
        MOV     ES,CS:D_0023            ; (=9EF2H)
        MOV     DS,WORD PTR CS:D_007B+2 ; (=0A000H)
        CLD                             ; Clear direction
        REP     MOVSW; Rep when cx 0 Mov [si] to es:[di]
        PUSH    CS
        POP     DS
        MOV     CS:D_003B,0             ; (=0)
L_05ED:                 ; xref 9F02:05C7, 05D1
        JMP     L_0138
```

```
;               Indexed Entry Point

L_05F0:                         ; xref 9F02:00A1, 0134
        MOV     AX,CS:D_002B            ; (=0)
        MOV     CX,CS:D_002D            ; (=0)
        MOV     CS:D_003B,0AH           ; (=0)
        CMP     AX,0
        JL      L_0656                  ; Jump if
        CMP     AX,2
        JG      L_0656                  ; Jump if
        MOV     CS:D_003B,0BH           ; (=0)
        CMP     CX,0
        JL      L_0656                  ; Jump if
        CMP     CX,5
        JG      L_0656                  ; Jump if
        CMP     AX,0
        JNE     L_0625                  ; Jump if not equal
        MOV     CS:D_0025,CX            ; (=0)
L_0625:                                 ; xref 9F02:061E
        CMP     AX,1
        JNE     L_062F                  ; Jump if not equal
        MOV     CS:D_0027,CX            ; (=0)
L_062F:                                 ; xref 9F02:0628
        CMP     AX,2
        JNE     L_0639                  ; Jump if not equal
        MOV     CS:D_0029,CX            ; (=0)
L_0639:                                 ; xref 9F02:0632
        MOV     BX,CX
        MOV     CL,6
        SHL     AX,CL                   ; Shift w/zeros fill
        SHL     BX,1                    ; Shift w/zeros fill
        OR      AX,BX
        OR      AX,30H
        MOV     DX,WORD PTR CS:D_0021 ; (=0)
        ADD     DX,7
        OUT     DX,AL           ; port 7, DMA-1 bas&cnt ch 3
        MOV     CS:D_003B,0             ; (=0)
L_0656:                         ; xref 9F02:0603, 0608, 0614, 0619
        JMP     L_0138

;               Indexed Entry Point

L_0659:                         ; xref 9F02:00A3, 0134
        MOV     AX,CS:D_002B            ; (=0)
        MOV     DX,AX
        MOV     CS:D_003B,0AH           ; (=0)
        CMP     AX,0
        JL      L_06C3                  ; Jump if
        CMP     AX,2
        JG      L_06C3                  ; Jump if
        MOV     CL,6
        SHL     AX,CL                   ; Shift w/zeros fill
        OR      AX,30H
        CMP     DX,0
        JNE     L_0684                  ; Jump if not equal
        MOV     DX,CS:D_0025            ; (=0)
        JMP     SHORT L_0696
        DB      90H
L_0684:                                 ; xref 9F02:067A
        CMP     DX,1
        JNE     L_0691                  ; Jump if not equal
        MOV     DX,CS:D_0027            ; (=0)
        JMP     SHORT L_0696
        DB      90H
L_0691:                                 ; xref 9F02:0687
        MOV     DX,CS:D_0029            ; (=0)
L_0696:                         ; xref 9F02:0681, 068E
```

```
        SHL     DX,1                    ; Shift w/zeros fill
        OR      AX,DX
        MOV     DX,WORD PTR CS:D_0021 ; (=0)
        ADD     DX,7
        OUT     DX,AL           ; port 7, DMA-1 bas&cnt ch 3
        MOV     AX,CS:D_002B            ; (=0)
        MOV     DX,WORD PTR CS:D_0021 ; (=0)
        ADD     AX,4
        ADD     DX,AX
        MOV     AX,CS:D_002D            ; (=0)
        OUT     DX,AL           ; port 4, DMA-1 bas&add ch 2
        MOV     AL,AH
        JMP     SHORT $+3               ; delay for I/O
        NOP
        OUT     DX,AL           ; port 4, DMA-1 bas&add ch 2
        MOV     CS:D_003B,0             ; (=0)
L_06C3:                         ; xref 9F02:0669, 066E
        JMP     L_0138

;               Indexed Entry Point

L_06C6:                         ; xref 9F02:00A5, 0134
        MOV     AX,CS:D_002B            ; (=0)
        MOV     CS:D_003B,0AH           ; (=0)
        CMP     AX,0
        JL      L_0728                  ; Jump if
        CMP     AX,2
        JG      L_0728                  ; Jump if
        CMP     AX,0
        JNE     L_06E8                  ; Jump if not equal
        MOV     BX,CS:D_0025            ; (=0)
        JMP     SHORT L_06FA
        DB      90H
L_06E8:                                 ; xref 9F02:06DE
        CMP     AX,1
        JNE     L_06F5                  ; Jump if not equal
        MOV     BX,CS:D_0027            ; (=0)
        JMP     SHORT L_06FA
        DB      90H
L_06F5:                                 ; xref 9F02:06EB
        MOV     BX,CS:D_0029            ; (=0)
L_06FA:                         ; xref 9F02:06E5, 06F2
        SHL     BX,1                    ; Shift w/zeros fill
        MOV     CL,6
        SHL     AX,CL                   ; Shift w/zeros fill
        OR      AX,BX
        MOV     DX,WORD PTR CS:D_0021 ; (=0)
        ADD     DX,7
        OUT     DX,AL           ; port 7, DMA-1 bas&cnt ch 3
        SUB     DX,3
        MOV     BX,CS:D_002B            ; (=0)
        ADD     DX,BX
        IN      AL,DX           ; port 4, DMA-1 bas&add ch 2
        MOV     AH,AL
        JMP     SHORT $+2               ; delay for I/O
        IN      AL,DX           ; port 4, DMA-1 bas&add ch 2
        XCHG    AH,AL
        MOV     CS:D_002D,AX            ; (=0)
        MOV     CS:D_003B,0             ; (=0)
L_0728:                         ; xref 9F02:06D4, 06D9
        JMP     L_0138

;               Indexed Entry Point

L_072B:                         ; xref 9F02:00A7, 0134
        MOV     DX,CS:D_005B            ; (=0)
        IN      AL,DX           ; port 0, DMA-1 bas&add ch 0
        MOV     CL,4
```

```
        SAR     AX,CL                   ; Shift w/sign fill
        AND     AX,7
        MOV     CS:D_002B,AX            ; (=0)
        MOV     CS:D_003B,0             ; (=0)
        JMP     L_0138

;       Indexed Entry Point

L_0746:                         ; xref 9F02:00A9, 0134
        MOV     AX,CS:D_002B            ; (=0)
        MOV     CS:D_003B,0CH           ; (=0)
        CMP     AX,0
        JL      L_077E                  ; Jump if
        CMP     AX,0FH
        JG      L_077E                  ; Jump if
        MOV     DX,CS:D_005B            ; (=0)
        SHL     AX,1                    ; Shift w/zeros fill
        SHL     AX,1                    ; Shift w/zeros fill
        SHL     AX,1                    ; Shift w/zeros fill
        SHL     AX,1                    ; Shift w/zeros fill
        MOV     CS:D_0037,AX            ; (=0)
        OR      AX,CS:D_0047            ; (=0)
        OR      AX,CS:D_004B            ; (=0)
        OUT     DX,AL           ; port 0, DMA-1 bas&add ch 0
        MOV     CS:D_003B,0             ; (=0)
L_077E:                         ; xref 9F02:0754, 0759
        JMP     L_0138

;       Indexed Entry Point

L_0781:                         ; xref 9F02:00AB, 0134
        MOV     AX,3
        MOV     CS:D_0029,AX            ; (=0)
        MOV     AX,0B6H
        MOV     DX,WORD PTR CS:D_0021 ; (=0)
        ADD     DX,7
        OUT     DX,AL           ; port 7, DMA-1 bas&cnt ch 3
        MOV     AX,952H
        CMP     CS:D_0049,1             ; (=0)
        JNE     L_07A2                  ; Jump if not equal
        MOV     AX,3E8H
L_07A2:                                 ; xref 9F02:079D
        DEC     DX
        OUT     DX,AL           ; port 6, DMA-1 bas&add ch 3
        JMP     SHORT $+2               ; delay for I/O
        MOV     AL,AH
        OUT     DX,AL           ; port 6, DMA-1 bas&add ch 3
        INC     DX
        MOV     AX,3
        MOV     CS:D_0027,AX            ; (=0)
        MOV     AX,76H
        OUT     DX,AL           ; port 7, DMA-1 bas&cnt ch 3
        JMP     SHORT $+2               ; delay for I/O
        MOV     CS:D_0025,0             ; (=0)
        MOV     AX,30H
        OUT     DX,AL           ; port 7, DMA-1 bas&cnt ch 3
        MOV     AX,CS:D_002B            ; (=0)
        SHL     AX,1                    ; Shift w/zeros fill
        DEC     DX
        DEC     DX
        OUT     DX,AL           ; port 5, DMA-1 bas&cnt ch 2
        JMP     SHORT $+2               ; delay for I/O
        MOV     AL,AH
        OUT     DX,AL           ; port 5, DMA-1 bas&cnt ch 2
        JMP     SHORT $+2               ; delay for I/O
        MOV     DX,CS:D_005B            ; (=0)
L_07D7:                                 ; xref 9F02:07DD
        IN      AL,DX           ; port 0, DMA-1 bas&add ch 0
        JMP     SHORT $+2               ; delay for I/O
        AND     AX,20H
        JZ      L_07D7                  ; Jump if zero
L_07DF:                                 ; xref 9F02:07E5
        IN      AL,DX           ; port 0, DMA-1 bas&add ch 0
        JMP     SHORT $+2               ; delay for I/O
        AND     AX,20H
        JNZ     L_07DF                  ; Jump if not zero
        MOV     AX,0FFFFH
        INC     DX
        INC     DX
        OUT     DX,AL           ; port 2, DMA-1 bas&add ch 1
        JMP     SHORT $+2               ; delay for I/O
        OUT     DX,AL           ; port 2, DMA-1 bas&add ch 1
        DEC     DX
        DEC     DX
L_07F2:                                 ; xref 9F02:07F8
        IN      AL,DX           ; port 0, DMA-1 bas&add ch 0
        JMP     SHORT $+2               ; delay for I/O
        AND     AX,20H
        JZ      L_07F2                  ; Jump if zero
L_07FA:                                 ; xref 9F02:0800
        IN      AL,DX           ; port 0, DMA-1 bas&add ch 0
        JMP     SHORT $+2               ; delay for I/O
        AND     AX,20H
        JNZ     L_07FA                  ; Jump if not zero
        INC     DX
        INC     DX
        IN      AL,DX           ; port 2, DMA-1 bas&add ch 1
        JMP     SHORT $+2               ; delay for I/O
        MOV     CL,AL
        IN      AL,DX           ; port 2, DMA-1 bas&add ch 1
        MOV     CH,AL
        MOV     AX,0FFFFH
        SUB     AX,CX
        MOV     CS:D_002D,AX            ; (=0)
        MOV     CS:D_003B,0             ; (=0)
        JMP     L_0138

;       Indexed Entry Point

L_081F:                         ; xref 9F02:00AD, 0134
        MOV     DX,CS:D_005B            ; (=0)
L_0824:                         ; xref 9F02:0828, 084B
        IN      AL,DX           ; port 0, DMA-1 bas&add ch 0
        AND     AX,20H
        JNZ     L_0824                  ; Jump if not zero
        ADD     DX,5
        MOV     CS:D_0029,0             ; (=0)
        MOV     AX,0B0H
        OUT     DX,AL           ; port 5, DMA-1 bas&cnt ch 2
        DEC     DX
        MOV     AL,0FFH
        OUT     DX,AL           ; port 4, DMA-1 bas&add ch 2
        NOP
        NOP
        JMP     SHORT $+3               ; delay for I/O
        NOP
        OUT     DX,AL           ; port 4, DMA-1 bas&add ch 2
        MOV     DX,CS:D_005B            ; (=0)
        IN      AL,DX           ; port 0, DMA-1 bas&add ch 0
        AND     AX,20H
        JNZ     L_0824                  ; Jump if not zero
L_084D:                                 ; xref 9F02:0851
        IN      AL,DX           ; port 0, DMA-1 bas&add ch 0
        AND     AX,20H
        JZ      L_084D                  ; Jump if zero
L_0853:                                 ; xref 9F02:0857
```

```
        IN      AL,DX           ; port 0, DMA-1 bas&add ch 0
        AND     AX,20H
        JNZ     L_0853                  ; Jump if not zero
        ADD     DX,4
        IN      AL,DX           ; port 4, DMA-1 bas&add ch 2
        MOV     CL,AL
        JMP     SHORT $+3               ; delay for I/O
        NOP
        IN      AL,DX           ; port 4, DMA-1 bas&add ch 2
        MOV     CH,AL
        MOV     AX,0FFFFH
        SUB     AX,CX
        MOV     CS:D_002B,AX            ; (=0)
        MOV     CS:D_003B,0             ; (=0)
        JMP     L_0138

;       Indexed Entry Point

L_0878:                         ; xref 9F02:00AF, 0134
        MOV     AX,CS:D_002B            ; (=0)
        MOV     CS:D_0039,0             ; (=0)
        CMP     AX,0
        JLE     L_088D                  ; Jump if  or =
        INC     CS:D_0039               ; (=0)
L_088D:                                 ; xref 9F02:0886
        MOV     CS:D_003B,0             ; (=0)
        JMP     L_0138

;       Indexed Entry Point

L_0897:                         ; xref 9F02:00B1, 0134
        PUSH    CS:D_002B               ; (=0)
        PUSH    CS:D_0023               ; (=9EF2H)
        POP     WORD PTR CS:D_0077+2; (=0)
        POP     CS:D_0077               ; (=0)
        MOV     AX,CS:D_002D            ; (=0)
        DEC     AX
        SHL     AX,1                    ; Shift w/zeros fill
        ADD     AX,CS:D_0077            ; (=0)
        MOV     CS:D_003F,AX            ; (=0)
        PUSH    CS:D_0035               ; (=0)
        PUSH    CS:D_0033               ; (=0)
        PUSH    CS:D_0031               ; (=0)
        PUSH    CS:D_002F               ; (=0)
        POP     CS:D_0061               ; (=0)
        POP     CS:D_0065               ; (=0)
        POP     CS:D_006D               ; (=0)
        POP     CS:D_006B               ; (=0)
        CALL    S_0934
        JNC     L_08EB                  ; Jump if carry=0
        JMP     L_0138
L_08EB:                                 ; xref 9F02:08E6
        CALL    S_09D1
L_08EE:                                 ; xref 9F02:092F
        MOV     DX,CS:D_005B            ; (=0)
        MOV     AX,CS:D_0047            ; (=0)
        OR      AX,CS:D_0037            ; (=0)
        OR      AX,8
        OUT     DX,AL; port 0, DMA-1 bas&add ch 0
        NOP
        NOP
        JMP     SHORT L_0905
        DB      90H
L_0905:                         ; xref 9F02:0902, 0909
        IN      AL,DX           ; port 0, DMA-1 bas&add ch 0
        AND     AX,8
        JZ      L_0905                  ; Jump if zero
        CALL    S_03B2
```

```
        CMP     BX,0
        JE      L_0918                  ; Jump if equal
        MOV     CS:D_003B,BX            ; (=0)
L_0918:                                 ; xref 9F02:0911
        LDS     DI,DWORD PTR CS:D_0077
                                        ; (=0) Load 32 bit ptr
        MOV     [DI],AX
        INC     DI
        INC     DI
        PUSH    CS
        POP     DS
        CMP     DI,CS:D_003F            ; (=0)
        JA      L_0931                  ; Jump if above
        MOV     CS:D_0077,DI            ; (=0)
        JMP     SHORT L_08EE
L_0931:                                 ; xref 9F02:0928
        JMP     L_0138

;
;       SUBROUTINE
;
;       Called from: 9F02:08E3, 0C3F
;
S_0934  PROC    NEAR
        MOV     CS:D_003B,0DH           ; (=0)
        CMP     CS:D_0061,7             ; (=0)
        JBE     L_0946                  ; Jump if below or =
L_0943:                 ; xref 9F02:095D, 096A, 0981, 0990
        JMP     L_09CF
L_0946:                                 ; xref 9F02:0941
        INC     CS:D_003B               ; (=0)
        MOV     AX,CS:D_0065            ; (=0)
        CMP     CS:D_005F,1             ; (=0)
        JE      L_095A                  ; Jump if equal
        ADD     AX,800H
L_095A:                                 ; xref 9F02:0955
        CMP     AX,1000H
        JAE     L_0943                  ; Jump if above or =
        MOV     CS:D_0065,AX            ; (=0)
        MOV     AX,CS:D_006D            ; (=0)
        CMP     AX,800H
        JAE     L_0943                  ; Jump if above or =
        TEST    CS:D_006B,1             ; (=0)
        JNZ     L_0998                  ; Jump if not zero
        MOV     AX,CS:D_0065            ; (=0)
        ADD     AX,CS:D_006D            ; (=0)
        CMP     AX,1000H
        JAE     L_0943                  ; Jump if above or =
        MOV     CS:D_0069,AX            ; (=0)
        MOV     AX,CS:D_0065            ; (=0)
        SUB     AX,CS:D_006D            ; (=0)
        JC      L_0943                  ; Jump if carry Set
        MOV     CS:D_0067,AX            ; (=0)
        JMP     SHORT L_09B9
L_0998:                                 ; xref 9F02:0973
        MOV     AX,CS:D_0065            ; (=0)
        ADD     AX,CS:D_006D            ; (=0)
        CMP     AX,1000H
        JAE     L_09CF                  ; Jump if above or =
        MOV     CS:D_0067,AX            ; (=0)
        MOV     AX,CS:D_0065            ; (=0)
        SUB     AX,CS:D_006D            ; (=0)
        JC      L_09CF                  ; Jump if carry Set
        MOV     CS:D_0069,AX            ; (=0)
L_09B9:                                 ; xref 9F02:0996
        INC     CS:D_003B               ; (=0)
        CMP     CS:D_006B,1             ; (=0)
        JA      L_09CF                  ; Jump if above
        MOV     CS:D_003B,0             ; (=0)
```

```
                CLC                                 ; Clear carry flag
                RETN
L_09CF:                                 ; xref 9F02:0943, 09A4, 09B3, 09C4
                STC                                 ; Set carry flag
                RETN
S_0934          ENDP
;
;               SUBROUTINE
;
;               Called from: 9F02:08EB, 0C5B
;
S_09D1          PROC    NEAR
                PUSH    CS:D_005F                   ; (=0)
                PUSH    CS:D_0039                   ; (=0)
                PUSH    CS:D_0047                   ; (=0)
                PUSH    CS:D_0059                   ; (=7)
                PUSH    CS:D_0057                   ; (=0)
                MOV     CS:D_005F,1                 ; (=0)
                MOV     CS:D_0039,0                 ; (=0)
                MOV     CS:D_0063,0                 ; (=0)
                MOV     AX,CS:D_0061                ; (=0)
                MOV     CS:D_0047,AX                ; (=0)
                MOV     CS:D_0059,AX                ; (=7)
                MOV     CS:D_0057,AX                ; (=0)
                CALL    S_026E
L_0A12:                                             ; xref 9F02:0A79
                MOV     AH,1
                INT     16H             ; Keyboard i/o  ah=function 01h
                                        ; get status, if zf=0  al=char
                JZ      L_0A1C                      ; Jump if zero
                CMP     AL,1BH
                JE      L_0A7B                      ; Jump if equal
L_0A1C:                                             ; xref 9F02:0A16
                CALL    S_03B2
                TEST    CS:D_006B,1                 ; (=0)
                JNZ     L_0A4E                      ; Jump if not zero
                CMP     CS:D_0063,1                 ; (=0)
                JE      L_0A39                      ; Jump if equal
                CMP     AX,CS:D_0067                ; (=0)
                JBE     L_0A74                      ; Jump if below or =
                JMP     SHORT L_0A79
L_0A39:                                             ; xref 9F02:0A2E
                DEC     CS:D_0063                   ; (=0)
                CMP     AX,CS:D_0069                ; (=0)
                JAE     L_0A7B                      ; Jump if above or =
                CMP     AX,CS:D_0067                ; (=0)
                JB      L_0A79                      ; Jump if below
                JMP     SHORT L_0A74
L_0A4E:                                             ; xref 9F02:0A26
                CMP     CS:D_0063,1                 ; (=0)
                JE      L_0A5F                      Jump if equal
                CMP     AX,CS:D_0067                ; (=0)
                JAE     L_0A74                      ; Jump if above or =
                JMP     SHORT L_0A79
L_0A5F:                                             ; xref 9F02:0A54
                DEC     CS:D_0063                   ; (=0)
                CMP     AX,CS:D_0069                ; (=0)
                JBE     L_0A7B                      ; Jump if below or =
                CMP     AX,CS:D_0067                ; (=0)
                JA      L_0A79                      ; Jump if above
                JMP     SHORT L_0A74
L_0A74:         ; xref 9F02:0A35, 0A4C, 0A5B, 0A72
                INC     CS:D_0063                   ; (=0)
L_0A79:         ; xref 9F02:0A37, 0A4A, 0A5D, 0A70
                JMP     SHORT L_0A12
L_0A7B:                                 ; xref 9F02:0A1A, 0A43, 0A69
                POP     CS:D_0057                   ; (=0)
                POP     CS:D_0059                   ; (=7)
                POP     CS:D_0047                   ; (=0)
                POP     CS:D_0039                   ; (=0)
                POP     CS:D_005F                   ; (=0)
                CALL    S_026E
                RETN
S_09D1          ENDP

;               Indexed Entry Point

L_0A98:                                 ; xref 9F02:00B3, 0134
                MOV     CS:D_003B,11H               ; (=0)
                CMP     CS:D_0049,1                 ; (=0)
                JNE     L_0AFA                      ; Jump if not equal
                DEC     CS:D_003B                   ; (=0)
                CMP     CS:D_002B,0                 ; (=0)
                JE      L_0AC4                      ; Jump if equal
                CMP     CS:D_002B,8                 ; (=0)
                JL      L_0AFA                      ; Jump if
                CMP     CS:D_002B,0FH               ; (=0)
                JG      L_0AFA                      ; Jump if
L_0AC4:                                             ; xref 9F02:0AB2
                MOV     DX,WORD PTR CS:D_0021 ; (=0)
                ADD     DX,3
                MOV     AX,CS:D_002B                ; (=0)
                OUT     DX,AL           ; port 3, DMA-1 bas&cnt ch 1
                MOV     CS:D_003B,0                 ; (=0)
                MOV     CS:D_005F,0                 ; (=0)
                CMP     AL,0
                JE      L_0AFA                      ; Jump if equal
                CMP     AL,8
                JE      L_0AFA                      ; Jump if equal
                CMP     AL,0AH
                JE      L_0AFA                      ; Jump if equal
                CMP     AL,0CH
                JE      L_0AFA                      ; Jump if equal
                CMP     AL,0EH
                JE      L_0AFA                      ; Jump if equal
                MOV     CS:D_005F,1                 ; (=0)
L_0AFA:         ; xref 9F02:0AA5, 0ABA, 0AC2, 0AE1
                ; 0AE5, 0AE9, 0AED, 0AF1
                JMP     L_0138

;               Indexed Entry Point

L_0AFD:                                 ; xref 9F02:00B5, 0134
                MOV     AX,CS:D_004B                ; (=0)
                MOV     CS:D_002B,0                 ; (=0)
                CMP     AX,8
                JNE     L_0B12                      ; Jump if not equal
                INC     CS:D_002B                   ; (=0)
L_0B12:                                             ; xref 9F02:0B0B
                MOV     AX,WORD PTR CS:D_007B ; (=0)
                SUB     AX,CS:D_007F                ; (=0)
                SHR     AX,1                        ; Shift w/zeros fill
                MOV     CS:D_002D,AX                ; (=0)
                MOV     DX,CS:D_005B                ; (=0)
                IN      AL,DX           ; port 0, DMA-1 bas&add ch 0
                AND     AX,7
                MOV     CS:D_0047,AX                ; (=0)
                MOV     CS:D_002F,AX                ; (=0)
                MOV     CS:D_0031,0FFFFH            ; (=0)
                CMP     CS:D_0049,1                 ; (=0)
                JNE     L_0B4D                      ; Jump if not equal
                INC     DX
                JMP     SHORT $+3                   ; delay for I/O
                NOP
                IN      AL,DX           ; port 1, DMA-1 bas&cnt ch 0
                AND     AX,0FH
```

```
         MOV     CS:D_0031,AX           ; (=0)
L_0B4D:                                 ; xref 9F02:0B3F
         MOV     CS:D_003B,0            ; (=0)
         JMP     L_0138
D_0B57   DW      0 ; xref 9F02:0B6F, 0B86, 0C05, 0C6B
                                        ; 0C8F
D_0B59   DW      OFFSET L_0B86 ; Data table (indexed access)
                                        ; xref 9F02:0B7D
D_0B5B   DW      OFFSET L_0C14          ; xref 9F02:0B7D
D_0B5D   DW      OFFSET L_0C8A          ; xref 9F02:0B7D

;        Indexed Entry Point

L_0B5F:                        ; xref 9F02:00B7, 0134
         PUSH    CS
         POP     DS
         MOV     D_00BB,DS              ; (=9F02H)
         MOV     D_003B,13H             ; (=0)
         MOV     BX,D_002B              ; (=0)
         CMP     BX,D_0B57              ; (=0)
         JA      L_0B81                 ; Jump if above
         MOV     D_003B,0               ; (=0)
         SHL     BX,1                   ; Shift w/zeros fill
         JMP     WORD PTR D_0B59[BX] ;*(=0B86H) 3 entries
L_0B81:          ; xref 9F02:0B73, 0C11, 0C77, 0D48
         JMP     L_0138
D_0B84   DW      0 ; xref 9F02:0D4B, 0D6B, 0D81, 0D91

;        Indexed Entry Point

L_0B86:                        ; xref 9F02:0B59, 0B7D
         MOV     D_0B57,0               ; (=0)
         MOV     AX,D_002D              ; (=0)
         MOV     D_0089,AX              ; (=0)
         MOV     D_0C7A,AX              ; (=0)
         MOV     AX,D_0023              ; (=9EF2H)
         MOV     D_008B,AX              ; (=0)
         MOV     WORD PTR D_0C7A+2,AX ; (=0)
         MOV     AX,D_002F              ; (=0)
         MOV     WORD PTR D_0085,AX     ; (=0)
         MOV     D_0C7E,AX              ; (=0)
         MOV     D_0C80,AX              ; (=0)
         OR      CX,CX                  ; Zero ?
         MOV     AX,D_0057              ; (=0)
         OR      CL,AL
         OR      CL,8
         MOV     AX,D_0037              ; (=0)
         OR      CL,AL
         MOV     AX,D_0059              ; (=7)
         SUB     AX,D_0057              ; (=0)
         OR      CH,AL
         MOV     D_0083,CX              ; (=0)
         INC     AX
         MOV     D_0C82,AX              ; (=0)
         MOV     D_0C84,AX              ; (=0)
         MOV     D_0C86,0               ; (=0)
         CMP     D_005F,1               ; (=0)
         JNE     L_0BE0                 ; Jump if not equal
         OR      D_0C86,1               ; (=0)
         NOP                 ;*ASM fixup - sign extn byte
L_0BE0:                                 ; xref 9F02:0BD8
         CMP     D_0039,1               ; (=0)
         JNE     L_0BFD                 ; Jump if not equal
         OR      D_0C86,2               ; (=0)
         NOP                 ;*ASM fixup - sign extn byte
         MOV     AX,D_0057              ; (=0)
         SHL     AX,1                   ; Shift w/zeros fill
         SHL     AX,1                   ; Shift w/zeros fill
         SHL     AX,1                   ; Shift w/zeros fill
         SHL     AX,1                   ; Shift w/zeros fill
         MOV     AH,AL
         MOV     D_0C88,AX              ; (=0)
L_0BFD:                                 ; xref 9F02:0BE5
         MOV     AX,D_0031              ; (=0)
         CALL    S_0D4B
         JC      L_0C11                 ; Jump if carry Set
         MOV     D_0B57,1               ; (=0)
         MOV     D_003B,0               ; (=0)
L_0C11:                                 ; xref 9F02:0C03
         JMP     L_0B81

;        Indexed Entry Point

L_0C14:                        ; xref 9F02:0B5B, 0B7D
         CMP     D_002D,0               ; (=0)
         JE      L_0C44                 ; Jump if equal
         PUSH    D_0033                 ; (=0)
         PUSH    D_002D                 ; (=0)
         PUSH    D_0031                 ; (=0)
         PUSH    D_002F                 ; (=0)
         POP     D_0061                 ; (=0)
         POP     D_0065                 ; (=0)
         POP     D_006B                 ; (=0)
         POP     D_006D                 ; (=0)
         DEC     D_006B                 ; (=0)
         CALL    S_0934
         JC      L_0C77                 ; Jump if carry Set
L_0C44:                                 ; xref 9F02:0C19
         CMP     D_0035,0               ; (=0)
         JE      L_0C54                 ; Jump if equal
L_0C4B:                                 ; xref 9F02:0C52
         MOV     DX,D_005B              ; (=0)
         IN      AL,DX          ; port 0, DMA-1 bas&add ch 0
         TEST    AL,20H                 ; ' '
         JZ      L_0C4B                 ; Jump if zero
L_0C54:                                 ; xref 9F02:0C49
         CMP     D_002D,0               ; (=0)
         JE      L_0C5E                 ; Jump if equal
         CALL    S_09D1
L_0C5E:                                 ; xref 9F02:0C59
         MOV     D_031B,0               ; (=0)
         MOV     AL,8
         MOV     DX,D_005B              ; (=0)
         OUT     DX,AL          ; port 0, DMA-1 bas&add ch 0
         MOV     D_0B57,2               ; (=0)
         MOV     D_003B,0               ; (=0)
L_0C77:                                 ; xref 9F02:0C42
         JMP     L_0B81
D_0C7A   DW      0, 0 ; xref 9F02:0B92, 0CAB, 0CCA, 0D0B
D_0C7E   DW      0 ; xref 9F02:0BA4, 0CB2, 0CCE, 0D0F, 0D32
D_0C80   DW      0 ; xref 9F02:0BA7, 0CA0, 0D2F
D_0C82   DW      0 ; xref 9F02:0BC7, 0D15, 0D1F
D_0C84   DW      0 ; xref 9F02:0BCA, 0D1B
D_0C86   DW      0 ; xref 9F02:0BCD, 0BDA, 0BE7, 0CE2, 0CED
D_0C88   DW      0 ; xref 9F02:0BFA, 0CF5, 0D01, 0D23, 0D29

;        Indexed Entry Point

L_0C8A:                        ; xref 9F02:0B5D, 0B7D
         PUSH    AX
         PUSH    BX
         PUSH    CX
         PUSH    DX
         PUSH    ES
         CMP     D_0B57,2               ; (=0)
         JNE     L_0CB9                 ; Jump if not equal
```

```
        CMP     D_002D,0                ; (=0)
        JE      L_0CB2                  ; Jump if equal
        CALL    S_0D81
        MOV     AX,D_0C80               ; (=0)
        SUB     AX,WORD PTR D_0085      ; (=0)
        JZ      L_0CB9                  ; Jump if zero
        MOV     CX,AX
        LES     BX,DWORD PTR D_0C7A; (=0) Load 32 bit ptr
        JMP     SHORT L_0CCE
        DB      90H
L_0CB2:                                 ; xref 9F02:0C9B
        CMP     D_0C7E,0                ; (=0)
        JNE     L_0CBC                  ; Jump if not equal
L_0CB9:                 ; xref 9F02:0C94, 0CA7, 0CD5
        JMP     SHORT L_0D2F
        DB      90H
L_0CBC:                                 ; xref 9F02:0CB7
        CMP     WORD PTR D_0085,0       ; (=0)
        JNE     L_0CC6                  ; Jump if not equal
        CALL    S_0D81
L_0CC6:                                 ; xref 9F02:0CC1
        MOV     CX,D_002F               ; (=0)
        LES     BX,DWORD PTR D_0C7A; (=0) Load 32 bit ptr
L_0CCE:                 ; xref 9F02:0CAF, 0D2D
        MOV     AX,D_0C7E               ; (=0)
        CMP     AX,WORD PTR D_0085      ; (=0)
        JBE     L_0CB9                  ; Jump if below or =
        MOV     AX,ES:[BX]
        SHR     AX,1                    ; Shift w/zeros fill
        SHR     AX,1                    ; Shift w/zeros fill
        SHR     AX,1                    ; Shift w/zeros fill
        SHR     AX,1                    ; Shift w/zeros fill
        TEST    D_0C86,1                ; (=0)
        JNZ     L_0CED                  ; Jump if not zero
        SUB     AX,800H
L_0CED:                                 ; xref 9F02:0CE8
        TEST    D_0C86,2                ; (=0)
        JZ      L_0D05                  ; Jump if zero
        MOV     DX,D_0C88               ; (=0)
        AND     AH,8FH
        OR      AH,DH
        ADD     DH,10H
        MOV     D_0C88,DX               ; (=0)
L_0D05:                                 ; xref 9F02:0CF3
        MOV     ES:[BX],AX
        ADD     BX,2
        MOV     D_0C7A,BX               ; (=0)
        DEC     D_0C7E                  ; (=0)
        JZ      L_0D2F                  ; Jump if zero
        DEC     D_0C82                  ; (=0)
        JNZ     L_0D2D                  ; Jump if not zero
        PUSH    D_0C84                  ; (=0)
        POP     D_0C82                  ; (=0)
        MOV     DX,D_0C88               ; (=0)
        MOV     DH,DL
        MOV     D_0C88,DX               ; (=0)
L_0D2D:                                 ; xref 9F02:0D19
        LOOP    L_0CCE                  ; Loop if cx 0
L_0D2F:                 ; xref 9F02:0CB9, 0D13
        MOV     AX,D_0C80               ; (=0)
        SUB     AX,D_0C7E               ; (=0)
        MOV     D_0031,AX               ; (=0)
        CMP     WORD PTR D_0085,0       ; (=0)
        JNE     L_0D43                  ; Jump if not equal
        CALL    S_0D81
L_0D43:                                 ; xref 9F02:0D3E
        POP     ES
        POP     DX
```

```
        POP     CX
        POP     BX
        POP     AX
        JMP     L_0B81
;
;                SUBROUTINE
;
;       Called from: 9F02:0C00
;
S_0D4B  PROC    NEAR
        CMP     D_0B84,1                ; (=0)
        JE      L_0D71                  ; Jump if equal
        CMP     AX,2
        JB      L_0D79                  ; Jump if below
        CMP     AX,7
        JA      L_0D79                  ; Jump if above
        MOV     D_031B,2                ; (=0)
        ADD     AX,8
        MOV     BX,31DH
        CALL    S_02AC
        MOV     D_0B84,1                ; (=0)
L_0D71:                                 ; xref 9F02:0D50
        MOV     D_003B,0                ; (=0)
        CLC                             ; Clear carry flag
        RETN
L_0D79:                 ; xref 9F02:0D55, 0D5A
        MOV     D_003B,7                ; (=0)
        STC                             ; Set carry flag
        RETN
S_0D4B  ENDP
;
;                SUBROUTINE
;
;       Called from: 9F02:0C9D, 0CC3, 0D40
;
S_0D81          PROC    NEAR
                CMP     D_0B84,0        ; (=0)
                JE      L_0D97          ; Jump if equal
                MOV     D_031B,2        ; (=0)
                CALL    S_02DF
                MOV     D_0B84,0        ; (=0)
L_0D97:                                 ; xref 9F02:0D86
        RETN
S_0D81  ENDP
D_0D98  DW      0       ; xref 9F02:0D9A, 0DB0, 0E7E

;               Indexed Entry Point

L_0D9A:                 ; xref 9F02:00B9, 0134
        CMP     D_0D98,0                ; (=0)
        JE      L_0DB9                  ; Jump if equal
        MOV     D_004B,0                ; (=0)
        MOV     D_0055,0                ; (=0)
        CALL    S_02DF
        MOV     D_0D98,0                ; (=0)
L_0DB6:         ; xref 9F02:0DDB, 0DE8, 0DF6, 0DFC, 0E03
        JMP     L_0E95
L_0DB9:                                 ; xref 9F02:0D9F
        MOV     D_003B,8                ; (=0)
        MOV     WORD PTR D_007B,0       ; (=0)
        PUSH    D_002D                  ; (=0)
        POP     WORD PTR D_007B+2       ; (=0A000H)
        PUSH    WORD PTR D_007B         ; (=0)
        POP     D_007F                  ; (=0)
        MOV     AX,D_002F               ; (=0)
        CMP     AX,1
```

```
         JL      L_0DB6                 ; Jump if
         SHL     AX,1                   ; Shift w/zeros fill
         MOV     D_0043,AX              ; (=0)
         MOV     AX,WORD PTR D_007B+2   ; (=0A000H)
         CMP     AX,0
         JL      L_0DB6                 ; Jump if
         MOV     D_003B,14H             ; (=0)
         MOV     AX,D_0031              ; (=0)
         CMP     AX,0FH
         JA      L_0DB6                 ; Jump if above
         CMP     AX,D_0033              ; (=0)
         JA      L_0DB6                 ; Jump if above
         CMP     D_0033,0FH             ; (=0)
         JA      L_0DB6                 ; Jump if above
         SHL     AX,1                   ; Shift w/zeros fill
         SHL     AX,1                   ; Shift w/zeros fill
         SHL     AX,1                   ; Shift w/zeros fill
         SHL     AX,1                   ; Shift w/zeros fill
         MOV     D_0E9C,AX              ; (=0)
         MOV     D_0E9A,AX              ; (=0)
         MOV     AX,D_0033              ; (=0)
         SHL     AX,1                   ; Shift w/zeros fill
         SHL     AX,1                   ; Shift w/zeros fill
         SHL     AX,1                   ; Shift w/zeros fill
         SHL     AX,1                   ; Shift w/zeros fill
         MOV     D_0E9E,AX              ; (=0)
         MOV     AX,D_0035              ; (=0)
         MOV     D_0EA4,AX              ; (=0)
         MOV     D_003B,7               ; (=0)
         MOV     AX,D_002B              ; (=0)
         AND     AX,7
         CMP     AX,2
         JB      L_0E95                 ; Jump if below
         MOV     D_0055,0               ; (=0)
         MOV     AX,D_002B              ; (=0)
         AND     AX,8
         JZ      L_0E4A                 ; Jump if zero
         INC     D_0055                 ; (=0)
L_0E4A:                                 ; xref 9F02:0E44
         MOV     AX,D_0059              ; (=7)
         MOV     WORD PTR D_0E98,AX     ; (=0)
         MOV     D_0047,AX              ; (=0)
         MOV     AX,D_005F              ; (=0)
         MOV     D_0EA8,AX              ; (=0)
         MOV     AX,D_0039              ; (=0)
         MOV     D_0EAA,AX              ; (=0)
         MOV     DX,D_005B              ; (=0)
         MOV     AX,D_0057              ; (=0)
         OR      AX,D_0E9A              ; (=0)
         OUT     DX,AL          ; port 0, DMA-1 bas&add ch 0
         PUSH    AX
         PUSH    DX
         MOV     AX,D_002B              ; (=0)
         AND     AX,7
         ADD     AX,8
         MOV     BX,0EACH
         CALL    S_02AC
         POP     DX
         POP     AX
         MOV     D_0D98,1               ; (=0)
         MOV     D_003B,0               ; (=0)
         MOV     D_004B,8               ; (=0)
         OR      AX,D_004B              ; (=0)
         OUT     DX,AL          ; port 0, DMA-1 bas&add ch 0
L_0E95:                         ; xref 9F02:0DB6, 0E36
         JMP     L_0138
D_0E98   DB      0, 0 ; xref 9F02:0E4D, 0F79
D_0E9A   DW      0 ; xref 9F02:0E10, 0E66, 0F7F, 0F8B, 0FA2
```

```
D_0E9C   DW      0 ; xref 9F02:0E0D, 0F98
D_0E9E   DW      0 ; xref 9F02:0E1E, 0F92
D_0EA0   DW      0 ; xref 9F02:0EF3, 0F11, 0F21
D_0EA2   DW      0 ; xref 9F02:0EF9, 0F17, 0F24
D_0EA4   DW      0 ; xref 9F02:0E24, 0EE6, 0EED, 0F0A, 0F28
D_0EA6   DW      0 ; xref 9F02:0EF0, 0F1B
D_0EA8   DW      0 ; xref 9F02:0E56, 0F2C
D_0EAA   DW      0 ; xref 9F02:0E5C, 0F36
         DB      50H, 53H, 51H, 52H, 1EH, 57H
         DB      0EH, 1FH,0B0H, 20H,0E6H, 20H
         DB      0FFH, 36H, 37H, 00H,0FFH, 36H
         DB      5FH, 00H,0FFH, 36H, 39H, 00H
         DB      83H, 3EH, 4BH, 00H, 08H, 74H
         DB      03H,0E9H,0E3H, 00H,0C7H, 06H
         DB      5FH, 00H, 01H, 00H,0C7H, 06H
         DB      39H, 00H, 00H, 00H,0A1H, 9AH
         DB      0EH,0A3H, 37H, 00H,0C7H, 06H
         DB      4BH, 00H, 00H, 00H
L_0EE6:                                 ; xref 9F02:0F88
         CMP     D_0EA4,1               ; (=0)
         JBE     L_0EFF                 ; Jump if below or =
         MOV     AX,D_0EA4              ; (=0)
         MOV     D_0EA6,AX              ; (=0)
         MOV     D_0EA0,0               ; (=0)
         MOV     D_0EA2,0               ; (=0)
L_0EFF:                         ; xref 9F02:0EEB, 0F1F
         PUSH    D_0047                 ; (=0)
         CALL    S_03B2
         POP     D_0047                 ; (=0)
         CMP     D_0EA4,1               ; (=0)
         JBE     L_0F2C                 ; Jump if below or =
         ADD     D_0EA0,AX              ; (=0)
         JNC     L_0F1B                 ; Jump if carry=0
         INC     D_0EA2                 ; (=0)
L_0F1B:                                 ; xref 9F02:0F15
         DEC     D_0EA6                 ; (=0)
         JNZ     L_0EFF                 ; Jump if not zero
         MOV     AX,D_0EA0              ; (=0)
         MOV     DX,D_0EA2              ; (=0)
         IDIV    D_0EA4         ; (=0) ax,dxrem=dx:ax/data
L_0F2C:                                 ; xref 9F02:0F0F
         CMP     D_0EA8,1               ; (=0)
         JE      L_0F36                 ; Jump if equal
         SUB     AX,800H
L_0F36:                                 ; xref 9F02:0F31
         CMP     D_0EAA,1               ; (=0)
         JNE     L_0F46                 ; Jump if not equal
         MOV     BX,D_0037              ; (=0)
         AND     AH,8FH
         OR      AH,BL
L_0F46:                                 ; xref 9F02:0F3B
         PUSH    DS
         LDS     DI,D_007B      ; (=0) Load 32 bit ptr
         MOV     [DI],AX
         INC     DI
         INC     DI
         POP     DS
         MOV     WORD PTR D_007B,DI     ; (=0)
         CMP     DI,D_0043              ; (=0)
         JB      L_0F69                 ; Jump if below
         CMP     D_0055,0               ; (=0)
         JE      L_0FB1                 ; Jump if equal
         PUSH    D_007F                 ; (=0)
         POP     WORD PTR D_007B        ; (=0)
L_0F69:                                 ; xref 9F02:0F58
         MOV     AX,D_0047              ; (=0)
         INC     AX
         CMP     AX,D_0059              ; (=7)
```

```
          JBE     L_0F76                 ; Jump if below or =
          MOV     AX,D_0057              ; (=0)
L_0F76:                                  ; xref 9F02:0F71
          MOV     D_0047,AX              ; (=0)
          CMP     AX,WORD PTR D_0E98     ; (=0)
          JE      L_0F8B                 ; Jump if equal
          OR      AX,D_0E9A              ; (=0)
          MOV     DX,D_005B              ; (=0)
          OUT     DX,AL        ; port 0, DMA-1 bas&add ch 0
          JMP     L_0EE6
L_0F8B:                                  ; xref 9F02:0F7D
          MOV     BX,D_0E9A              ; (=0)
          ADD     BX,10H
          CMP     BX,D_0E9E              ; (=0)
          JBE     L_0F9C                 ; Jump if below or =
          MOV     BX,D_0E9C              ; (=0)
L_0F9C:                                  ; xref 9F02:0F96
          MOV     D_004B,8               ; (=0)
          MOV     D_0E9A,BX              ; (=0)
          OR      AX,BX
          OR      AX,D_004B              ; (=0)
          MOV     DX,D_005B              ; (=0)
          OUT     DX,AL        ; port 0, DMA-1 bas&add ch 0
L_0FB1:                                  ; xref 9F02:0F5F
          POP     D_0039                 ; (=0)
          POP     D_005F                 ; (=0)
          POP     D_0037                 ; (=0)
          POP     DI
          POP     DS
          POP     DX
          POP     CX
          POP     BX
          POP     AX
          IRET                           ; Interrupt return

SEG_A     ENDS
          END     START
```

CROSS REFERENCE - KEY ENTRY POINTS

seg:off	type	label
9F02:0000	FAR	START

Interrupt Usage Synopsis
Interrupt 16H : get status, if zf = 0 al = char

I/O	Port Usage Synopsis
Port 0	: DMA-1 bas&add ch 0
Port 1	: DMA-1 bas&cnt ch 0
Port 2	: DMA-1 bas&add ch 1
Port 3	: DMA-1 bas&cnt ch 1
Port 4	: DMA-1 bas&add ch 2
Port 5	: DMA-1 bas&cnt ch 2
Port 6	: DMA-1 bas&add ch 3
Port 7	: DMA-1 bas&cnt ch 3
Port 20H	: 8259-1 int command
Port 21H	: 8259-1 int IMR
Port 21H	: 8259-1 int comands

2 Ocurrences of Non-Standard I/O ports used.

REFERENCES

Crussel Mason, "The Art and Science of Protective Relaying", Engineering Planning and Development Section General Electric Company Schenectady, New York, 1979.

David J. Comer, "Modern Electronic Circuit Design", California State University, 1981.

General Electric, "Silicon Controlled Rectifier Manual including; Triac and the Industry's Broadest Line of Thyristor and Rectifier Components".

Irving M. Gottlieb, "Solid State Power Electronic", McGraw Hill Publisher, 1984

Jacob Millman, Ph D and Christos C. Halkias, Ph D, "Electronic Devices and Circuit Design", Columbia University, 1986.

Madava Rao, "Power System Protection Static Relays", Department of Electrical Engineering, University of Roorkee, India, 1979.

Richard F. Shea, "Principles of Transistor circuits", Electronic Laboratory of General Electric Company, Electronics Park Syracuse, New York, 1986.

Sunil S. Rao, "Switchgear and Protection", McGraw Hill Publisher, 1983.

The Electricity Council, "Power System Protection 2", Peter Peregrinus LTD, 1986.

Abraham I. Pressman, "Design Solid State Circuit for Digital computers", John F Rider Publisher, Inc., New York, 1980.

Robert F. Coughlin and Frederick F. Driscoll, "Operational Amplifier and Linear Integrated Circuits", Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1987.

Ralph J. Smith, "Circuit, Devices and Systems", John Willey & Son, Inc, 1971.

Bharat Kinariwala, Franklin F. Kuo and Nai-Kuan Tsao, "Circuit and Computation", John Willey & Son, Inc, 1978.

John Staudhammer, "Circuit Analysis by Digital Computer", Prentice Hall, Inc., Englewood Cliff, New Jersey, 1976.

Spectrum, Inc., "MICRO-CAP", Advanced Microcomputer Circuit Analysis, 1982 - 1988.

Alen L. Wyatt, "Assembly Language", Que Corp., Carmel, Indiana, 1987.

"Macro Assembler", Professional Development System, Microsoft, 1991.

"Advance BASIC", I B M, 1988.

"Professional BASIC compiler", Microsoft, 1991.

"Quick BASIC", Microsoft, 1989.

"Turbo BASIC", Borland International, 1988.

"Object Compiler", Microsoft and IBM, 1991.