National Institute of Electricity and Electronics

INELEC - Boumerdes

DEPARTMENT OF RESEARCH

THESIS

Presented in partial fulfilment of the requirements of the

DEGREE OF MAGISTER

in Applied Electronics

by

Kamal MEGHRICHE

A Microprocessor-Based Multiplexed System for Automotive Industry

Defended on april 21, 1991 before the jury :

President : *Dr. TEDJINI BAILICHE Hacène, Maître de conférence, U.S.T.H.B*

Members : *Dr. AL-LAMI Bassim, Maître de conférences, I.N.E.L.E.C.*
Dr. BOURDOUCEN Hadj, I.N.E.L.E.C.
Dr. HARICHE Kamel, I.N.E.L.E.C.
Dr. DIAF Moussa, Chargé de recherche, Université de Tizi-Ouzou

Invited Member : *Dr. MORT Neil, Control Engineering Dept., University of Sheffield.*

Registration Number : 01/91

oOo DEDICATION oOo

To my mother

To the memory of my father

To my nephews Redouane and Jalal

To all my family

I dedicate this work.

ACKNOWLEDGMENT

I would like to express my great debt to my supervisor and teacher, Dr. B. J. AL-LAMI, for his continuous help, suggestions and supervision during the whole period of this project development.

A special note of appreciation is due to the Postgraduate Studies Officer, Mr. A. MOUFEK, for his unlimited support to the final success of our work.

My sincere gratitude is due to Dr. TEDJINI BAILICHE Hacene (U.S.T.H.B.) for his acceptance to be the President of the Jury.

My thanks are due to Dr. K. HARICHE (INELEC) and Dr. M. DIAF (University of Tizi-Ouzou) for their kind acceptance to be members of the jury.

I wish to express my great debt to Dr. H. BOURDOUCEN for his detailed proofreading of the manuscript, his fruitfull suggestions and his acceptance to be a member of the jury.

My thanks are due to Dr. M. O. TOKHI from the department of control engineering, University of Sheffield, for his acceptance to be a guest member of the jury.

My gratitude is due to Mr. A. BENAZZOUZ, Director General, Mr. A. HAMDI, Dean of Studies and Pr. V. MITRA for their continuous interest to the project.

I would like to thank Dr. S. BANKS, Dr. A. MORRIS from the department of control engineering (University of Sheffield), as well as Dr. B. CHANANE for their assistance and interest during my training in the University of Sheffield, England.

This project could not be achieved without the contribution of the National Company of Industrial Vehicles (SNVI-CVI), namely, Mr AMAZIT, Mr D. MAHIOUT from the 'Unité Etudes et Recherches' department and all the team of (U.E.R.) technicians.

Coming back to INELEC, I would like to acknowledge the support and help provided by Mr. B. OUHAB and Mr. D. BENAZZOUZ.

The encouragements of Mr. Y. HAMADA, Mr. A. BOUKLACHI, Mr. Z. SARI, and all other INELEC faculty were welcomed. The help provided by Mrs F. ZERARI and all the library staff is greatly appreciated.

I could not close this acknowledgment without expressing my sincere gratitude to two persons to whom I owe a great deal. The sacrifice, moral support and efforts, made by my unique brother MOSTEFA and my fiancée Miss A. SIFI to encourage me achieve this work, were of great benefit.

My appreciation and gratitude are due to all other persons who contributed in a way or another to the final success of this project.

This thesis report describes a modern alternative solution to the wiring complexity problem by introducing multiplexing techniques which consist of replacing the conventional wiring harness by another system that highly reduces the number of conductors being used, provides monitoring tools, and exhibits high reliability and simplicity of installation and service.

The importance of the multiplex system is growing as a means to solve the various problems related to vehicle harness, and a variety of multiplexed wiring solutions have been proposed throughout the world. However, due to the highly competitive nature of the automotive market, only few things, such as system characteristics, could be known about the already or being developed prototypes. In fact, apart from some standard requirements of multiplexed systems and some solution approaches such as the Society of Automotive Engineers J1850 SAE, each car manufacturer is concealing the details of its proposed solutions thus making attempts for comparative study very difficult.

This report has been subdivided into seven chapters. Chapter I sheds some light on the early beginning of multiplexing in motor vehicles and its development. It also gives a brief discussion about the requirements and the criteria to be fulfilled by a multiplexed system. The last part describes the different classes of multiplexing as defined by the Society of Automotive Engineers.

Chapter II discusses theoritical design aspects from constraints to strategies as well as it gives a general description of the different network topologies.

A general description of the INELEC Multiplexed System (IMS) along with an extensive coverage of the system hardware (master and slave) are given in chapter III.

Chapter IV discusses the communication protocol, data integrity, reliabilty and recovery within the system. Namely classes of failures, error detection, diagnosis and recovery are presented.

Chapter V provides a description of the system interfacing and its prototype implementation based on the electrical system of a '25L4' type minibus provided by the Algerian Company of Industrial Vehicles (SNVI). Also, it gives an idea about the state of the art in the field of intelligent power switching.

Chapter VI discusses the different software routines developed within the system from the master as well as the slave control units side.

Chapter VII concludes this thesis report with a brief discussion of future technology and suggests different points for further work.

PREFACE

The increasing number of electrical and electronic units being used within a motor vehicle as well as the growing need for monitoring systems, has caused the number of cables to carry these signals and feeds to increase. Consequently, the electrical systems of most motor vehicles are becoming more complex with a resultant considerable increase in the wiring complexity and the difficulty of installation. Although these cables are tied together to form a compact harness, the bulk, and weight of the looms and connectors makes accomodation difficult. In addition, the grouping together of supply cables often causes the cables at the center of the loom to overheat; as a result of the increase in total harness resistance, the efficiency of the system is lowered.

One solution to the problem, is to use a remote switching system. This system, which has been in use during recent years, uses a common power cable and numerous signal cables to control the switching devices. Switching is normally performed using relays. The signal cable to each relay only carries a small current so the cables can be smaller, in diameter size, than those originally used. This solution still suffers wiring complexity and the total length of the cables used is still high.

ABSTRACT

There are many applications where modern control systems are becoming more and more sophisticated with a resultant increase in the complexity of wiring interconnections.

This complexity is due to the requirement of one or more conductors to connect the control panel to each specified function. These connectors are to carry power, control and monitoring signals to remote locations.

One of these applications is the electrical wiring interconnection of a motor vehicle. Although there are many different systems as there are car manufacturers, it is still a common interest for all to replace the present conventional, costly and non-intelligent system by another that highly reduces the number of conductors, increase intelligence and easy to install and service.

The aim in this study is to design and develop an electronic switching and monitoring system based on the multiplexing of digital codes by local control station and remote monitoring units. The system is to use the suitable VLSI and power devices to build a prototype that could be useful product to the Algerian Car Industry.

Keywords: Multiplexing, microprocessor L.A.N., vehicle harness.

TABLE OF CONTENT

CHAPTER ONE

MULTIPLEXED WIRING HARNESS

. Introduction

. History and development

. Requirements of a multiplexed system

. Classes of multiplexing

Multiplexed wiring harness

I.1 Introduction

The market for automotive electronics in general is only just beginning to open up, with predictions that 25% of the cost of the average car (Fig. 1.1) will be in electronics by the mid 1990s [1]. This presents the designer with quite a challenge as the vehicle environment is relatively harsh.

The wiring harness in a car has been increasing dramatically over the last decade (Plate 1) and there always seems to be yet another accessory to add. An attractive alternative to the present wiring system, which requires a separate power line to each electrical/electronic device, fed generally via the dashboard controls, is to wire each device to a common power ring. The switching of each device is activated upon receipt of a coded signal sent via a single transmission line. Single-chip microcomputers are used as local units to achieve the required functional tasks.

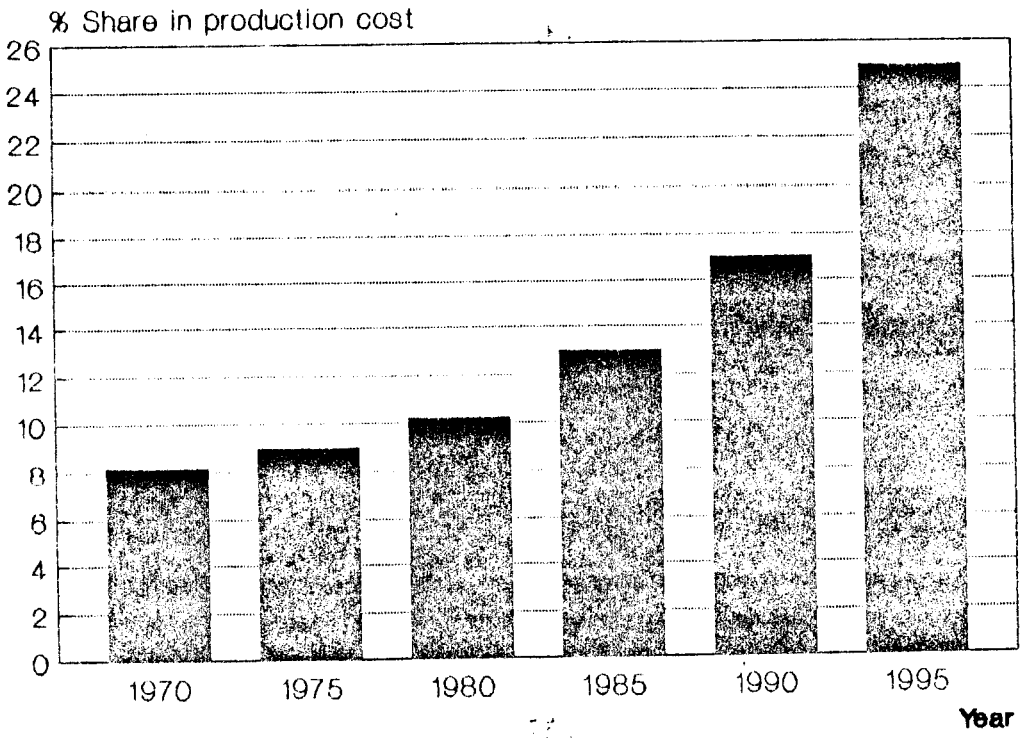


Fig. 1.1 Share of cost of electronic components in total vehicle cost.

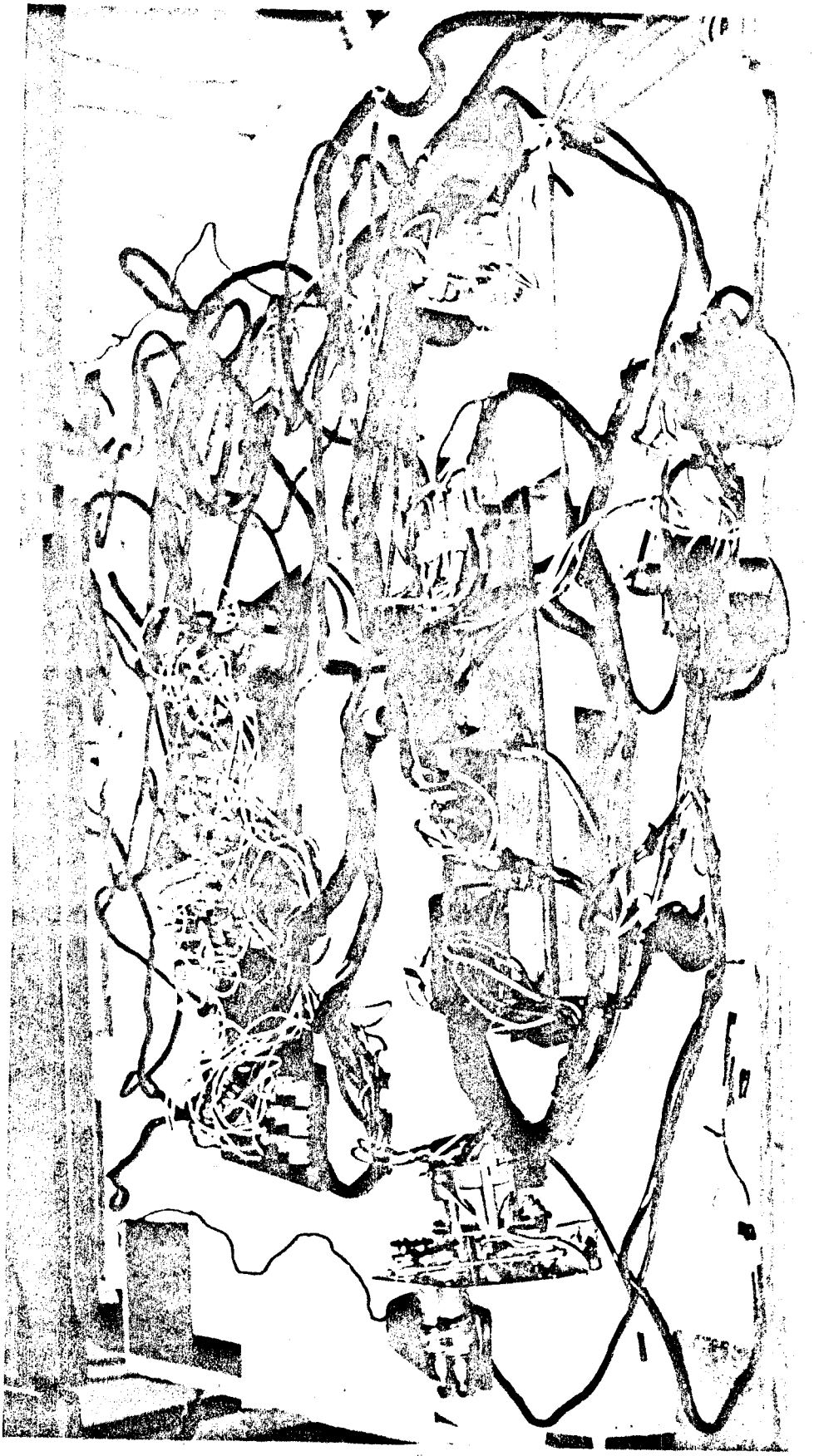


Plate 1: Typical vehicle harness.

Many technical proposals have been made which allow the replacement of today's large and complex wiring harnesses with less awkward and more rational ones. Nevertheless, as always seems to be the case with new developments, the introduction into volume production of multiplexed wiring systems is still awaited. This is mainly due to the fact that each car manufacturer is developing its own system with specific characteristics and application. The birth of a universal standard in multiplex technology does not seem to be for the near future.

I.2 History and development

Automotive multiplex wiring is rapidly developing now that the demands for more electronic option features are being felt in the marketplace. The vehicle wire harness congestion problem has been identified for many years, and the solution of time-division multiplexing over a serial data bus has been suggested many times in the past. Such example solutions are U.S. patents #3 564 280 dated 2-16-71, #3 648 057 dated 3-7-72, and #3 742 447 dated 6-26-73 by Songefest and Estes [2], which cover the pertinent details of this method as it applies to the automotive requirements.

Research and development of the automotive multiplex system was started around 1975, followed by tentative slagging of research for a certain period. In the background of growing needs and inexpensiveness and improved reliability of semiconductor devices as seeds, research and development activities have activated again and we are now in presence of a variety of multiplex systems as there are car manufacturers. Some examples of developed multiplex systems are shown in table 1.1 [3]

Car manufacturer	System	Type	Addressing	Topology	Media	Year
Mazda (Japan)	PALNET	Distributed	Broadcast	Bus	Twisted pair	89
Ford (USA)	VNP	Distributed	Physical addr.	Bus	Twisted pair	86
Bosch (Germany)	CAN	Distributed	Broadcast	Bus	Shielded twisted pair	86
Chrysler (USA)	CCD	Distributed	Broadcast	Bus	Twisted pair	86
Nissan (Japan)	CEDRIC	Distributed	Physical addr.	Bus	Optical fiber	81
Toyota (Japan)	CROWN	Centralized	Physical addr.	Bus	Twisted pair	-
GM (USA)	ALLANTE	Centralized	Physical addr.	Loop	AV	-

Table 1.1 Examples of multiplexed wiring systems.

I.3 Requirements of a multiplexed system

A list, though not exhaustive, of the basic requirements of a multiplexed wiring system is given below:

1. Cost effectiveness: this important requirement may be assisted by keeping the system as simple as possible, and the simpler the system, the easier and cheaper is its maintenance.
2. The technology used should have good immunity to noise and supply variation. CMOS is the most suitable technology for automotive application.
3. The system should provide fail safe (emergency) operation.
4. The multiplexed system should be transparent to the vehicle's driver, in other words, the driver should notice no difference between the response to controls in a multiplexed system and a non-multiplexed one.
5. Flexibility: the multiplexed system should be easily adaptable to other vehicle models.

1.4 Classes of multiplexing

The society of Automotive Engineers (SAE) [4,5] divides communication requirements in a multiplexed system into three classes: A, B, and C. Classes are defined on the basis of speed and accuracy, see table 1.2.

Table 1.2 shows that class A multiplexing refers to low speed signals. These types of signals are found in simple body control applications such as turn signals, dashboard indicators, where speed and accuracy are not of critical importance.

Class B signals are faster, and require more accuracy. They flow over a bus that is a type of local area network (LAN) within the vehicle. This type of LAN can provide diagnostic capability.

Class C multiplexing defines bus requirements for high speed signals. these types of signals are suitable for real-time applications, where speed and accuracy are of critical importance, such as anti-lock braking system (ABS), power train, engine control, and ignition and fuel injection system.

The class B network is intended to support communications that would perform all the functions of class A. In a similar way, class C network is intended to support all functions of class B network.

A new concept of multiplexing known as class D [6] has been recently introduced (1987). These types of signals are mainly encountered in shared data block systems and enhanced diagnostic tools, such as trip computer and remote detection and sensing.

CLASS	SPEED Kbit/s	LATENCY ms	MESSAGE SIZE byte	ERROR HANDLING
A	1	20 - 50	1	LOW
B	10 - 100	5 - 50	2 - 10	MEDIUM
C	1000	<5	2 - 10	HIGH
D	-	50 - 100	0.5K - 2K	-

TABLE 1.2 Functional characteristics of different classes of multiplexing

Efforts for standardization of a serial bus for in-vehicle communication, will be pursued towards the design of a "Universal" system which could be tailored with relatively minor changes to suit a variety of vehicle specifications from different manufacturers. However, at the current stage of design and development, one cannot escape some basic specifications due particularly to the design constraints and the specific requirements of car manufacturers. The following chapter will describe the constraints and the design approaches behind the development of the INELEC Multiplexed System (IMS).

CHAPTER TWO

DESIGN ISSUES

. Design constraints

. Electronic design options

. Network topologies

. Design strategy

Design Issues

II.1 Design constraints

This project has been proposed on the basis of a mutual agreement between the Department of Postgraduate Studies of INELEC and the Research and Development Department of the Algerian Company of Industrial Vehicles, (SNVI-CVI), which has expressed an interest for the advantages that may be brought by the implementation of a multiplexed wiring system in replacement of the existing wiring harness installed so far on the different vehicle models.

The manufacturer considered the system from the different sides of engineering and concluded that our proposed solution will respond to their needs of improvment in the quality of the product. This has motivated the company to offer its services by providing the necessary automotive equipment for the testing and implementation of a prototype.

It was mutually agreed that the product should respond to a maximum of integration to minimize the importation of electronic devices, for the cost effectiveness of the product, by selecting primarily the locally fabricated devices. The system was then designed accordingly although the design requires integrated circuits, such as microprocessors, which, unfortunately, are still imported from abroad. However, the proposed solution was considered satisfactory in response to the following requirements:

(i) The system has to be cost effective, otherwise there will be no interest to replace the conventional harness by a multiplexed one though more 'intelligent', better reliable and easy to install and service.

(ii) The system should be compatible with the existing electrical system and should be easily adaptable with relatively minor changes to other vehicle specifications.

(iii) The system should be easy to use and transparent to the vehicle's driver.

Other design constraints are inherent to the nature of the vehicle's environment. [7,8]

Climatic conditions:

Temperature: the temperature range for automotive environment appliances exercises a decisive influence on the choice of the circuit and components. Typical ambient temperature range varies between -30 and +85 °C. The upper temperature limit may reach 100 or 120 °C depending on the mounting location (engine compartment, exhaust system). To these influences must be added the self-heating resulting from the power dissipation of the electronic devices.

Humidity: protection measures against humidity of electronic devices and components are essential for system reliability and life-time. Electronic devices have to be protected not only against splash water and condensation but also against other operating materials such as oils, petrol, and greases or even road grit if their mounting location permits contact with such agents.
Sealed housing, lacquering of printed circuits, and sealing with synthetic resins or silicon rubber are examples of such protection measures.

- Mechanical vibration
- Electromagnetic interference
- Supply variations
- Switching transients
- Power consumption

All these parameters have to be taken into account for the choice of system architecture, electronic components used, and mounting locations of electronic modules.

II.2 Electronic design options

As we have seen previously, inexpensiveness and improved reliability of semiconductor devices have played an important role in launching multiplex system research activities. Three electronic approaches to multiplexing are considered: wired logic, custom controllers, and microprocessors.

Table 2.1 gives a comparison of their main characteristics. Wired logic is far below to fit the growing needs of the automotive industry. Custom controllers option exhibits no adaptability, which makes its application range very specific and limited. It is a poor choice to suit the changing and diverse nature of the automotive market.

Intelligent electronics or microprocessor option seems to be the most suitable choice for multiplex system development and implementation. A microprocessor-based system offers much more design flexibility and system adaptability. It also allows to plant autonomous intelligence at different locations of the vehicle which, in turn, increases system reliability.

Approach	Wired logic	Custom controllers	Microprocessors
Functions determined by	Hardware	Hardawre	Software
Circuitry	Dedicated	Dedicated	Basically same for each application
Critical factors	Logic and circuit design	Design and development	Software development
Number of components per system	High	Low	Low
Adaptability to other models	Low	Nil	High
Unit cost optimum at	Low produc. runs	Very high prod. runs	Medium production runs

Table 2.1 Comparison of electronic design options.

Following, is a discussion of the different ways to interconnect microprocessor-based systems.

II.3 Network topologies

The principal technology alternatives that determine the nature of a local network are the topology and the communication medium of the network [9]. Together, they in large determine the type of data that may be transmitted, the speed and efficiency of communications, and even the kinds of applications that the network may support.

Based on these two technologies, 2 classes of local networks are defined with different configurations:

Store and Forward: a network in which a complete packet or block of data is received into a buffer in the memory of an intermediate node before being retransmitted on the route to its destination. In general, the nodes are interconnected by independent point-to-point transmission lines.

Broadcast: a network in which a message transmitted from one station is received by all other stations. In general, the stations are all connected to a common transmission line and so a particular station must detect that a message is addressed to itself. All stations are directly connected by the common transmission medium and so communication does not involve an intermediate node.

II.3.1 Example configurations

a) Complete interconnection

Each node in the network is connected by a dedicated point-to-point transmission line to all other nodes as shown in figure 2.1

This store and forward topology exhibits high integrity as the failure of one node does not affect communication between the others. However, it is not commonly used because of the high implementation and expansion costs.

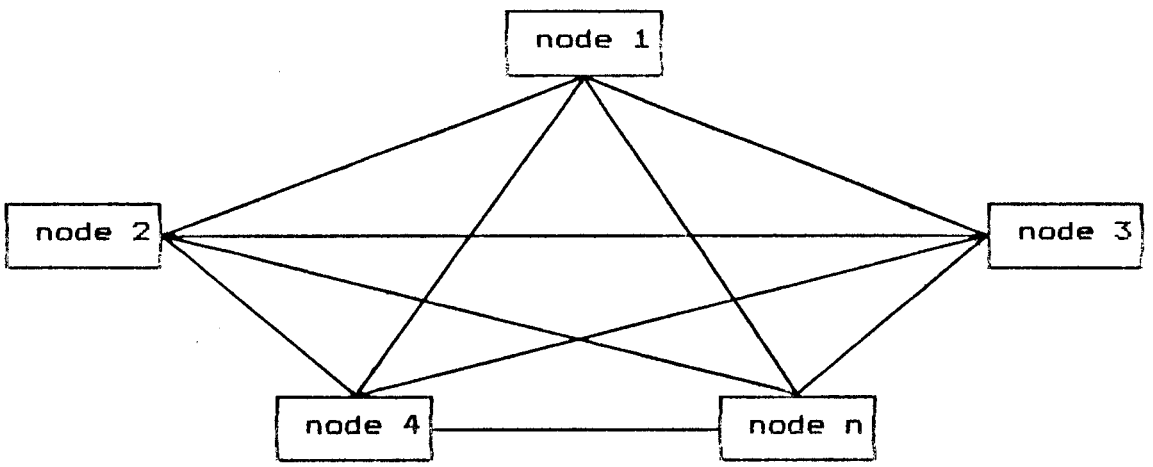


Fig. 2.1 Complete interconnection

b) Mesh interconnection

A topology in which each node is to be connected to at least 2 other nodes to provide alternate paths (Fig. 2.2). This structure exhibits also high integrity but requires complex routing software. The network is store and forward and so the delay depends on the number of intermediate nodes.

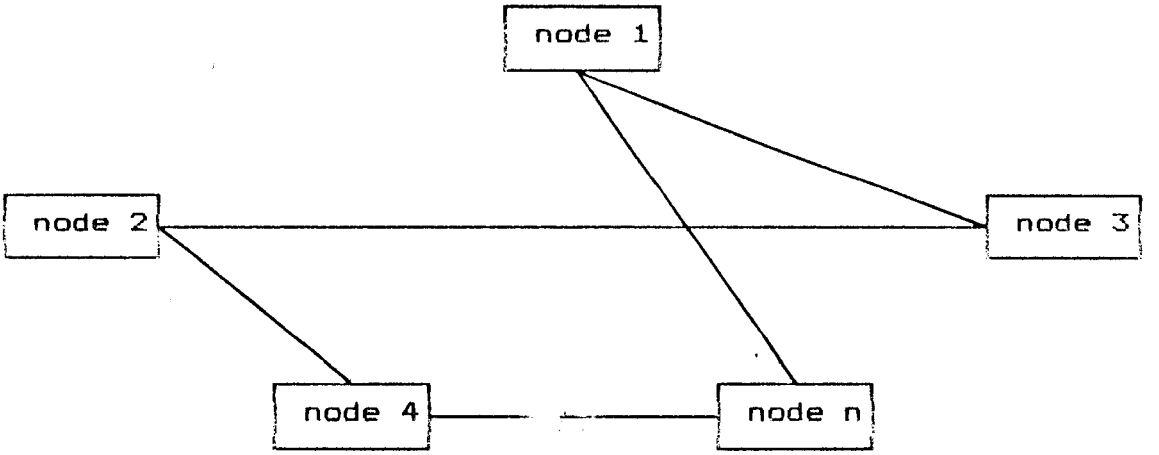


Fig. 2.2 Mesh topology

c) Star topology

In a star topology, a central switching element is used to connect all stations in the network (Fig. 2.3). The central element uses circuit switching to establish a dedicated pattern between two stations wishing to communicate.

This store and forward configuration provides low wiring costs, which also results in low expansion cost and low delays.

A major drawback of this topology is that if the central switching element fails, the entire network is lost. Also any cut of the communication medium will cause permanent isolation of the corresponding station.

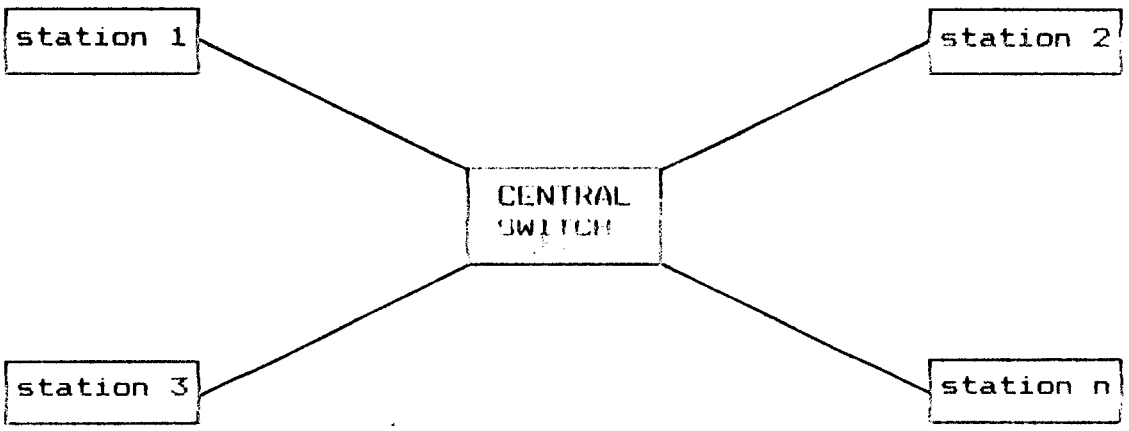


Fig. 2.3 Star topology

d) Tree topology

It is an extension of the star topology (Fig. 2.4) and so has very similar characteristics. Namely, it is a store and forward topology where all the nodes occur in a hierarchical sequence from one end to another. Implementation complexity and reliability problems make this a poor choice for connecting multiple processor networks.

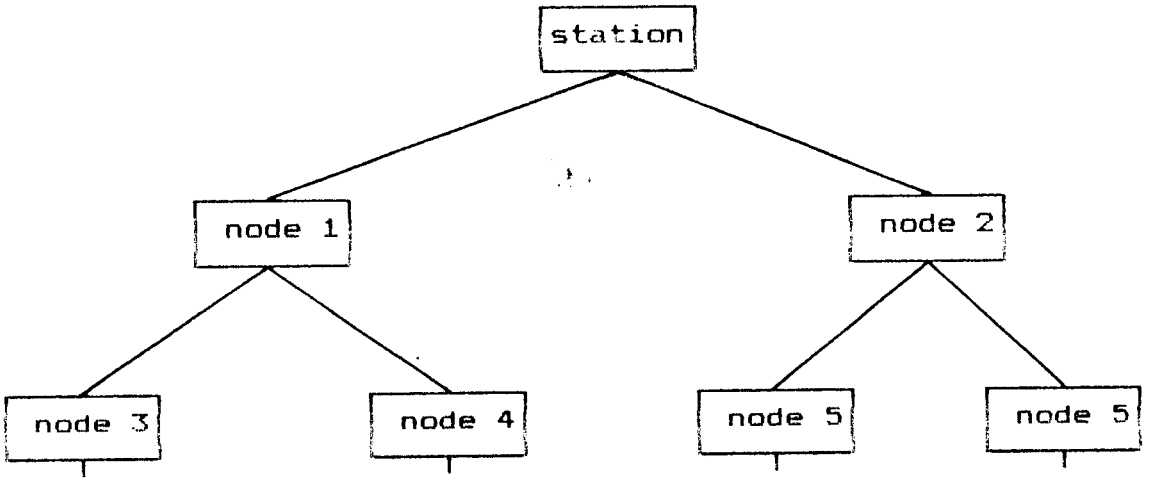


Fig. 2.4 Tree topology

e) Bus topology

Also known as a highway or multidrop link (Fig. 2.5). It is a broadcast system in which all the stations are connected to a common communication medium. Bus topology exhibits low wiring and expansion costs as well as simple software (broadcast communication). However, it suffers integrity problem as the communication fails if the transmission medium is cut. Also, there is no way to communicate with isolated stations unless redundant transmission line is provided.

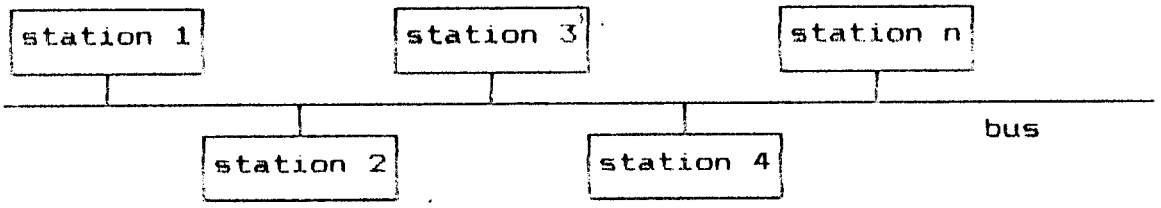


Fig. 2.5 Bus topology

f) Ring topology

In a ring topology (Fig. 2.6), the processors are logically connected into a circular structure. This has the advantage that any transmission medium can be used: twisted pair, coaxial cable or optical fiber. Typical application suited to a ring system, is a network in which the processors are not in physical proximity and in which each processor is normally occupied with local tasks, not involving the other processors.

The topology can be considered as an extension of the bus structure and so exhibits similar characteristics except that complete isolation of any station requires a cut at both sides of the connection and hence it provides better integrity.

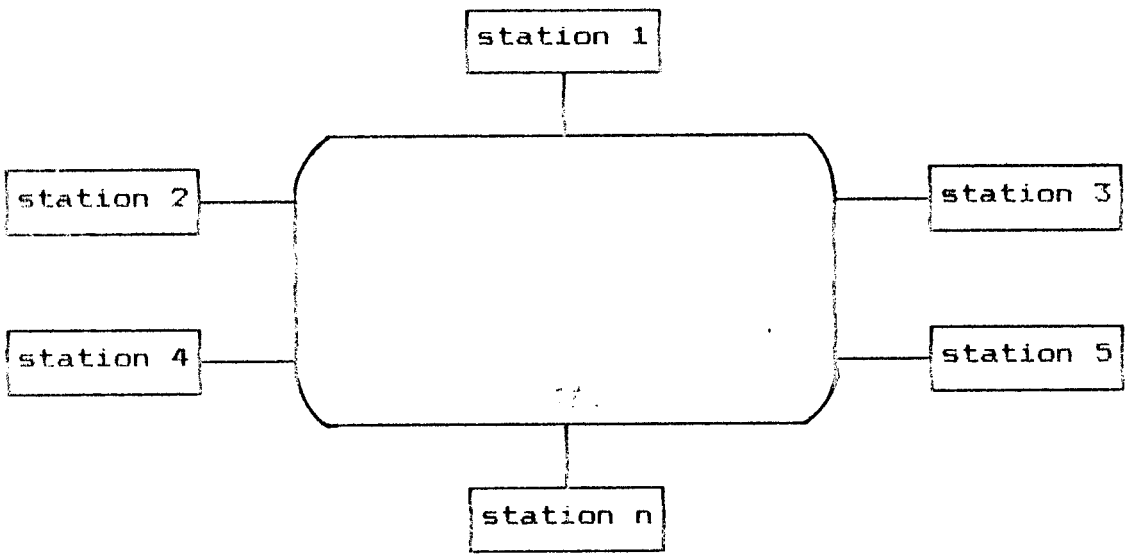


Fig. 2.6 Ring topology

II.3.2 A comparison of network classes

Table 2.2 outlines the arguments for the Store and Forward and Broadcast classes of networks [10].

BROADCAST	STORE AND FORWARD
- No routing, simpler software	- More complex communication software
- End-to-end error control	- Point-to-point controls required for efficiency and end-to-end controls needed for intermediate node failures.
- The destination address must be recognized before a message can be received by a station	- The message is first received then the destination address can be checked at the message has reached its destination or should be routed on
- One transmission line has to support all the communication with the network, high speed lines (1 to 10 Mbits) required.	- Multiple low speed transmission lines can be operated in parallel.
- Wait for access contention delay	- Intermediate nodes delay
- Transmission path can be completely passive and so be inherently reliable	- Transmission path includes switching nodes so is less reliable
- Redundant paths required in case of line cut	- Alternate paths easily achieved in mesh networks
- Low wiring costs	- More wiring required
- Suitable for local networks	- Suitable for geographically dispersed networks

Table 2.2 Comparison of Broadcast and Store and Forward classes

II.4 Design strategy

Having examined the different possible network topologies for the multiplexed wiring harness and the parameters affecting the design process, it is time to give the idea about the philosophy behind the design of the INELEC Multiplexed System (IMS).

System electronics:

For design flexibility and adaptability reasons, a standard low cost microprocessor has been used in conjunction with a set of single-chip microcomputers to constitute the core of the system's electronics. This is due particularly to the changing nature of the automotive industry. Another point is that the SNVI-CVI company has required that the system can be practically implemented on its various vehicle models with few or no changes, therefore system electronics has to exhibit some flexibility to be easily adaptable to meet different requirements of different models. This feature cannot be achieved using other electronic options such as wired logic or custom controllers.

Multiplexed network:

It appears that a broadcast ring architectured network used in a centralized environment is more appropriate to our system than any other configuration as it suffers less wiring costs and presents easier expansion facilities and adaptation to other vehicle models.

We have chosen a master-slave centralized system instead of a distributed system for the following reasons:

- A peripheral failure does not upset the whole system.
- In the event of the master controller failure, the slave units switch to emergency operation.
- System expansion flexibility.
- Easy localization of failures.
- Simpler bus arbitration software.

Technology:

Concerning the technology used to implement the system, we have seen previously that CMOS was the most appropriate technology to automotive applications, particularly in terms of immunity to supply variations and low power consumption. However, we were not able to suit this requirement as CMOS devices were not available during the time of development. Instead, we used NMOS and even some TTL devices.

Communication:

Synchronous communication is usually selected in applications where continuous data transmission/reception is required and/or where the enhanced data recovery capability of this method is needed. The important distinction between synchronous and asynchronous communications is the fact that in the first one, a clock signal that is in perfect synchronization with the received data is required at the receiving unit. Synchronous communication is most useful in applications such as continuous remote control and guidance. It is not commonly used with small microprocessor systems.

In asynchronous systems, the communication medium is in mark (binary 1) condition in its idle state. As each character is transmitted, it is preceded by a start bit or transition from mark to space (binary 0), which indicates to the receiving unit that a character is being transmitted. At the end of that character transmission, the line is returned to mark condition by one or more stop bits. The start and stop bits permit to synchronize data transmission on a character-by-character basis.

Asynchronous communication is advantageous as it is inexpensive due to simple interfacing required.

Power bus:

In a ring architectured system, for reasons of ease of installation and harness simplicity, it seems expedient to construct the power line using a uniform wire cross-section, designed to handle the maximum load currents at any point, and loop it around the vehicle along with the communication bus. To reduce cost and weight, it is preferable that each electronic module and its corresponding sub-harness use power line whose cross section varies in accordance with individual demands.

In the following chapter the system description is given.

CHAPTER THREE

INELEC MULTIPLEXED SYSTEM

. General description

. Master control unit hardware

. Slave control unit hardware

. Load switching and monitoring

INELEC Multiplexed System

III.1 General description

The system consists of the master control unit (MCU), the slave (local) control units (SCU) and the transmission lines: a power bus and a control bus. Figure 3.1 shows the overall system configuration.

A power supply is provided with each electronic module to supply it with the appropriate working voltages : +5 and +12 V are directly derived from the +24 V power bus through 7805 and 7812 voltage regulators respectively (see figure 3.2). Negative supply voltage (-12 V), required for bus communication, is provided using an Intersil 7660 dc-to-dc voltage converter whose output is regulated by a 7912 voltage regulator. Diode D_1 is used to protect the circuitry from battery reversal whereas capacitor C_1 is used to eliminate high frequency spikes.

The multiplexed harness is controlled by the MCU (Fig. 3.3) located in close proximity to the rear of the dashboard so that all control inputs can be connected directly.

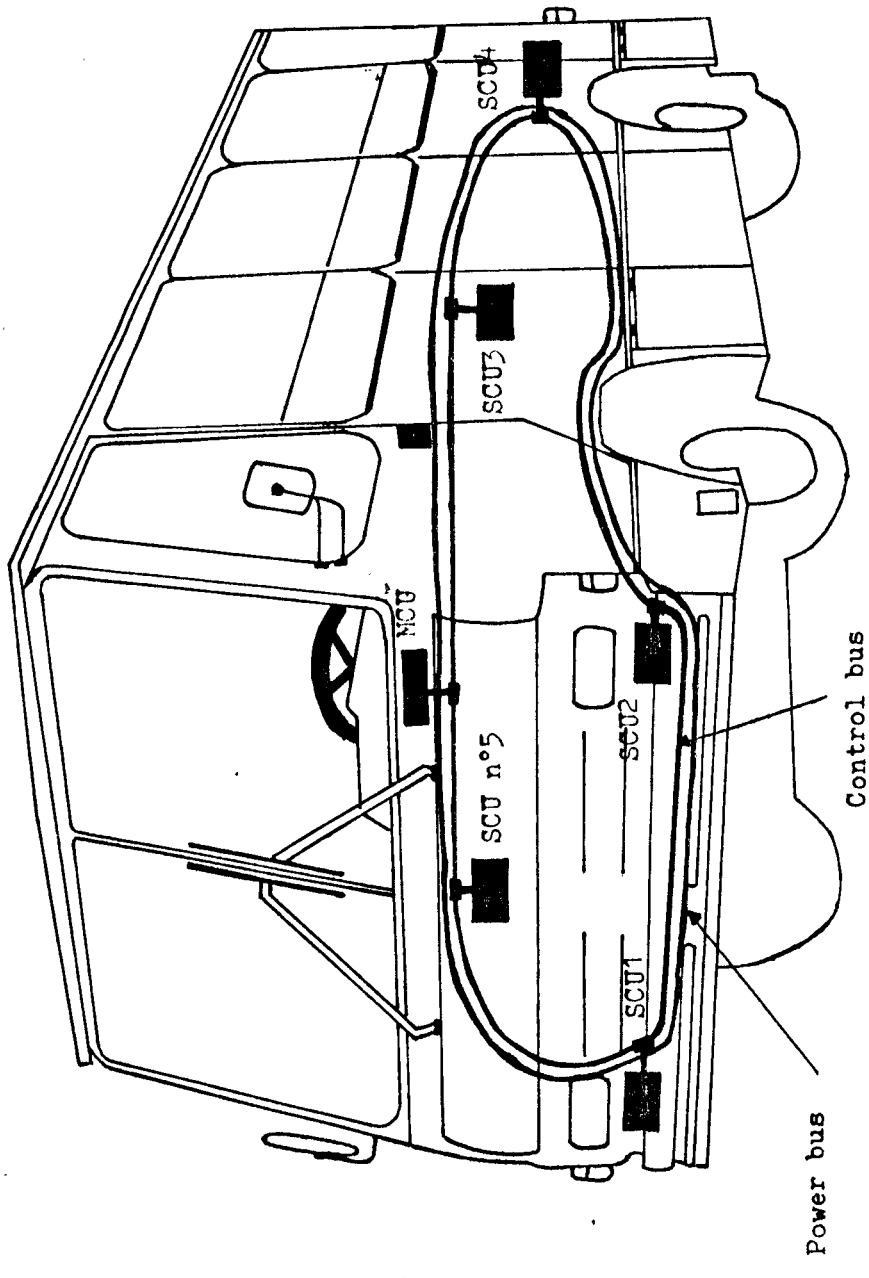


Fig. 3.1 Electronic harness configuration for '2504' minibus.

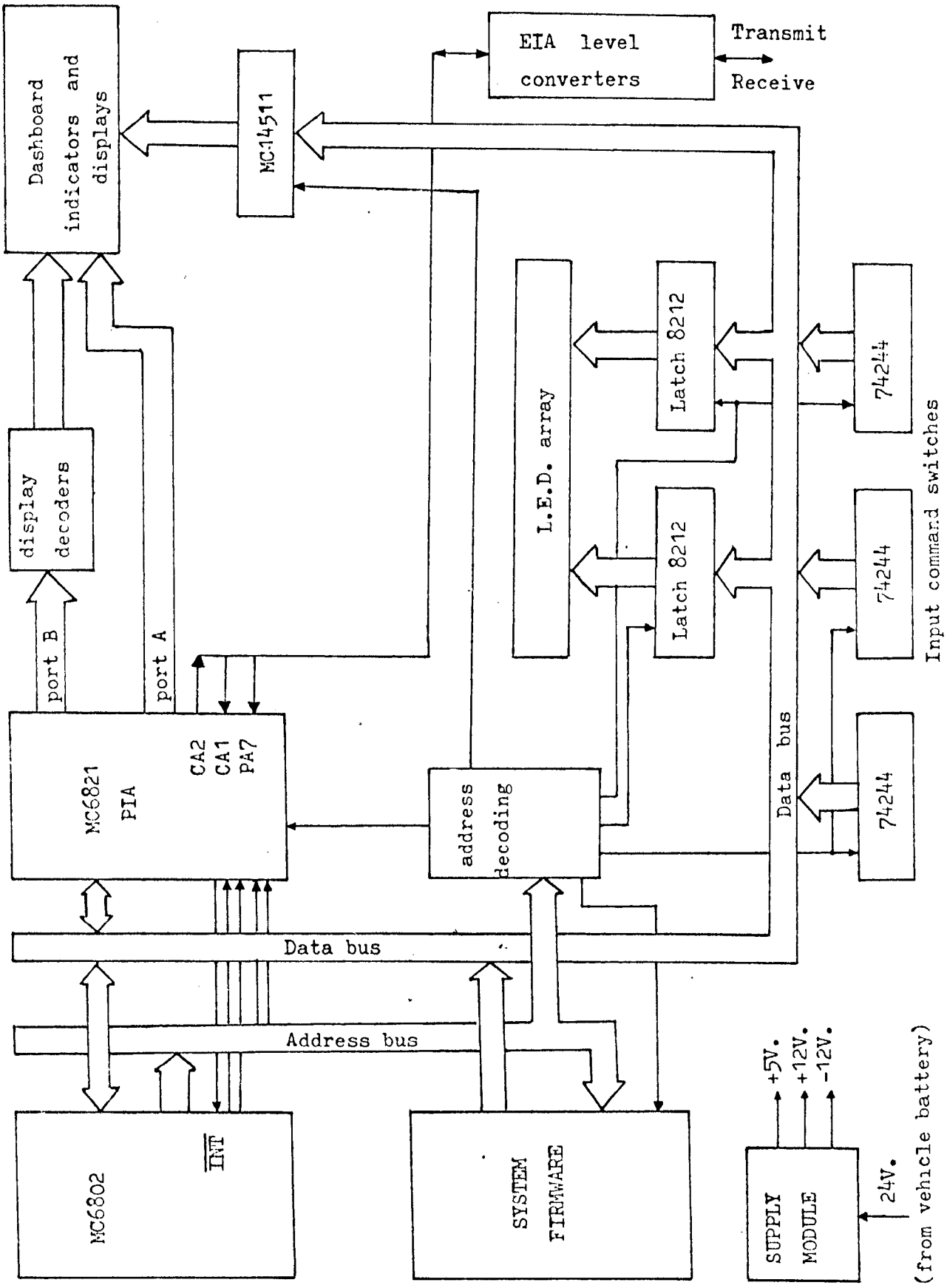


Fig. 3.3 Master control unit block diagram

Some control commands may be transparent to the MCU as they can locally be sensed and processed by the slave control units (eg. some bus models have their rear door open/close switch located at the rear of the vehicle). Only monitoring data of such control commands is reported to the MCU.

After interrogating driver's commands, the MCU broadcasts a message containing all necessary information about the functions to be performed by the slave control units. All the SCUs receive the message simultaneously, perform the required switching functions, but only one SCU is allowed to report back appropriate data (eg. motor RPM, temperature, etc...) and diagnostic information about the performance of the driven loads.

Figure 3.4 shows the schematic diagram of a typical SCU. It consists mainly of a processing module, a power supply module, and a power switching module.

Power up system restart can be initiated whenever appropriate by turning off and on the main power switch. A separate system restart switch can be provided but with extra wiring costs.

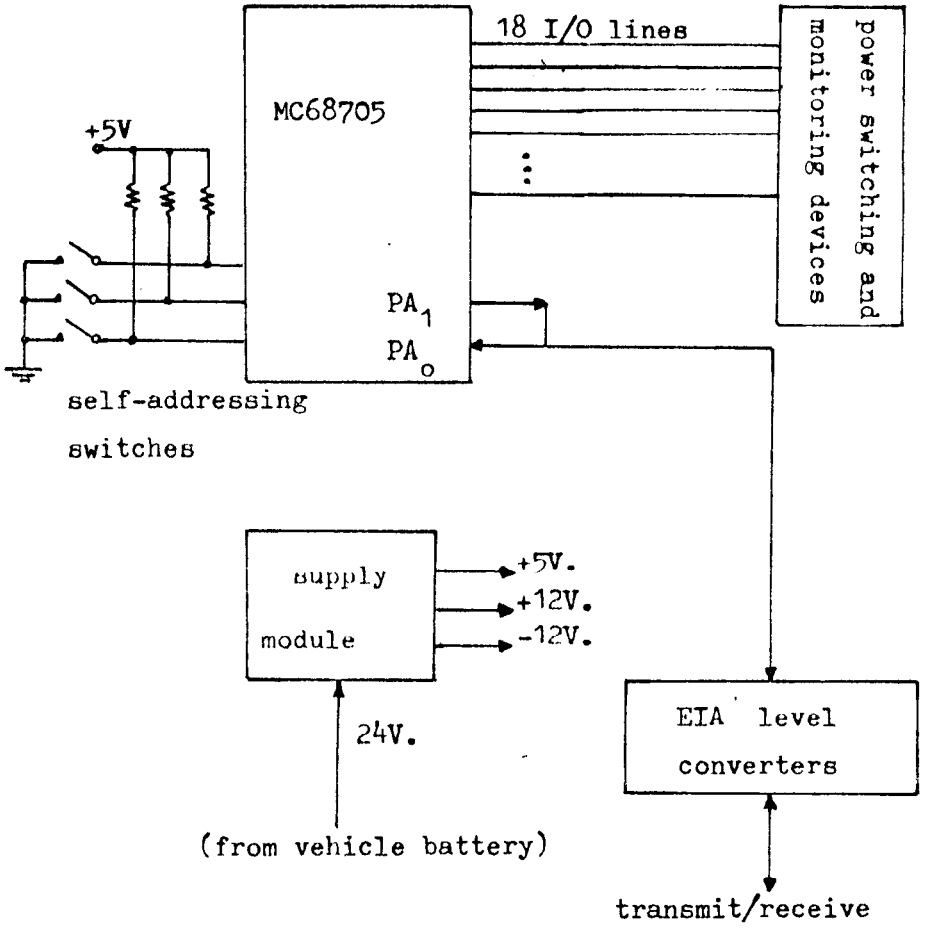


Fig. 3.4 Slave control unit block diagram

System specifications are summarized below:

- Multiplex transmission method: centralized LAN.
- Logical structure: master-slave.
- Bus access: polling.
- Priority control: no.
- Acknowledgement: inherent.
- Addressing: broadcast.
- Error detection: false start bit, parity and framing.
- Bit synchronization: asynchronous.
- Transmission rate: 10.4 Kbps.
- Topology: ring.
- Medium: coaxial cable.
- Communication levels: -12V, +12V.
- Bit time: 96 µS.

III.2 Master control unit hardware

The master control unit circuit is based upon the Motorola MC6802 microprocessor.

This microprocessor has the following features [11]:

- 8-bit word size
- 128 bytes of internal RAM, of which 32 are retained on power down
- On-chip clock circuit
- 16-bit memory addressing
- Software and external interrupts.

Figure 3.5 shows the circuit diagram. It consists mainly of a processing section and an I/O section.

The master control unit (MCU) 24 input lines are directly connected to the central processing unit (CPU) data bus via octal bus buffers (74244). They are memory-mapped so that they are accessed as if they were memory locations. The number of input control switches, if revealed to be not sufficient for other vehicle models, can be increased to cover all possible vehicle switching requirements provided that address decoding and the necessary software adjustments are to be met.

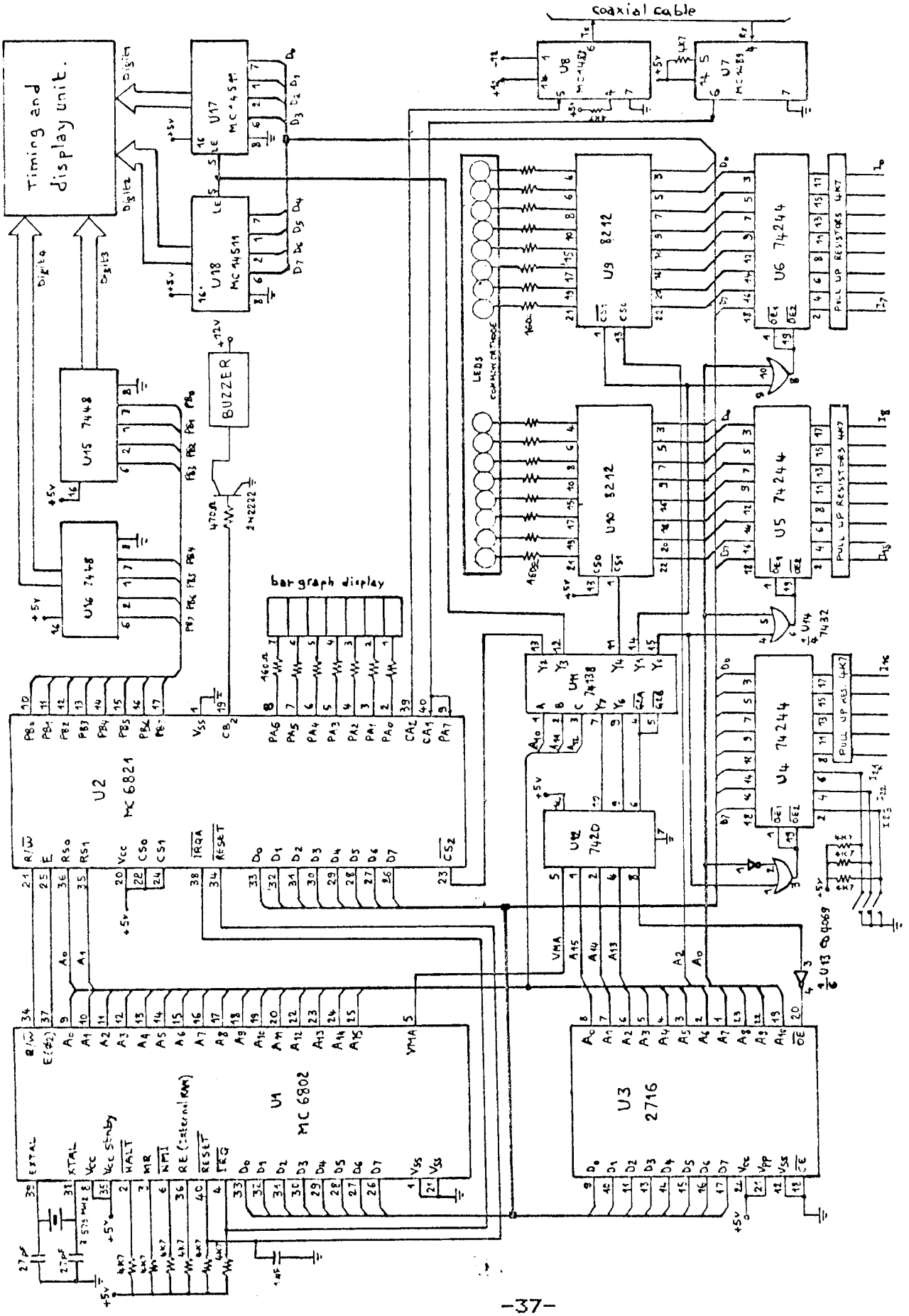


Fig. 3.5 Master Control Unit circuit diagram.

Table 3.1 shows the functional allocation of the different input lines. These lines are active low so that to eliminate spurious command inputs.

INPUT SWITCH	FUNCTION	SLAVE UNIT(S) CONCERNED
I0	Headlight main	1,2
I1	Headlight dip	1,2
I2	Flasher right	1,3
I3	Flasher left	2,4
I4	Back up light	1,2,3,4
I5	Hazard flasher	1,2,3,4
I6	Parking light	1,2,3,4
I7	Ceeling lights (passengers)	5
I8	Ceeling lights (driver)	5
I9	Windshield washer	5
I10	Windshield wiper right	5
I11	Windshield wiper left	5
I12	City horn	5
I13	Urban horn	5
I14	Brake (stop) light	2,4
I15	Ignition key	master

Table 3.1 Input control switches

The MCU controls a 4-digit liquid crystal display unit. The rightmost two digit are driven directly from the system data bus through two MC14511 BCD-to-seven segment latch/decoder/drivers. These two digits are used to display vehicle engine temperature in °C. The two other digits are driven by port B of the MC6821 Peripheral Interface Adapter (PIA) via two 7448 BCD-to-7-segment decoders and are used to display the number of motor rotations per minute (RPM).

It is worth to mention here that in order to prevent electrolytic decomposition of the liquid crystal display, an alternating drive voltage has been used. This is a low frequency square wave (7.67 Hz) applied to the 4-digit L.C.D. back-plane. Exclusive OR gates (7486) are used to provide out phase or in phase signals to the ON or OFF segments respectively. The low frequency signal is provided by the well known 555 timer connected in an astable configuration. The 7490 counter is used to derive the 7.67 Hz signal for L.C.D. back-plane and the 1.91 Hz signal used as flasher time-base.

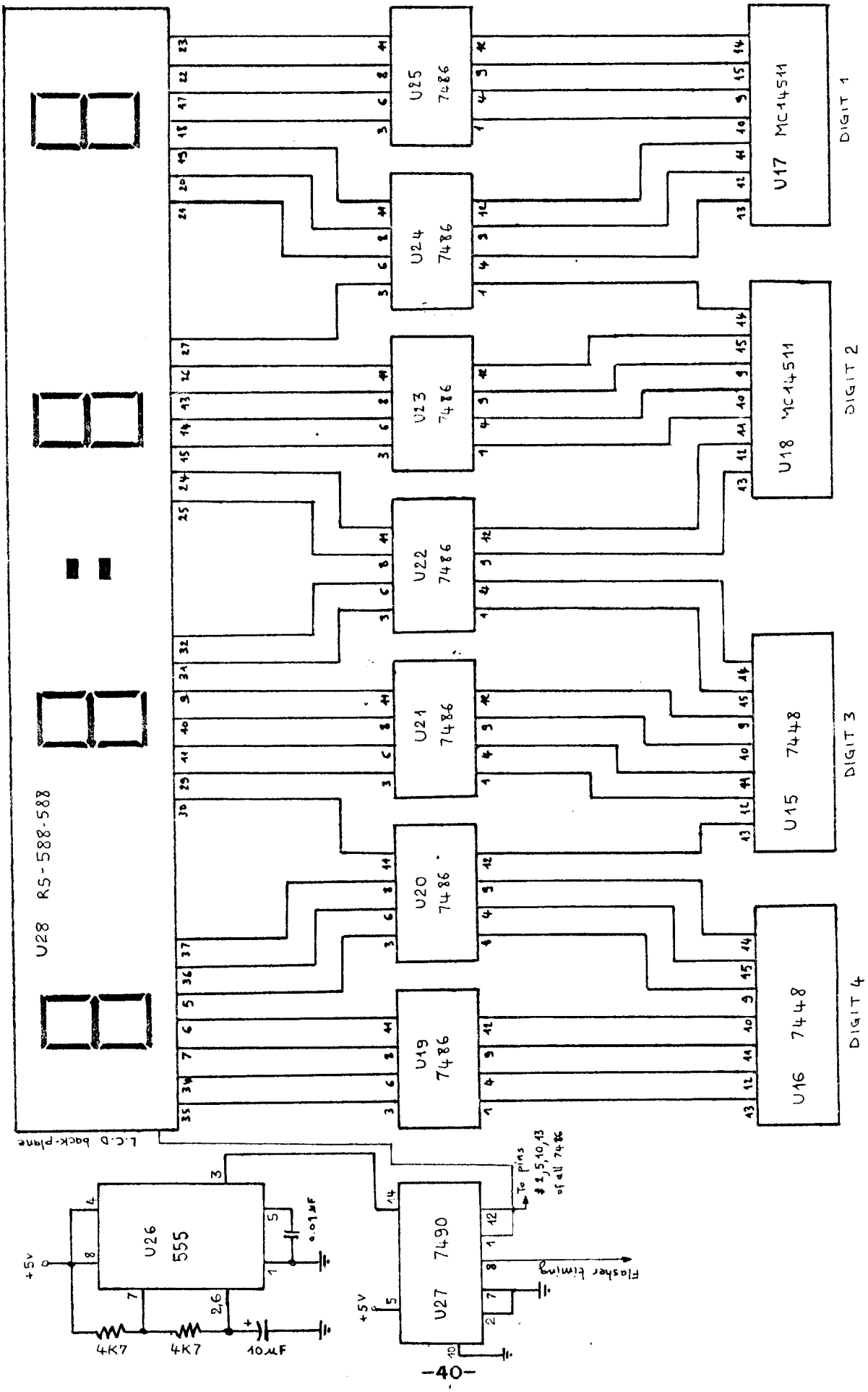


Fig. 3.6 Timing and display unit.

Port A (I/O lines PA_0 thru PA_6) of the PIA is used to drive an LED bar-graph to display the tank fuel level, while two 8212 latches are used to drive an array of 16 LEDs to display different dashboard indicators such as turn signals and headlight, as well as warning signals such as SCUs fault indication. Fault indication by a symbol being more appropriate than written failure indication (codes or sentences), symbols or graphs can be placed above the LEDs in order to guarantee a quick and unequivocal fault indication.

An electronic buzzer is driven by PIA port B I/O line CB2 to provide an audible signal synchronized with dashboard flasher LEDs indicators. Buzzer on and off switching is achieved using a 2N2222 transistor.

The serial communication timing is based upon the CPU clock which is driven by an external 3.579 MHz crystal. The peripheral interface adapter I/O port A line PA7 is used in conjunction with control lines CA1 and CA2 to connect the MCU to the overall system communication medium via two EIA level converters 1488 and 1489 that are used to translate 0 and 5 V signals to +12 and -12 V respectively and vice versa.

System memory and CPU peripheral devices are mapped in table 3.2.

bit counter	0000
output character	0001
input character	0002
byte counter	0003
parity counter	0004
interrupt flag	0005
B pseudo-accum.	0006
error flag	0007
slave counter	0008
dead lock flag	0009
display location	000A
diagnostic table	0010
slave table ptr	0020
time-out counter	0022
scrach RAM	0024
command message	0030
slave answer	0034
slave table	0050
diagstic. tab.ptr	0059
system stack area	0060

0000	system RAM
007F	not used
E000	switch 2 (U5)
E001	switch 1 (U4)
	not used
E400	switch 3 (U6)
E405	latch 8212 (U9)
E800	PIA DDRA (U2)
E801	PIA CRA
E802	PIA DDRB
E803	PIA CRB
EC00	MC14511 (U17,U18)
F000	latch 8212 (U10)
	not used
F800	EPROM 2716 (U3) system monitor and vectors

Table 3.2 System memory and I/O peripherals map

III.3 Slave control unit hardware

The slave units hardware (see Fig. 3.7) is based upon the Motorola MC68705P3 microcomputer unit. It is an EPROM member of the 6805 family of single-chip microcomputers [12]. Its main features are:

- 8-bit architecture,
- 112 bytes of RAM,
- 1804 bytes of user EPROM,
- internal 8-bit timer with 7 bit prescaler,
- on-chip generator,
- 20 I/O lines,
- External, timer and software interrupts,

all in a 28-pin package.

All the slave units, except for the engine unit, have the same logic and differ only in:

- their address code,
- the number of power switching devices used
- the presence of analog signal inputs.

The engine slave unit, see figure 3.8, is an R3 version of the same single-chip microcomputer family which incorporates an analog-to-digital converter and offers more I/O lines and EPROM size [13]. The microcomputer memory map of both versions is given in appendix A. Plate 2 shows a typical SCU module.

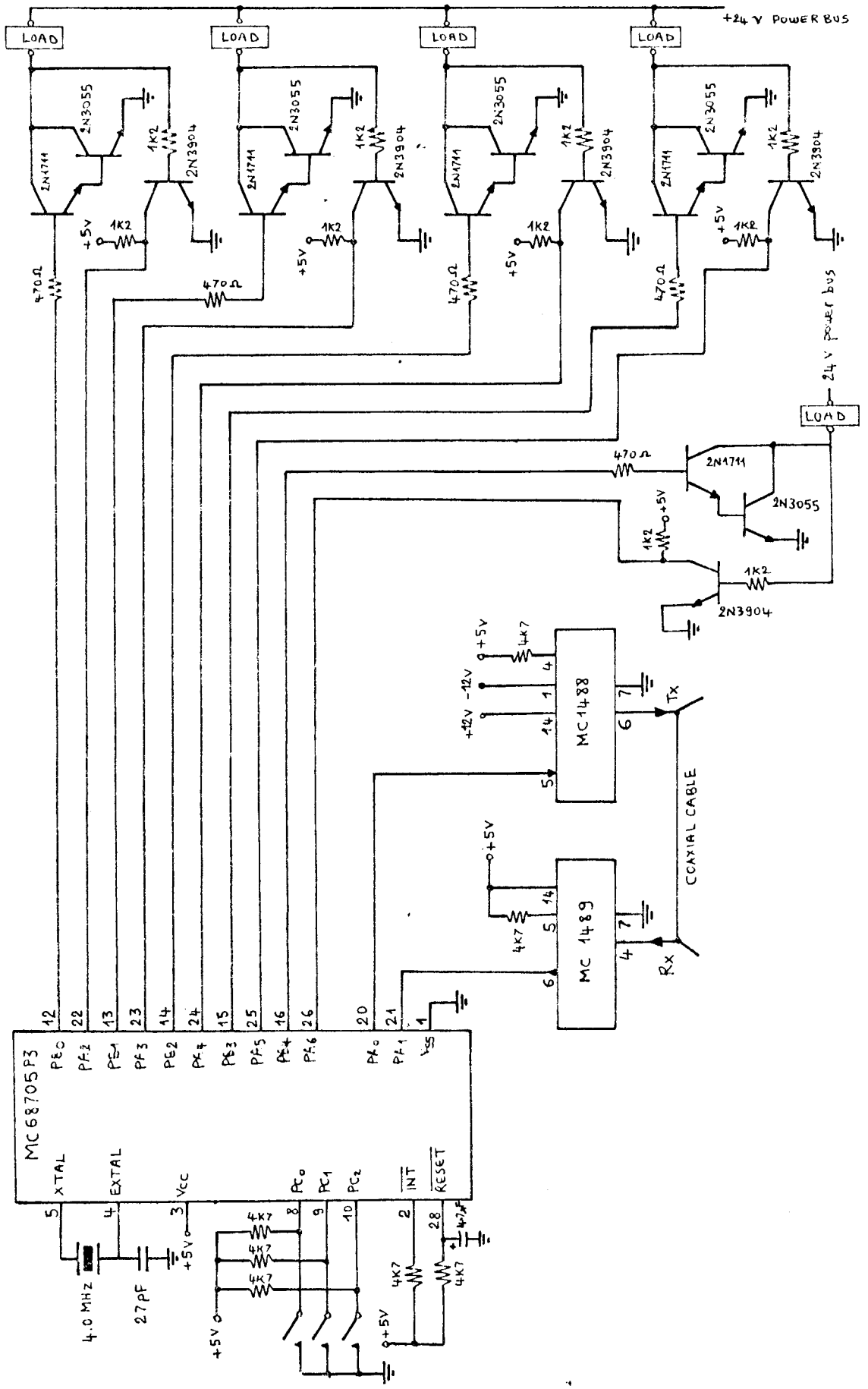


Fig. 3.7 Slave control unit circuit diagram.

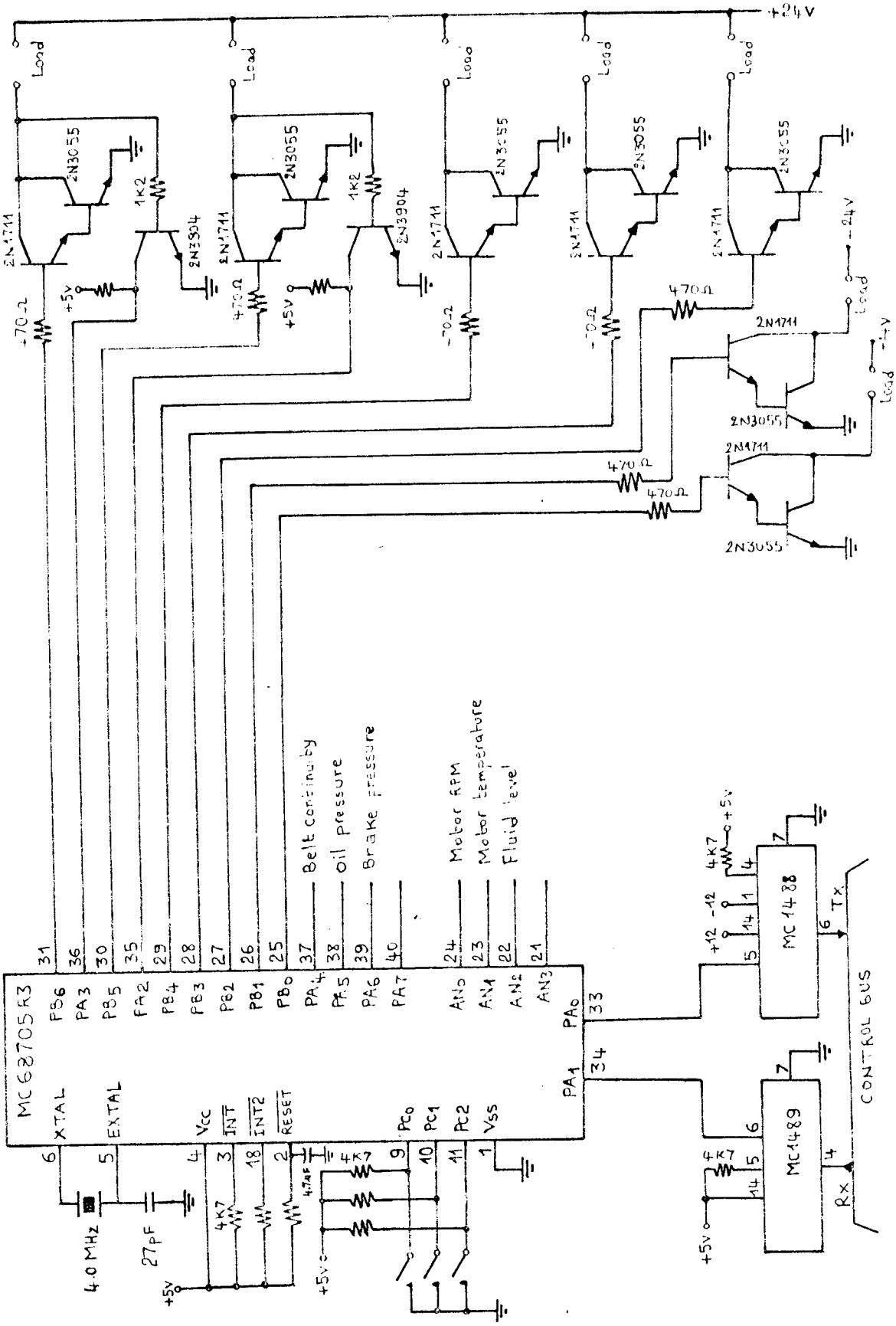


Fig. 3.8 Engine ECU circuit diagram.

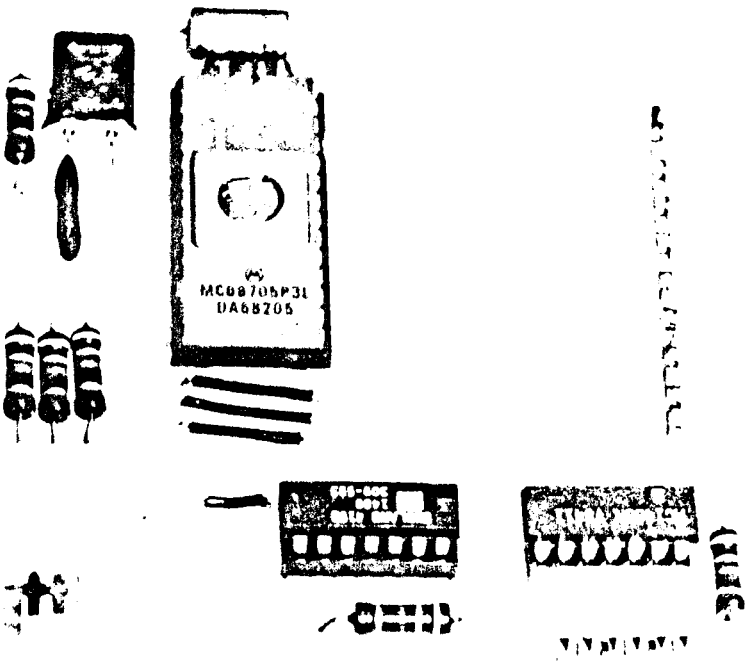


Plate 2: Typical SCU module.

The slave control units are designed to be self-addressing, i.e., each unit allocates an identification number to itself corresponding to the mounting location of the electronic module (see table 3.3). The address of the SCU is read from 3 switches via I/O port lines PC0, PC1 and PC2. This identification number forces the slave units to execute only those functions specific to their respective physical locations. This feature greatly simplifies system assembly and maintenance.

Serial communication is performed via I/O port lines PA0 and PA1. Timing reference is derived from an external 4.0 MHz crystal in order to match MCU bit timing.

All the units receive the message simultaneously and each one extracts the appropriate control bits from the three-byte control stream. Only the destination slave unit, to which was addressed the control message, is allowed to access the bus, feeding the master control unit with necessary data and diagnostic information which, in turn, constitutes an inherent positive acknowledgement.

UNIT	IDENTIFICATION NUMBER
MASTER CONTROLLER	0
SLAVE FRONT RIGHT	1
SLAVE FRONT LEFT	2
SLAVE REAR RIGHT	3
SLAVE REAR LEFT	4
SLAVE ENGINE	5

Table 3.3 Address allocation.

If there were no feedback from the concerned slave unit within a time slot of 20 mS, it will be automatically flagged so that the master will no more send messages to this unit and its corresponding dashboard indicator (LED) is turned ON.

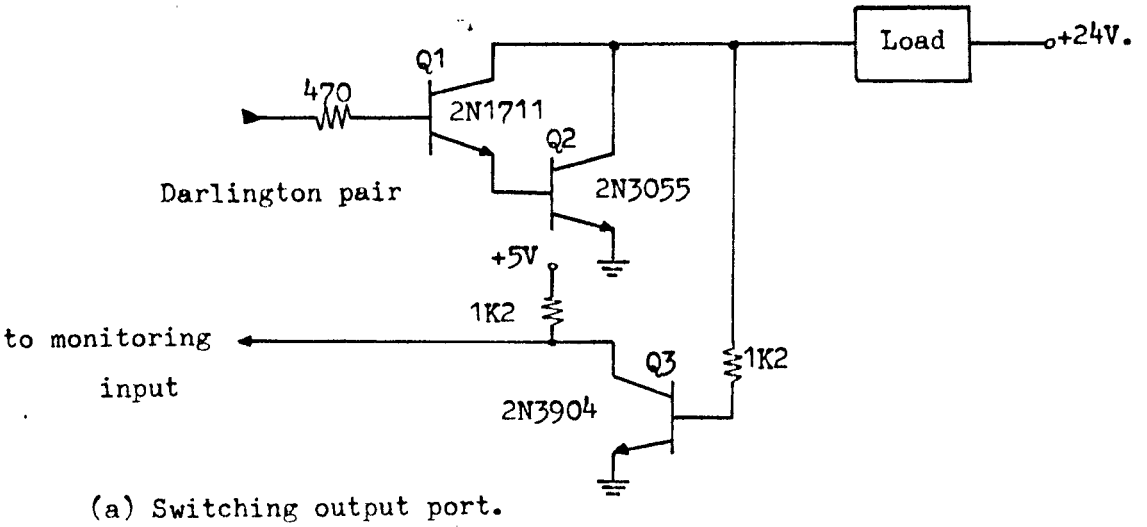
On the other hand, if there were no message emanating from the master control unit within a time slot of 25 mS, the slave control units are programmed to switch to emergency operation. The additional time slot of 5 mS is added so that the other slave units will not detect false master time-out in the time period during which the master is waiting an answer from an eventual defective slave unit. The fail safe operations performed are headlight dip, back up and hazard lights.

III.4 Load switching and monitoring

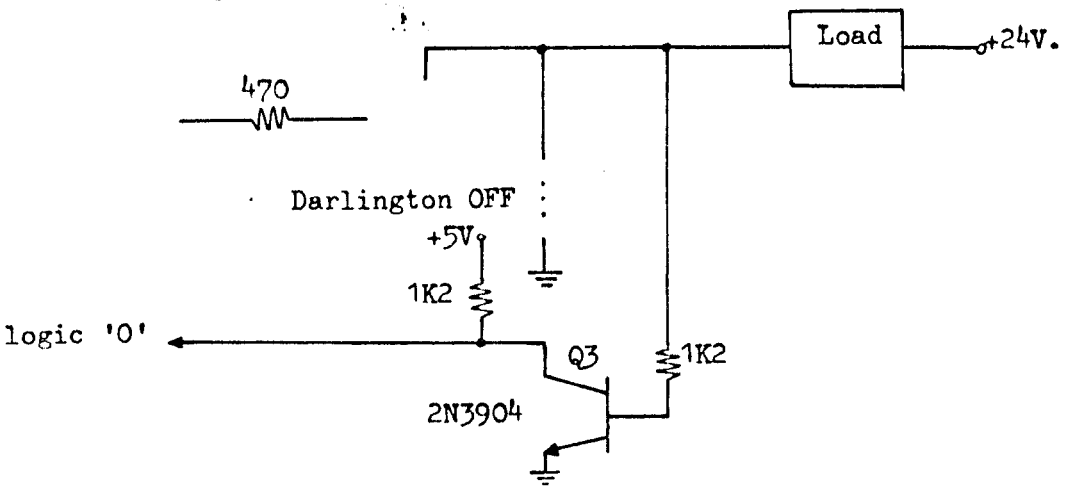
Power switching is achieved using NPN power transistors 2N3055 having a driving capability up to 15 A [14], preceded by a low power NPN transistor 2N1711 connected in a darlington pair configuration (see Figure 3.9-a). The use of NPN switching transistors requires that two fly-leads are to be connected to each load as the chassis ground (vehicle's body) cannot be directly connected to one of the load terminals. Direct chassis ground connection can be achieved using PNP darlington transistors that were unfortunately not available during the project development.

In the abscence of intelligent power switches, load monitoring was achieved using a general purpose transistor 2N3904. The load monitoring test is to be carried out while the power transistor is in the off condition. If the lamp filament is cut or the load fuse is blown, transistor Q3 will be cut off so that the monitoring output is a logical 1 indicating load fault condition (see Fig. 3.9-c). Figure 3.9-b shows that normal load condition will bias transistor Q3 so that the monitoring output will be a logical 0 indicating normal load operation.

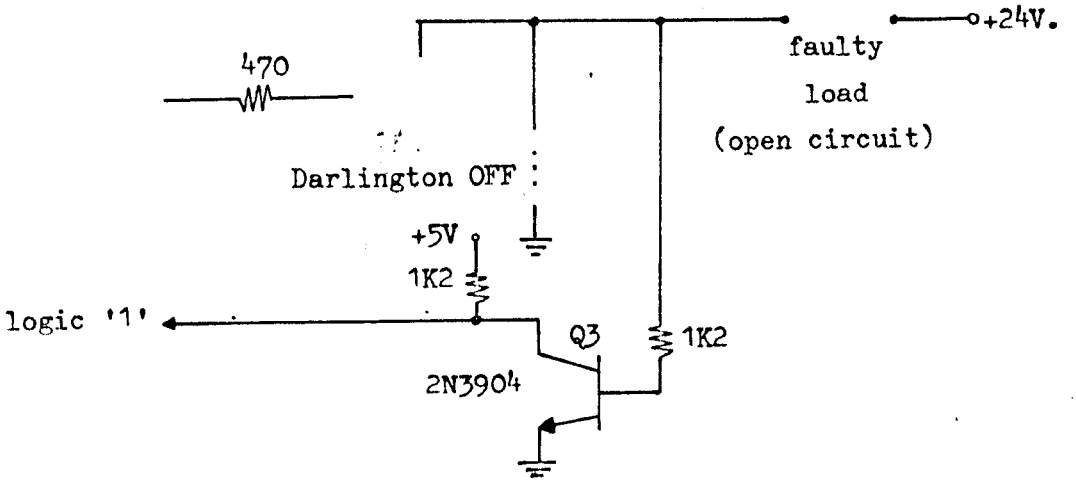
Plate 3 shows a typical power switching module.



(a) Switching output port.



(b) Equivalent circuit for normal operation.



(c) Equivalent circuit for fault condition.

Fig. 3.9 Load monitoring circuit diagram.

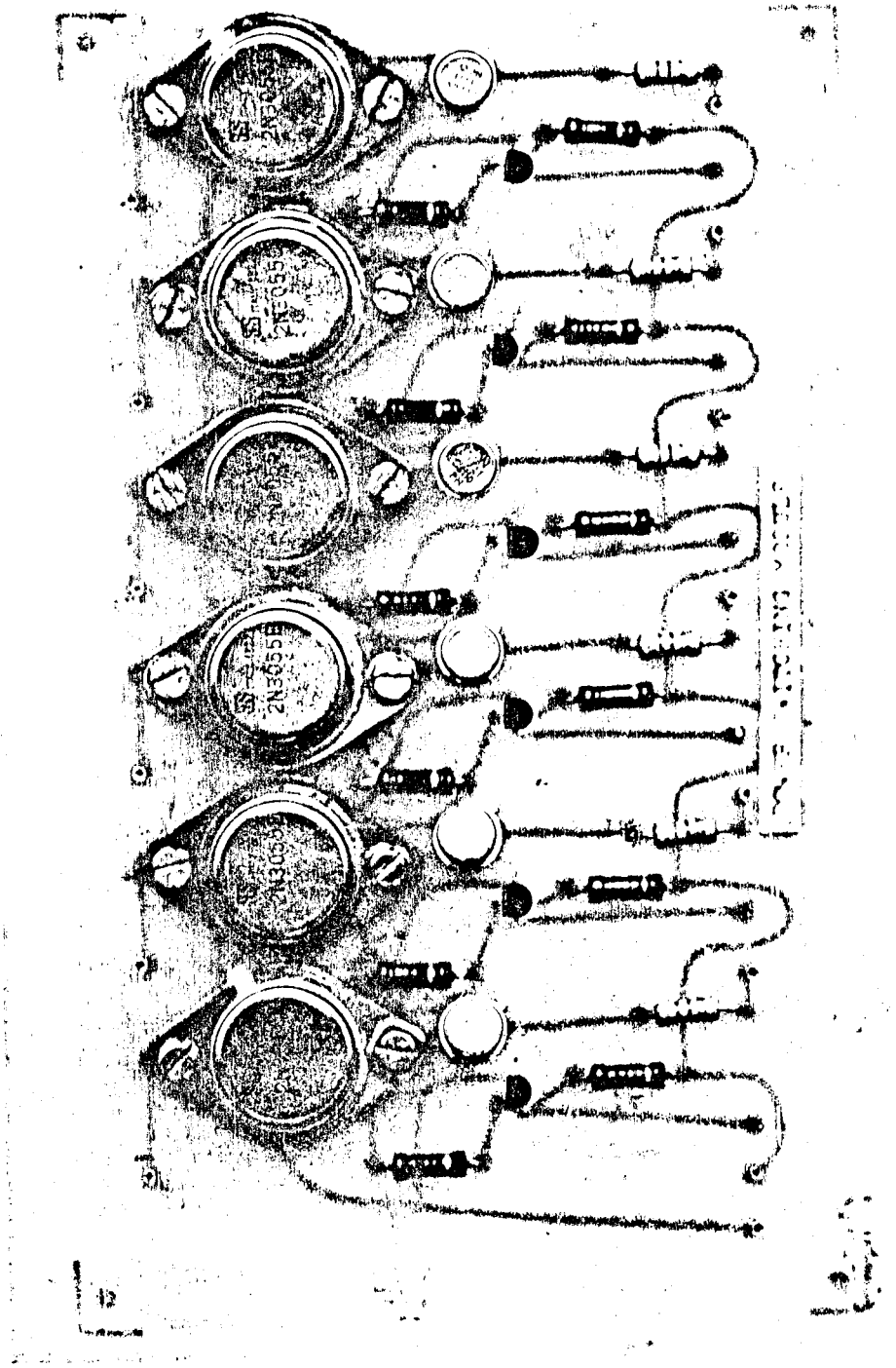


Plate 3: Typical power switching module.

CHAPTER FOUR

INTERPROCESSOR COMMUNICATION

. Serial communication

. System protocol

. Communication medium

. Data integrity

. Failure conditions

. Data recovery

Interprocessor Communication

IV.1 Serial communication

The automotive industry recognizes the advantages of developing standards for serial communication. A standard interface and protocol will limit prices and increase reliability through improved diagnostics.

IMS communication process incorporates some features of the SAE J1850 class B standard, concerning message size, bit time duration, inter-byte separation,latency and error handling.

The communication between the master and the slave units is asynchronous as it is cheap and simple. Bit timing and message serialization are software generated thus reducing system hardware and offering more system protocol flexibility as the baud rate and the message length can be easily changed to suit other requirements.

IV.2 System protocol

Protocol refers to the set of rules that are followed by interconnected devices on the bus in order to insure the orderly transfer of information. It is sometimes referred to as the line discipline or Data Link Control. Signetics outlines the functions performed by the protocol as in the following [15]:

1) establish and terminate a connection between two stations.
2) assure message integrity through error detection, requests for transmission, and positive or negative acknowledgement.
3) identify sender and receiver through polling or selection.
4) handle special control functions.

Obviously, the choice of a line protocol for a particular multiplex system is crucial to the ultimate success of the system in a given application. At INELEC, we desired to make the best choice possible on protocol for our multiplex system.

INELEC Multiplexed System (IMS) uses polling technique for bus arbitration in a master-slave environment. The master control unit has the overall priority over the communication bus and allocates bus access to different slave units in a cyclic manner.

Figure 4.1-a shows the general format of the message interchanged between master and slave units.

Each message is of fixed format and consists of four bytes. Fixed data length was chosen because the variable data length does not prove effective for improvement of response in view of small difference between maximum and minimum data lengths and the large overhead size of the frame format for end of message control. The variable data length offers the formal freedom, yet not much practical merits because, in any case, the memory for maximum data length need be secured and the software becomes complicated.

The message consists of several parts (see Figure 4.1-b):

- Start of transmission 1 bit.
- To address (destination) 4 bits.
- From address (origin). 4 bits.
- Control commands or data 3 bytes.
- Parity, stop and inter-byte 1 bit each.

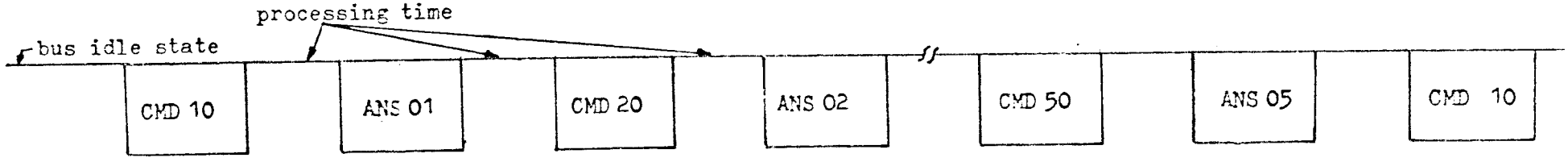


Fig. 4.1-a Protocol packet

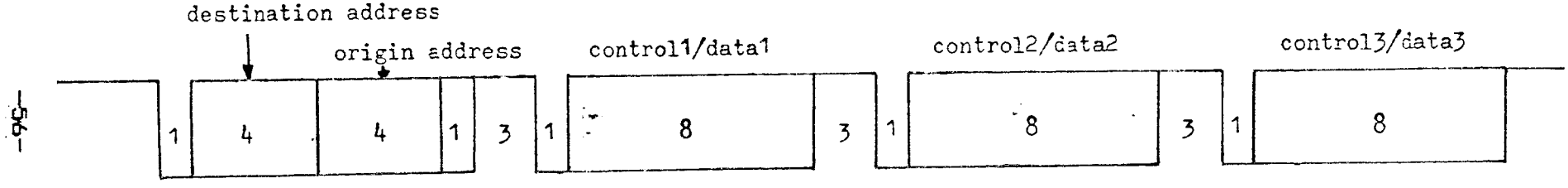


Fig. 4.1-b General message format

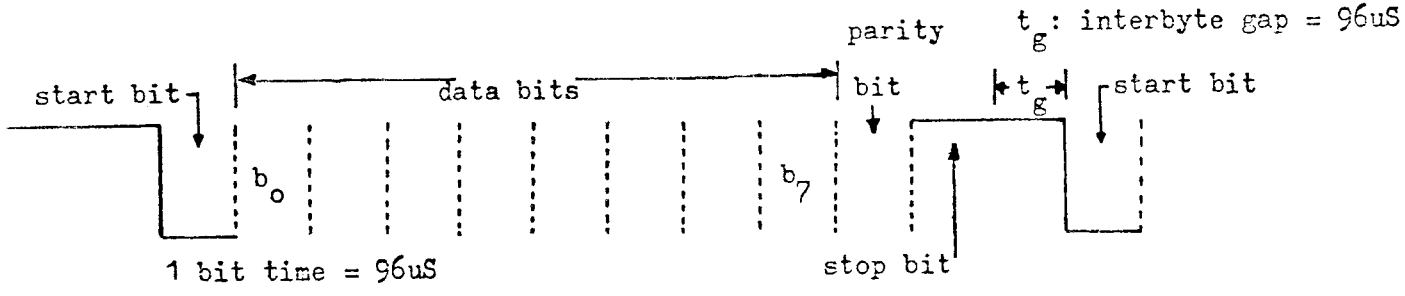


Fig. 4.1-c Message word format

IV.3 Communication medium

IMS uses a baseband 50-Ω coaxial cable as a communication medium (IEEE802 standard) [16], which is more suitable to digital data communication than the conventional 75-Ω coaxial cable, as it suffers less intense reflections from the insertion capacitance of the taps and provides better immunity against low frequency electromagnetic noise. As already mentionned, IMS is a broadcast ring-architectured network, consequently, simple passive T-connectors can be used to tap control signals at any point into the cable. This feature increases system expansion flexibility.

Other communication media can also be used such as twisted pair and fiber optics, but the later one requires active taps thus making it significantly more expensive. Table 4.1 gives a comparison of the main characteristics of a coaxial cable and an optical fiber.

	Coaxial	Fiber
Data rate/Km	50 Mbits	> 1000 Mbits
Accessibility to being taped	Easy	Difficult
Signal radiation	Yes	No
Bit error rate	1 in 10^{4}	1 in 10^{7}+
Grounding problem	Yes	No
Static problems	Yes	No
Size and weight by data rate	Large	Small
Installation cost	1N	2N

Table 4.1 Comparison of coaxial cable-optical fiber

IV.4 Data integrity

In automotive multiplex systems, because of the risk involved, there is no room for accepting a wrong message. The system has to verify that the data it is processing is correct before taking any potential dangerous or undesirable action. For instance, headlight lamps must not be accidently switched off while driving on a highway at night in order to prevent the driver from being contronted with sudden darkness.

The communication link between the master control unit and the different slave units has to be reliable, noise tolerant but also simple in order to keep within the requirements of the multiplexed system.

To achieve the highest possible data integrity, it is possible, using certain coding schemes, not only to detect errors but also to correct them as well [17]. However, the amount of redundancy introduced into every message is much higher than in error-detecting schemes only.

During data transfer from one unit to another, 3 bits are inserted within each transmitted byte for start of transmission, even parity and framing as shown in figure 4.1-c. This represents 27.27% of the transmitted data that do not convey any information. This unavoidable burden is the cost of maintaining system reliability at an acceptable level.

For this purpose three error detection schemes have been installed on the logical transfer layer of the protocol to insure communication integrity. These schemes are:

1- false start bit

2- parity error

3- framing error

1- Detecting false start bit

Many errors can occur during the communication process due particularly to noise as the vehicle medium is too harsh. Such noise signal looks like a start bit to the receiving unit and is referred to as a false start bit. A way to reduce the occurence frequency of such bits is to sample the line many times and use the majority logic to determine the bit value.

the receiving routine at both master and slave units level samples the line once after initial detection. It is required that this sample must be low for that bit to be considered as an actual start bit, otherwise the communication process is immediately aborted and the receiving unit(s) return(s) to the listen state.
This sample is taken at one half bit duration after initial detection (96/2=48μS), which is quite a 'long' period of time that overrides any noise spike occurence.

2- Even parity error

Parity is a simple error-detecting code. This is a single bit added to each byte which makes the total number of '1' bits even. This type of errors occurs whenever the total number of 1's counted at the receiving unit is found to be odd. However, even parity scheme does not detect double (or modulo 2) error occurences.

3- Framing error

This error-detecting scheme is used to synchronize data transfer from one unit to another and is particularly usefull to detect cases when the communication bus is shorted to ground as this bit must be always high.

IV.5 Failure conditions

IMS protocol incorporates properties which insure operational safety even in presence of errors. it allows to discriminate between reversible errors (message transfer corrupted by noise) and irreversible (permanent) failures such as hardware faults.

The detection of errors in IMS occurs at two levels. The first is local detection by the slave control units of deficient loads such as blown bulb filament, then monitoring data is reported to the driver via the master control unit, and the second is detection of a faulty electronic module by other modules.

IMS provides a powerful mechanism for detecting and isolating faulty slave units. In fact, polling technique allows the master control unit to keep track, in a cyclic manner, of the proper functioning of individual slave units. Any non-answering unit is automatically flagged so that no more messages will be addressed to it. On the other hand, the slave control units are able to detect master bus time-out and consequently enter the fail safe operating mode.

IV.6 Data recovery

In case message broadcasting has been corrupted (by noise), the slave units will abort message reception. All data immediately received prior to message corruption is considered not valid so that no switching action will be taken. Only one slave control unit (SCU #3) is allowed to access the communication bus and transmits a 'nil' message to inform the master control unit that the previous message has been canceled so that a retransmission action is to be taken. In fact any slave unit can be selected to request message retransmission. SCU #3 has been chosen because its physical position is less exposed to crash damage than the other SCUs.

In case message corruption occured during data transfer from slave unit #n to master, the MCU will ignore data received and broadcasts a new message to the next slave unit (SCU #n+1).

To avoid deadlock hang ups or illegal bus state, due particularly to bus shortage to ground or any other hardware or software fault that brings the communication bus to permanent ground potential, the master controller suspends message generation in case where the cumulative number of framing error occurences is greater than the number of slave units present whithin the vehicle. A general fault message is displayed on the dashboard (all SCUs fault indicators are turned ON) while the slave units will detect an illegal identification byte (both destination and origin addresses are equal to 0) thus leading to emergency operation.

CHAPTER FIVE

IMS INTERFACING

- Harness configuration
- System inputs
- System outputs
- Intelligent power switches

IMS Interfacing

V.1 Harness configuration

Figure 5.1 shows the partionning of the multiplex wiring prototype based on the '25L4' minibus. The control inputs are fed directly to the master control unit (MCU) without modification of the existing dashboard control switches:

Headlight switch (main and dip)

Flasher switch (2 positions: right and left)

Horn switch (2 positions: city and urban)

Parking switch

Hazard warning switch

Windscreen washer switch

Windscreen wiper switch right

Windscreen wiper switch left

Ceeling lights switch (driver)

Ceeling lights switch (passengers)

Stop light switch (fly-leads)

Ignition key switch

For cost reduction considerations, smaller sized control switches can be used instead of the existing ones, as switching is performed using logic potential levels only.

When the vehicle is not in use, a main power switch is provided with the multiplexed harness, in order to isolate the vehicle electronics and electrical system from the battery thus saving total system power consumption.

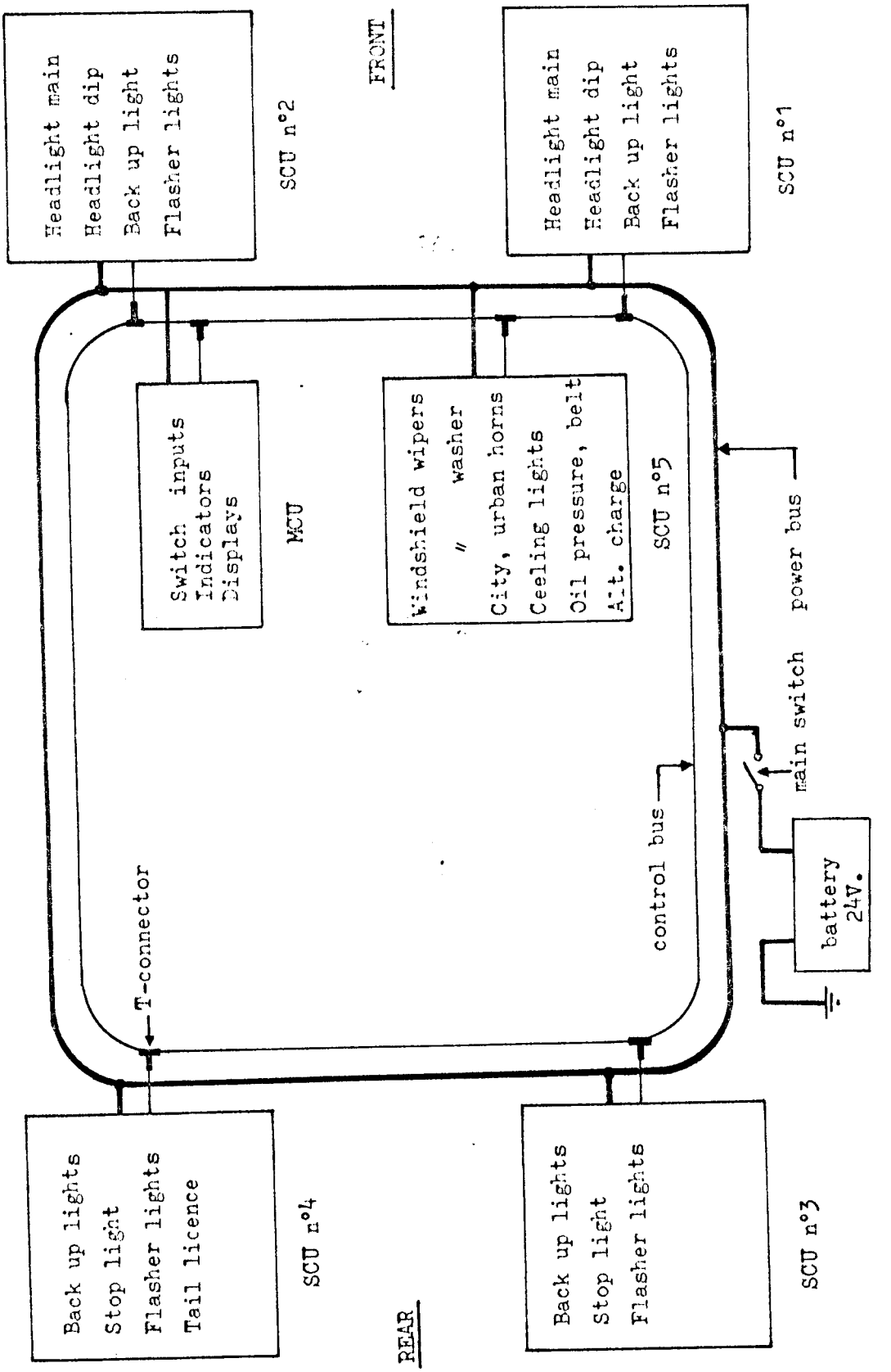
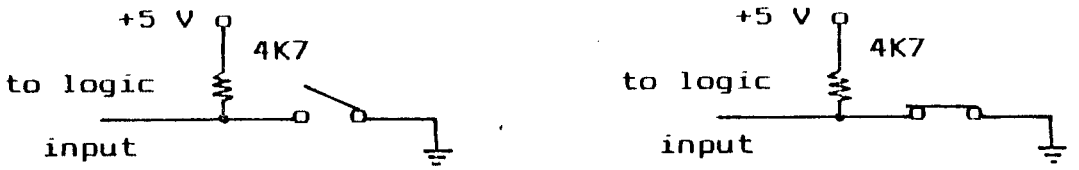


Fig. 5.1 Multiplexed harness configuration.

V.2 System inputs

There are three types of data inputs to the multiplexed system: control switches, analog signals provided by different sensors, and on-off information provided by on-off sensors.

Control switches can be directly connected to the logic inputs of the system. Figure 5.2 shows both positions of a control input switch. A closed switch corresponds to logic '0' whereas an open switch corresponds to logic '1'.



(a) Open switch (b) Closed switch

Fig. 5.2 Control switch interfacing.

Exception is done for flasher (left and right) and hazard switches, see figure 5.3, where the flasher timing signal (1.91 Hz) is inputted to the system through OR gates. A switch closure will cause the output of the OR gate to follow the timing signal, while it will be always high whenever the control switch is open.

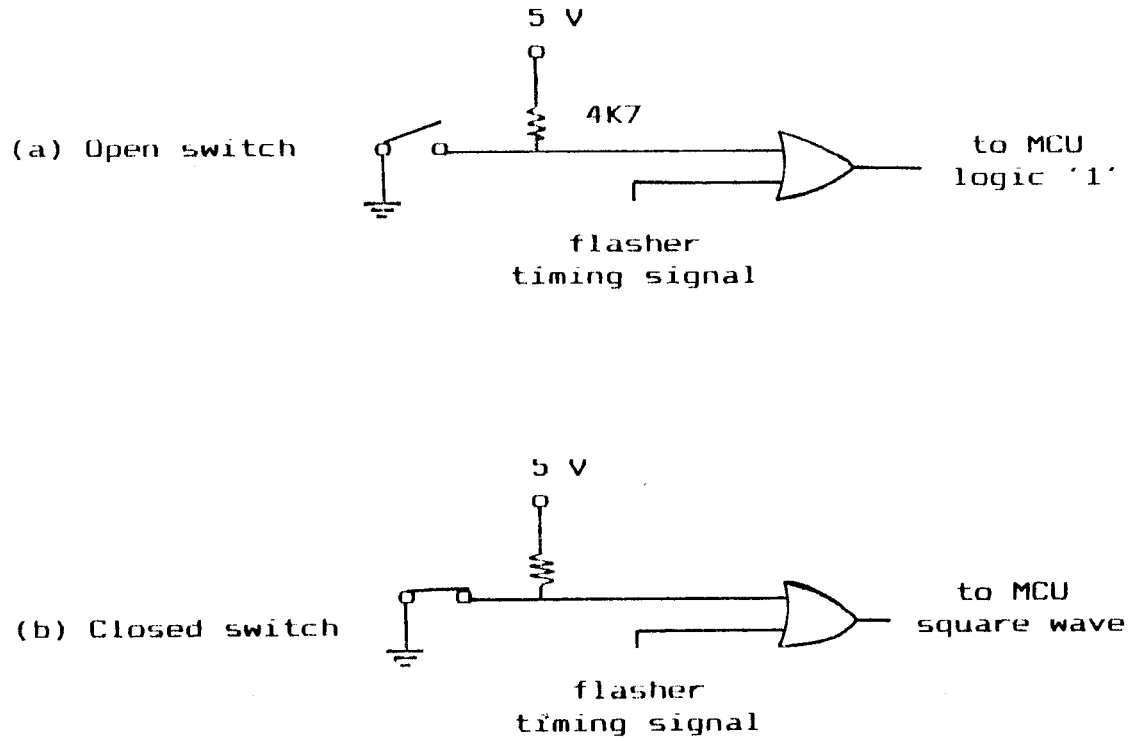


Fig. 5.3 Flasher and hazard switches interfacing.

Analog signals are provided to the system via the slave engine unit (SCU #5). This unit incorporates an 8-bit four-channel anolog-to-digital converter, giving a resolution of $1/2^8$ or approximately 0.5%, which is quite satisfactory for the majority of automotive monitoring applications (fuel level, engine temperature...).

On-off sensors are another source of input signals to the multiplexed system. These are: belt continuity, oil pressure and brake pressure. The output of this type of signals requires only to be down converted from battery level (24 volts) to logic level (5 volts). Figure 5.4 shows that when the ouput of the on-off sensor V_1 is between 15 and 28 volts, the input voltage V_2 to the SCU will be considered as a logic '1'. If the output voltage of the sensor drops below 15 volts, V_2 will be considered as a logic '0'.

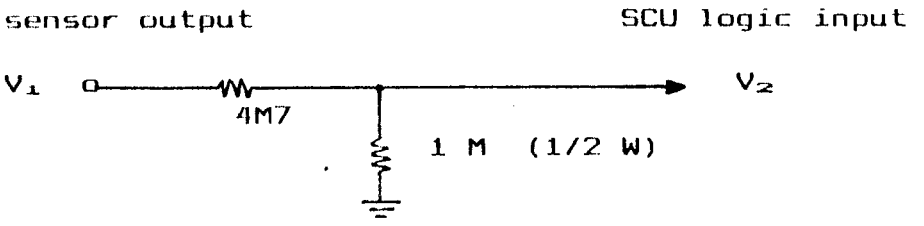


Fig. 5.4 On-off sensor interfacing.

Table 5.1 shows the slave units I/O ports functional allocation. Apart from port lines PA0 and PA1 used for interprocessor communication, the output ports correspond to the SCU driving outputs, while the input ports correspond to the SCU monitoring inputs.

Some of the electrical loads are not monitored as their performance can be directly seen or noticed by the vehicle's driver. Such loads are windshield wiper, windshield washer, city and urban horns which provide either visible or audible signals to the driver.

Function	Slave unit	Output port	Input port
Serial communication	all	PA0	PA1
Headlight main	1,2	PB0	PA2
Headlight dip	1,2	PB1	PA3
Back up light (front)	1,2	PB2	PA4
Back up light (side)	1,2	PB3	PA5
Flasher (turn signal)	1,2	PB4	PA6
Back up light (side)	3,4	PB0	PA2
Back up light (rear)	3,4	PB1	PA3
Stop light	3,4	PB2	PA4
Flasher (side)	3,4	PB3	PA5
Flasher (rear)	3,4	PB4	PA6
Tail licence light	4	PB5	PA7
Windshield wiper (right)	5	PB0	-
Windshield wiper (left)	5	PB1	
Windshield washer	5	PB2	
City horn	5	PB3	-
Urban horn	5	PB4	-
Ceeling lights (driver)	5	PB5	PA2
Ceeling lights (passenger)	5	PB6	PA3
Belt continuity (on-off)	5	-	PA4
Oil pressure (on-off)	5	-	PA5
Brake pressure (on-off)	5	-	PA6

Table 5.1 Slave I/O ports functional allocation

V.3 System outputs

There are two types of system outputs: Driving outputs and display/diagnostic ones.

System driving outputs correspond to the driving I/O port lines of the different slave units around the vehicle. Table 5.2 gives the typical current requirements of the different electrical parts of the vehicle. An in-line fuse is required on each I/O port output line to protect the devices from short circuit hazards.

Load	Current (A)
Headlight main	2.8
Headlight dip	2.6
Flasher	1.2
Back up (front lateral)	0.25
Back up (rear back)	0.35
Tail licence	0.25
Stop light	1.2
Ceeling light	5.6
Windshield washer	0.4
Windshield wiper	2.5
City horn	2.5
Urban horn	2.0

Table 5.2 Typical current requirements of '25L4' minibus electrical equipement.

Figure 5.5 shows the protoype display panel, where all the necessary information is displayed.

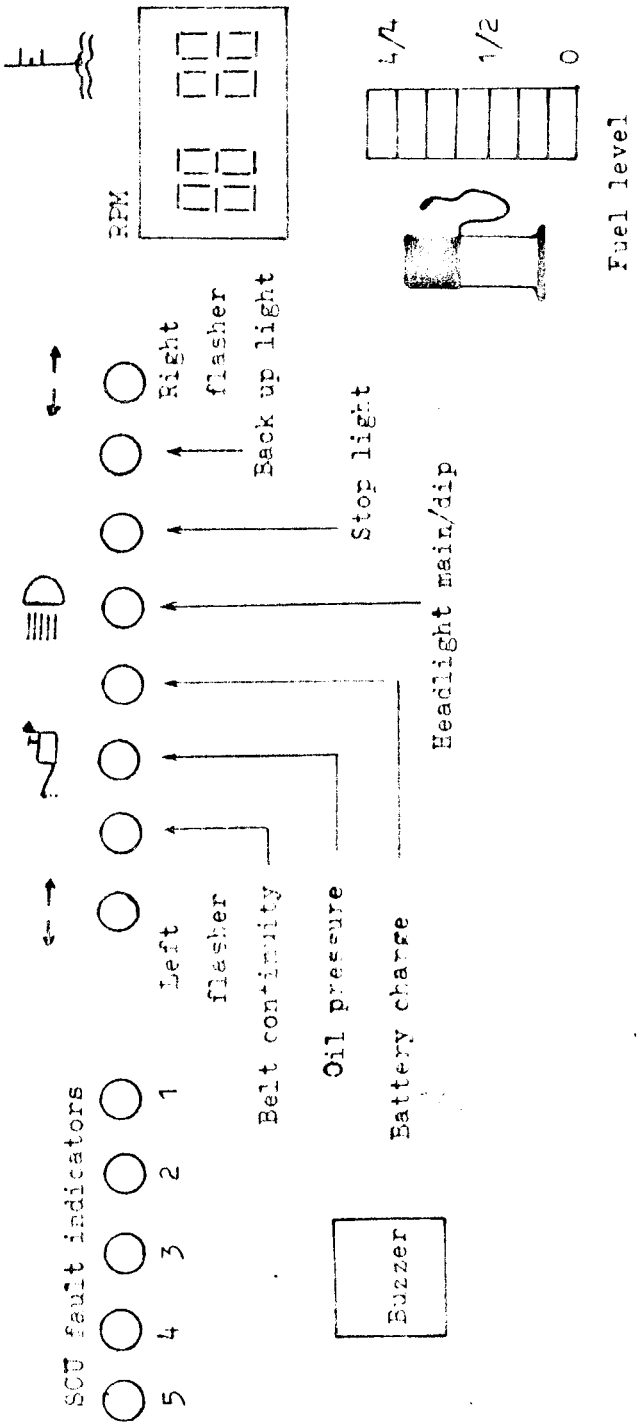


Fig. 5.5 Dashboard displays and indicators.

V.4 Intelligent power switches

Automotive experts agree that all-encompassing multiplex systems will not be practical until power switching issues are resolved. The problem is that power switches at each station or node must incorporate some form of intelligence in order to ease the work load of processors.

Power devices are available (Fig. 5.6) that translate encoded commands, monitor loads, and interact intelligently with other components [18, 19]. but the cost of these devices limits greatly their widespread use.

Texas Instruments Inc., reports that its smart power devices are cheap and moving toward commercial grade prices. Automotive power ICs from TI combine two technologies. A multiepitaxial bipolar process is used to make rugged low-cost high-side drivers for relays, lamps, and solenoids. A power Bidfet IC is optimized for use in full-H drive stepper motors.

Two chips employing the technologies are now available. The TLP401 serial I/O octal driver incorporates power Bidfets. The chip can switch up to 0.5 A at 60 V with an operating frequency of 1 MHz. The TLP440 high-side driver, with multiepitaxial bipolar transistors, can source up to 10 A at 85 V. Both ICs are protected against thermal faults, overcurrent, and battery reversal.

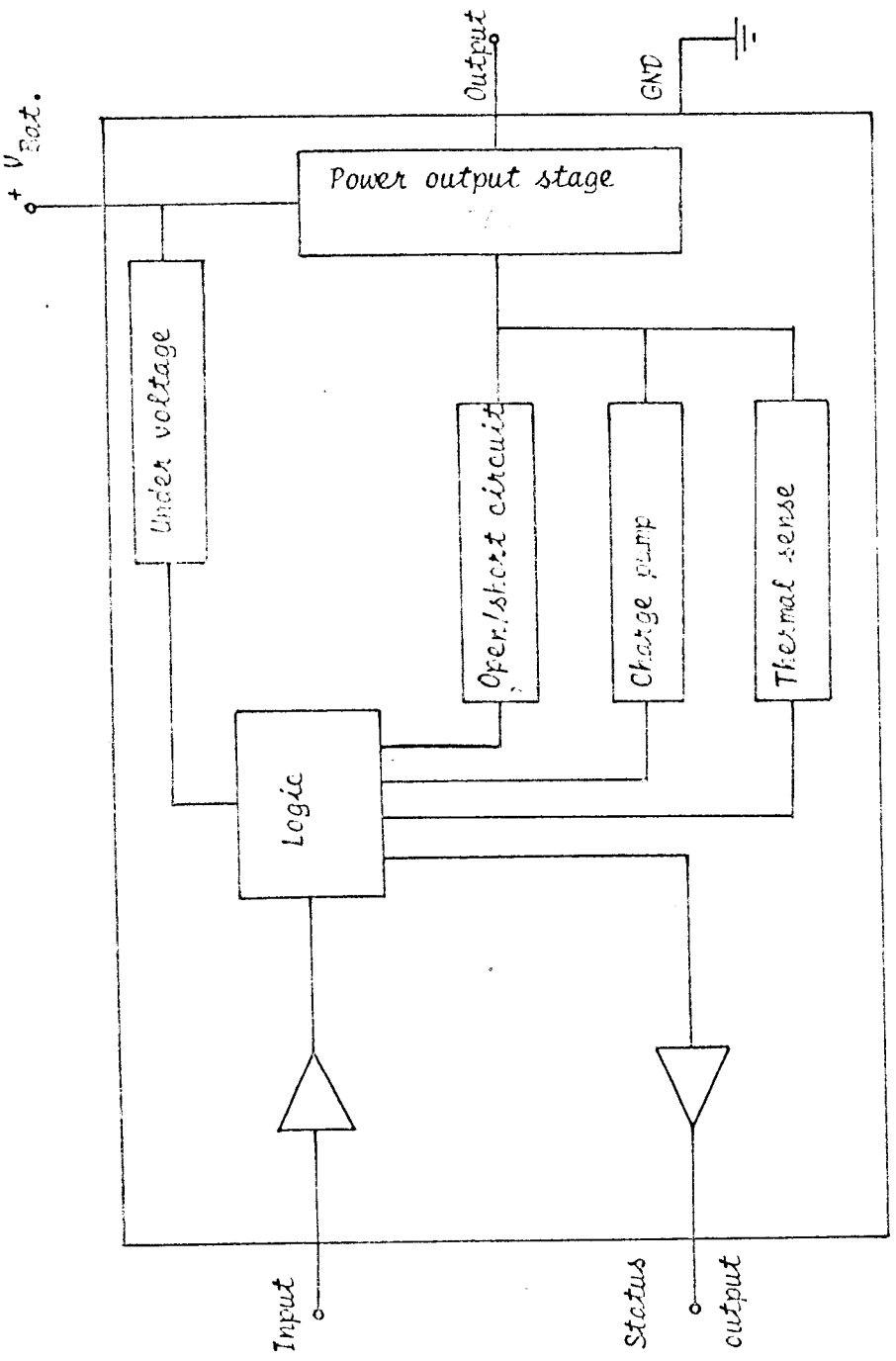


Fig. 5.6 Schematic of an intelligent power switch.

CHAPTER SIX

SYSTEM SOFTWARE

. Software design

. System software

. Master control unit software

. Slave control unit software

System Software

VI.1 Software design

This chapter describes the different subroutines developed within the control programs of the INELEC Multiplexed System. Different criteria have to be fulfilled by the software:

1) **Speed:** as time is an important factor in the system functioning.
2) **Economy:** keeping the code short will reduce the number of EPROMs to be used, and the software can fit in the limited EPROM capacity of single-chip microcomputers.
3) **Reliability:** avoiding deadlocks and endless loops.
4) **Flexibility:** the software can be easily extended or reconfigured to meet changing specifications of other vehicle models.
5) **Ergonomics:** easy to use.

VI.2 System software

The hardware of the master control unit and that of the slave control units being based on different processing units, consequently, software development was carried out on different equipment.

All MCU software was developed in machine language on a Motorola D5BUG-E 6802 kit [20] provided with an hexadecimal keyboard, a 6-digit 7-segment display and a cassette interface. The slave units software was developed in assembly language on a Motorola M68705EV.M [21] provided with a monitor/debugger, a one line assembler/disassembler, a programmer for on-chip EPROM, and a cassette interface.

VI.3 Master control unit software

Figure 6.1 shows the block diagram program structure of the master control unit.

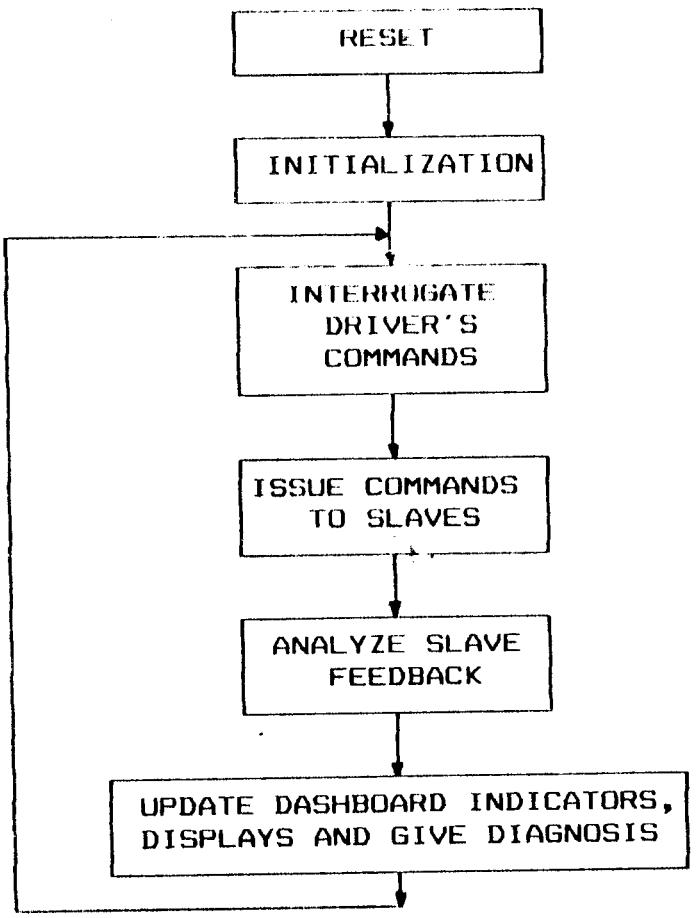


Fig. 6.1 Structure of MCU control program.

In the following material, a discussion of the different subroutines is given.

Initialization

This portion of MCU control program is executed each time a system power-up reset is initiated. The routine clears the system memory (RAM) working space, loads the system stack with hexadecimal value 007F representing top of stack, and initializes all the flags necessary for further CPU functioning. PIA ports A and B are set in the output mode, whereas the dashboard indicators and displays are cleared.

Built slave unit address table

This main program portion determines first the number of slave control units present in the vehicle by reading input switch 3. Command inputs I_{21}, I_{22}, and I_{23} are used to set the number of slave units. The minimum and maximum number values are 1 and 8 respectively. This number is used to decide the size of a table called SLAVTB. SLAVTB is built containing the message identification byte, i.e., the destination and origin addresses. An end-of-table flag ($FF_{HEXADECIMAL}$) is inserted at the end of this table to allow the CPU to recycle command message generation. This feature allows the same basic MCU software to be used for other vehicle models having different numbers of slave control units.

Initialize transmission

Before transmitting any command message, the CPU clears parity counter **ONECNT** and initializes the number of data bits **BITCNT** to be transmitted per word and the command table pointer with their repective appropriate values (see Figure 6.2). **BITCNT** can be reprogrammed to other values (eg. 7 bits) to meet other design constraints. The communication status register **ERRFLG** is also cleared and the interrupt mask bit is set to disable CPU self interruption during transmission.

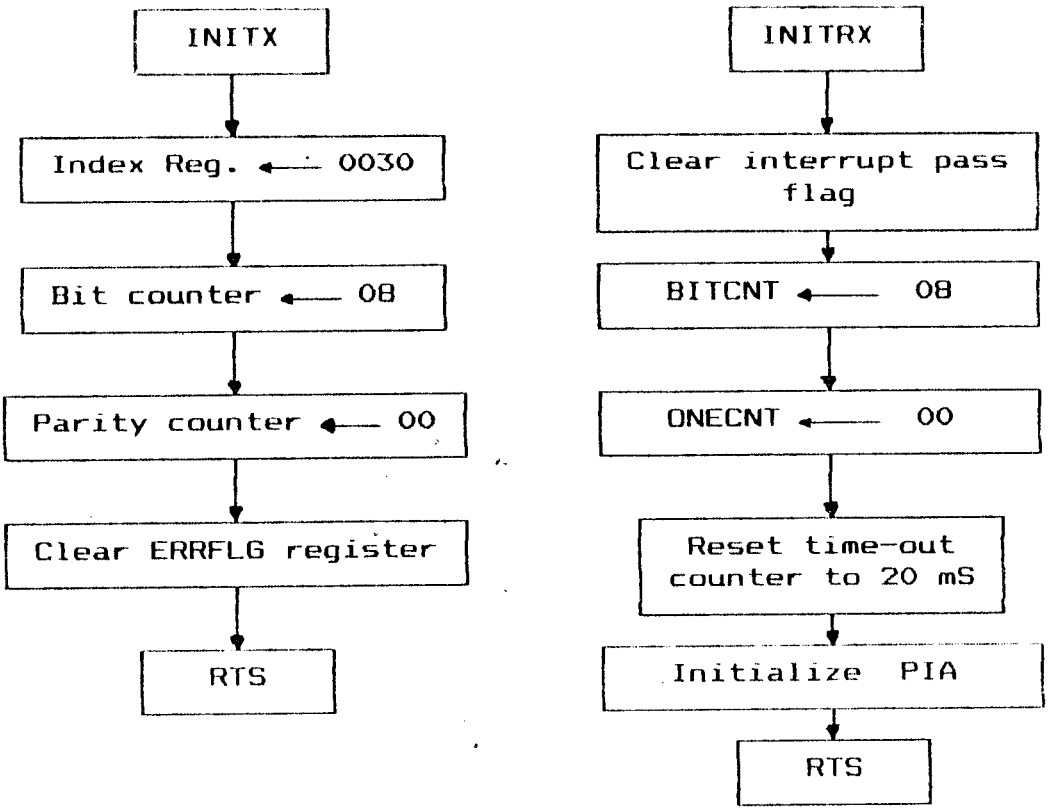


Fig. 6.2 Initialize transmission and reception subroutines.

Initialize reception

After broadcasting a command message to slaves, the master controller is expecting an answer (interrupt) from the addressed slave unit. The CPU has therefore to enable the interrupt by clearing the interrupt mask bit, clear the interrrupt pass flag (INTPAS) and the communication status register ERRFLG, initialize bit and parity counters (BITCNT and ONECNT) and reset the time-out counter TIMCTR to hexadecimal value 021F (=543).

The time-out duration is set to be 20 mS and is computed according to the execution time of the main program portion (see figure 6.3) during which the master is expecting a slave interrupt. In figure 6.3, The number between brackets indicates the total number of machine cycles (MC) required by the corresponding instruction. One machine cycle time (t_{MC}), based on the 3.579 MHz crystal, is:

$$t_{MC} = 1/f_{INTERNAL}$$

$$f_{INTERNAL} = f_{CRYSTAL}/4$$

$$t_{MC} = 4/(3.579 \times 10^{6}) = 1.1176 \times 10^{-6}\ S$$

```
WAIT    NOP                         (2)
        LDX     TIMCTR              (4)
        DEX                         (4)
        BEQ     TIMERR              (4)
        STX     TIMCTR              (5)
        BRA     $+2                 (4)
        TST     INTPAS              (6)
        BEQ     WAIT                (4)
                                    ----
                          total=    33 MC
```

Fig. 6.3 Time-out main program portion.

The slave time-out delay will be:

$$543 \times 33 \times 1.1176 \times 10^{-6}\ S = 20.02\ mS$$

Other time-out delays can be easily achieved by just initializing TIMCTR to the appropriate value (see Table 6.1).

Delay (mS)	Value in TIMCTR HEX.	DECIMAL
05	0088	136
10	0110	272
15	0197	407
20	021F	543
25	02A6	678
50	054C	1356

Table 6.1 Possible time-out delays.

Get command

This subroutine (see figure 6.4) is called each time prior to any message broadcasting, in order to update the command table **CMDTAB**, whose content is to be the newest reading of the three input command switches **SWICH1**, **SWICH2**, and **SWICH3**. As the control inputs are active low, this subroutine complements the input readings so that **CMDTAB** will contain the corresponding active high commands.

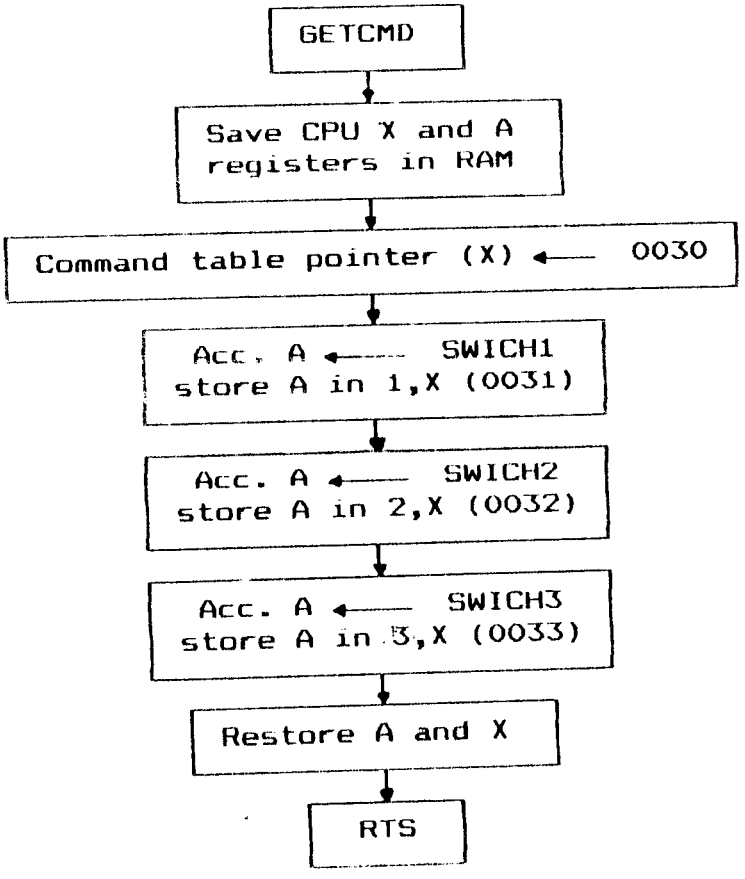


Fig. 6.4 Get command subroutine.

Transmit command

For multiplexed system proper functioning, interprocesor communication is very important if not vital. For this consideration, TRANSMIT and RECEIV subroutines are considered to be the heart of the system's software. The MCU bit timing is based on a 3.579 Mhz crystal giving a basic internal clock frequency of $f_{crystal}/4$ = 894.75 KHz corresponding to a time-period of 1.1176 µS.

Each bit is 96 µS, selected among the class B standard permissible values [22]. This corresponds to 96/1.1176 = 86 machine cycles.

This subroutine permits to the MCU to broadcast messages to slave units. It performs the following functions (see figure 6.5):

- Serialize the words to be transmitted.
- Keep track of the bit timing.
- Generate start, even parity, and stop bits for asynchronous transmission.

TRANSMIT is a tuned-execution time subroutine. Considering the program portion that corresponds to one bit transmission, we have:

```
            ROR     OUTCHR          (6)
AGAIN1      BCC     ZERO            (4)
            JSR     ONEOUT          (11)+(10)       *
            INC     ONECNT          (6)             *
            BRA     CONT            (4)             *
ZERO        JSR     ZEROUT          (11)+(10)       **
            TST     $               (6)             **
            BRA     CONT            (4)             **
CONT        LDA     B       #01     (2)
            JSR     DLYB            (29)
            BRA     $+2             (4)
            DEC     BITCNT          (6)
            BNE     AGAIN1          (4)
                                    ----------
                                    total = 86 MC
```

* : path for "1" only.
** : path for "0" only.

The total number of machine cycles per bit is always the same (86 MC), whether the transmitted bit is a "1" or a "0". In fact, dummy instructions are inserted in appropriate places to nil draw the time gap required by either of the paths to perform operations not available in the other, such as incrementing the number of 1'S counter (ONECNT) for parity.

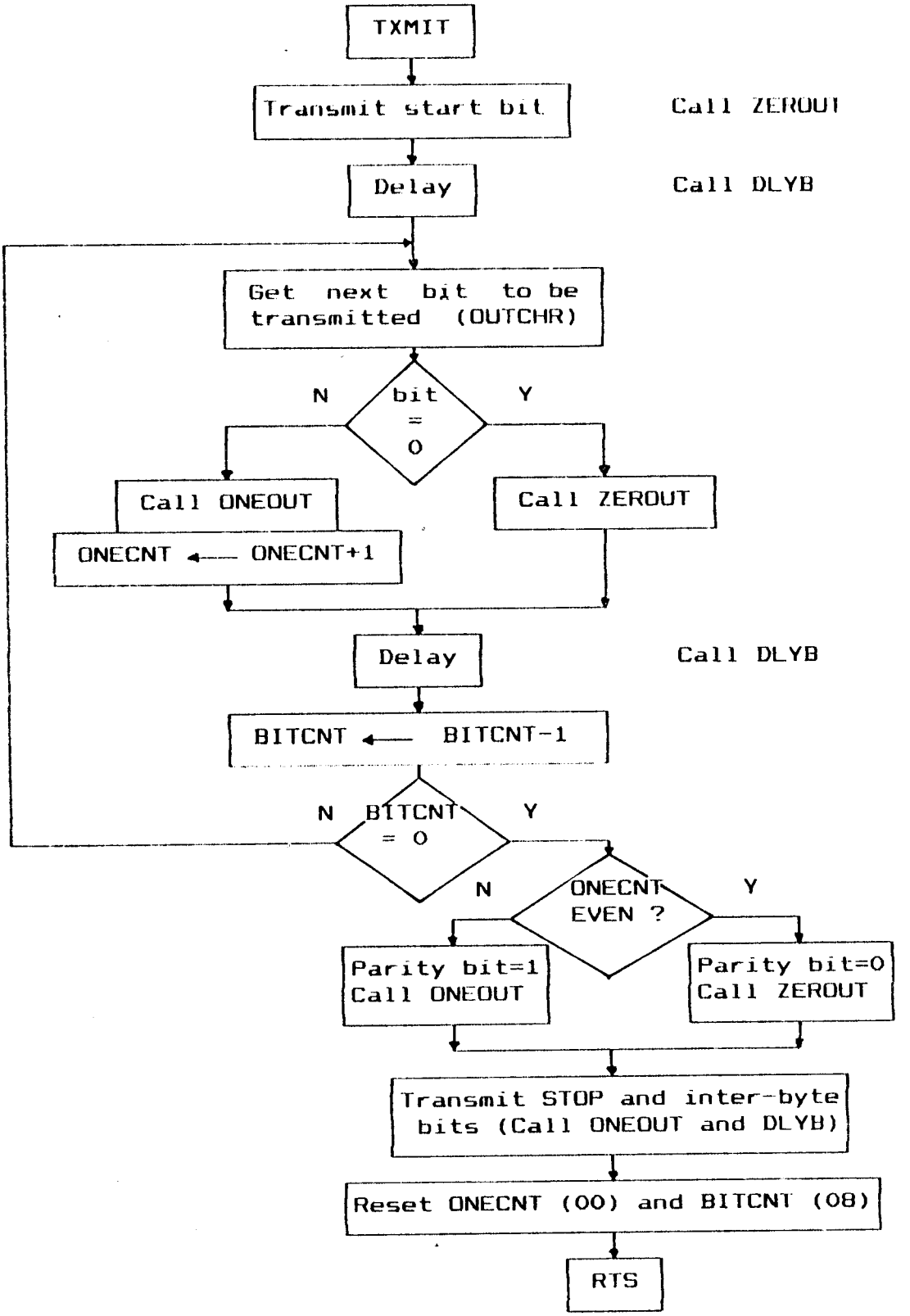


Fig. 6.5 Transmit data subroutine.

Interrupt routine

When the CPU interrupt is enabled, this subroutine is executed on the first high-to-low transition of the CPU interrupt (INT) line. This transition may be caused by either of two things: a noise burst or a start bit of a message stream emanating from one of the slave control units.

INTERRUPT subroutine (see Fig. 6.6) will first disable CPU interrupt so that subsequent high-to-low transitions will not cause other unwanted interrupts that may disturb the communication process. Control is then passed to RECEIV subroutine.

For successful reception, it is important to take into account the number of machine cycles taken by the CPU to acknowledge an interrupt. The average CPU time taken to acknowledge an interrupt is 11 machine cycles (MC) [23]. Adding 29 MC taken by INTERRUPT subroutine before calling RECEIV (see extract from program listing below), we end up with a total of 40 MC (11+29) or 40x1.1176 = 47.04 µS, i.e., half a bit time (±1 µS), which is the middle of the input bit where it will be sampled.

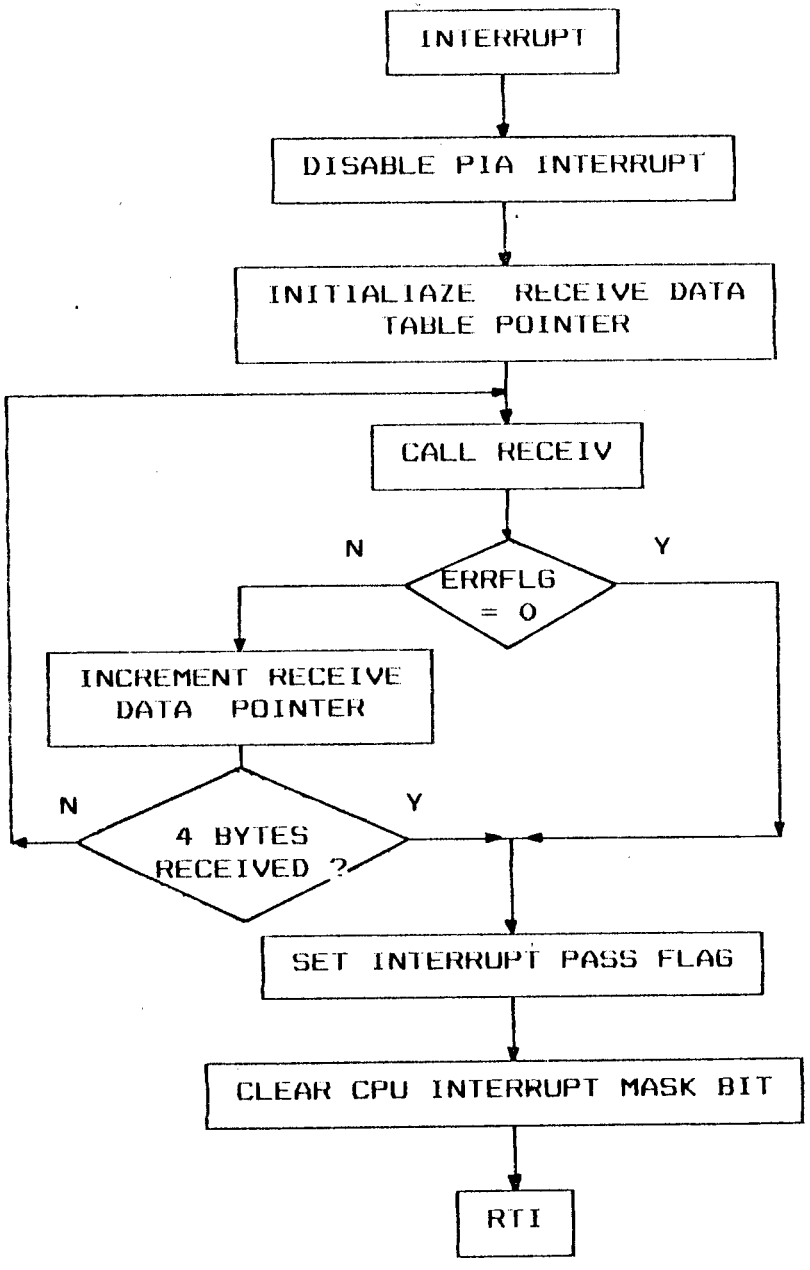


Fig. 6.6 Interrupt routine.

Below is an extract from the INTERRUPT subroutine:

```
NOP                          (2)
NOP                          (2)
TST     $                    (6)
LDA     A       #04          (2)
STA     A       PIACRA       (5)
LDX     #0035H               (3)
JSR     RECEIV               (9)
                             ----
                             total = 29 MC
```

Receive data

Having checked that the first bit received is an actual start bit, RECEIV (see Fig. 6.7) will proceed sampling the input data stream at a time interval of 86 MC, i.e., all the received bits will be sampled at their middle. This increases data communication integrity.

If the middle sample of the first received bit has been found to be high (i.e., false start bit), RECEIV will set the corresponding flag in the communication status register (bit b_0 of ERRFLG) and abort reception process immediately so that control returns to the main program. Otherwise, reception process continues with a check of both parity and framing errors.

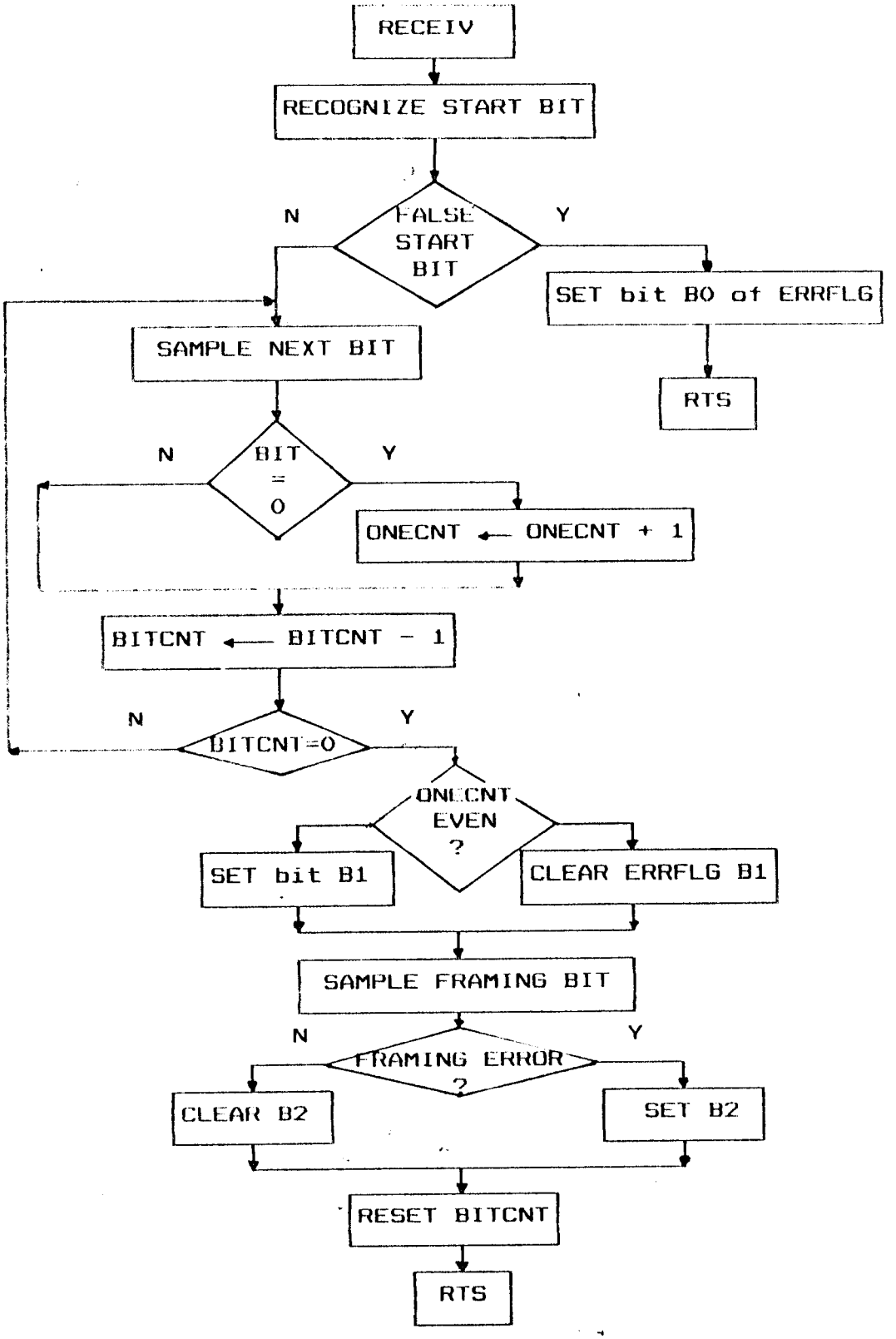


Fig. 6.7 Receive data subroutine.

Bus time-out

In case an SCU fails to respond to a control packet, the MCU will detect a bus time-out after a time delay of 20 mS. The corresponding identification byte, consisting of destination and origin addresses, of the failed SCU is replaced by a flag (hexadecimal value F0), so as the MCU will no more address messages to this destination. The failed SCU code number is also appended to a diagnostic table called DIAGTIC, needed by subsequent program subroutines.

Slave feedback analysis

If the addressed SCU responds to the MCU, the latter has first to check message integrity before taking any action.

The communication status register ERRFLG content is checked for different probable occuring errors as tabulated below:

error	ERRFLG bits B2 B1 B0
false start bit	B0 = 1
parity error	B1 = 1
framing error	B2 = 1

Table 6.1 Communication status register.

If a false start bit has occured, it will be merely ignored, and the MCU returns back to the listen state, waiting an interrupt from one of the slave control units.
On the other hand, if a parity or framing error were detected, the received data will not be taken into consideration, and the MCU proceeds to broadcast a new control packet.

A deadlock counter LOCK is set to count the number of framing error occurences in order to detect, eventually, permanent illegal bus state such as bus shortage to ground. An illegal bus state is assumed whenever the content of LOCK is greater than the number of SCUs present within the vehicle. LOCK is cleared each time a correct message is received.

Code conversion subroutine

This subroutine (CONVERT) is used to translate the faulty slave unit code numbers, present in DIAGTIC table, to another form which is more suitable for vehicle driver information. Suppose we are in presence of two faulty SCUs #01 and #04. CONVERT will process these codes to get a combined code that will be used, in a staightforward manner, by DISPLAY routine to switch on the corresponding dashboard indicators.

Dashboard indicators

The dashboard subroutine (DASHBRD) is considered as the MCU data dispatching center. After a correct message reception, this subroutine is called by the main program prior to update the dashboard displays and indicators. DASHBRD checks for nil message eventually sent by SCU #03, then proceeds dispatching different diagnostic data reported by the SCU.

Display routine

Before recycling message brodcasting, DISPLAY is called to update the different visual and audible dashboard indicators and displays, such as, flasher, headlight, brake, buzzer, fuel level, as well other diagnostic indicators.

VI.4 Slave control unit software

Figure 6.8 gives the main program structure of the slave control units.

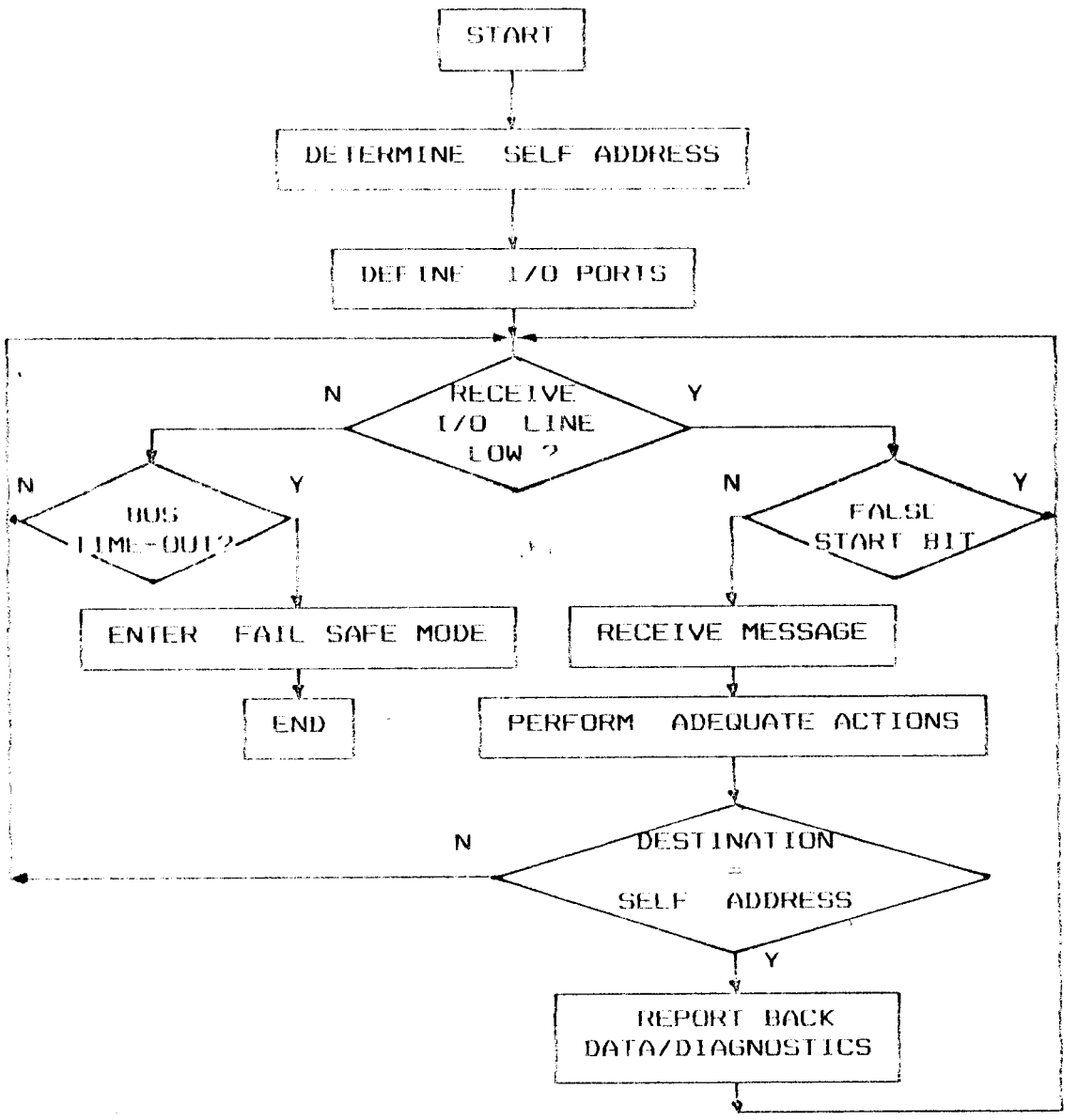


Fig. 6.8 SCU program structure.

Initialization

After a system power up reset, this program segment (see Fig. 6.9) is executed to prepare the SCU to receive control commands from the master. The power up time-out is made to be 47 mS, which is almost twice the normal bus time-out (25 mS), in order to give more time to the MCU to achieve all the necessary initialization operations.

The SCU starts by clearing all the I/O port lines, so that all the loads are in the off state. Afterwards, the SCU determines its address by reading I/O port lines PC_0, PC_1 and PC_2, and defines all I/O ports accordingly.

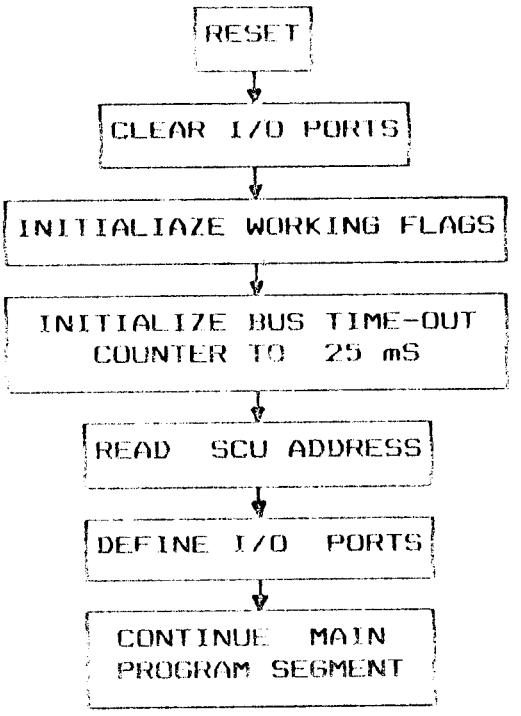


Fig. 6.9 Flowchart of SCU initialization segment.

The main program segment continues by polling the serial data input line PA_1. The SCU is now capable of collecting data and processing commands emanating from the master controller.

Receive data subroutine

A 4.0 MHz crystal is used as SCU bit timing reference. The internal operating frequency is $F_{crystal}/4$, or 1.0 MHz. The SCU has a fixed machine cycle (MC) equal to 1 µS.

The SCU receive data subroutine **RXVDAT** (see Fig. 6.10) is, unlike MCU, initiated by polling the communication bus instead of an interrupt. Reception process starts whenever there is a high-to-low transition on the input port line PA_1. Once a start bit has been recognized, the data stream is sampled at time intervals of 1-bit duration, i.e., 96 µS (96 MC), until the entire 4-byte control packet is received.

This tuned-execution time subroutine resides in 111 bytes and has the feature of being relocatable. Below is an excerpt of **RXVDAT** listing.

ADDRESS	OBJECT CODE	MNEMONIC		(MC)	
0257	03 00 00	BRCLR	1,$00,$025A	(10)	
025A	24 04	BCC	$260	(4)	
025C	3C 14	INC	$14	(6)	*
025E	20 04	BRA	$0264	(4)	*
0260	3D 14	TST	$14	(6)	**
0262	20 00	BRA	$0264	(4)	**
0264	36 12	ROR	$12	(6)	
0266	A6 04	LDA	#$04	(2)	
0268	AD 35	BSR	$029F	(54)	
026A	3A 10	DEC	$10	(6)	
026C	26 E9	BNE	$0257	(4)	
			TOTAL =	96 MC	

*: Path when a "1" is received.

**: Path when a "0" is received.

A variable delay subroutine is provided at hexadecimal address $029F, where the delay is equal to: content of register (A) x 10 + 14. So, in order to obtain the (54) MC delay, we have to put $04 (see line 0266) in register A, so that we will have 4 x 10 + 14 = 54 MC.

029F	4A	DECA		(4)
02A0	9D	NOP		(2)
02A1	26 FC	BNE	$029F	(4)
02A3	81	RTS		(6)

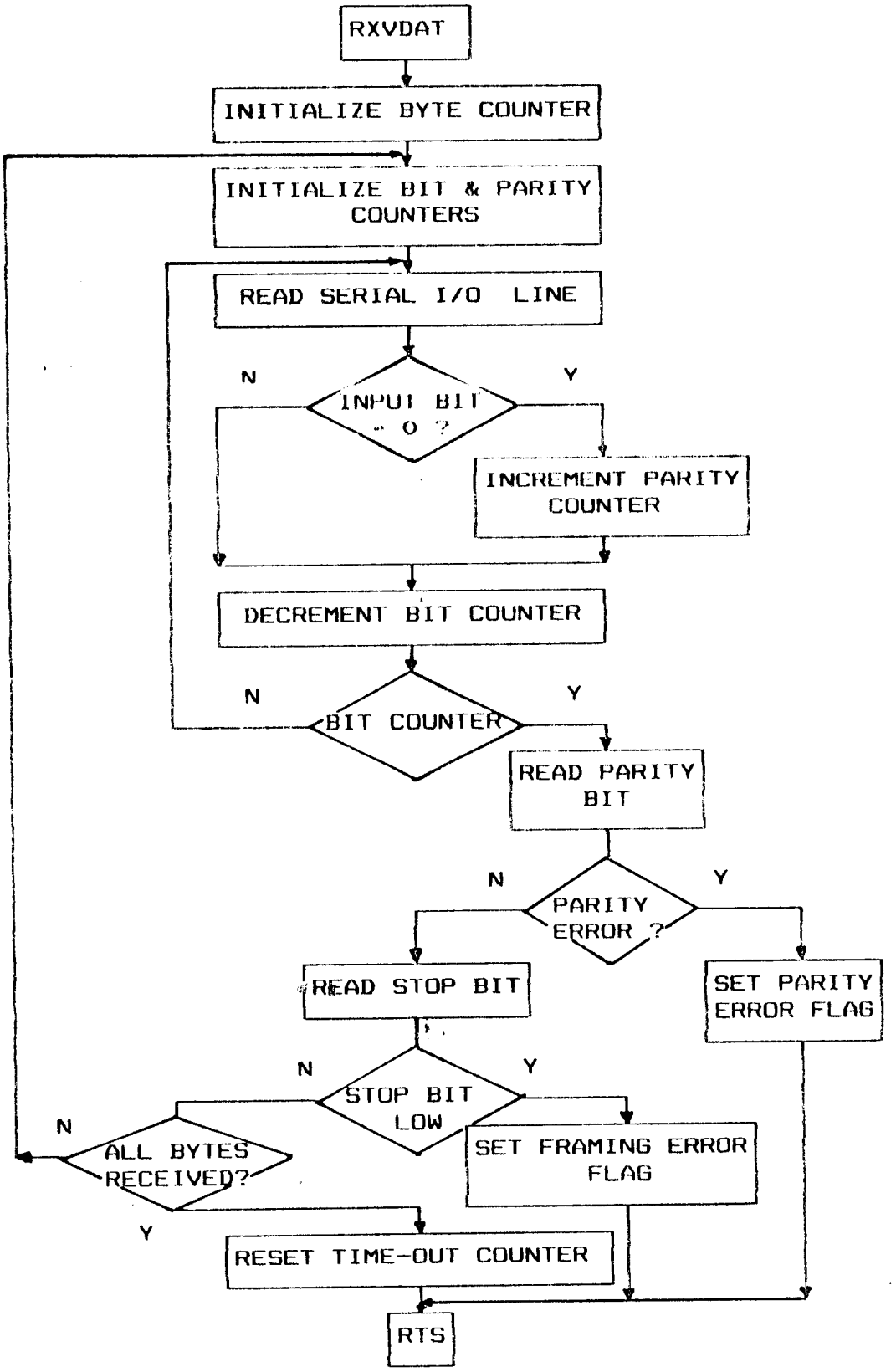


Fig. 6.10 Flowchart of SCU receive data subroutine.

Transmit data subroutine

The serial transmit data subroutine **TXMDAT** (Fig. 6.11) is similar to that of the MCU, except that the number of machine cycles per transmitted bit is 96 as 1 machine cycle (MC) is equal to 1 µS. Here again, there are two possible paths, see excerpt from program listing below, depending on whether the transmitted data is zero or one.

ADDRESS	OBJECT CODE	MNEMONIC		(MC)	
020A	AD 35	BSR	$241	(8/13)	*
020C	3C 14	INC	$14	(6)	*
020E	20 06	BRA	$0216	(4)	*
0210	AD 32	BSR	$0244	(8/13)	**
0212	3D 14	TST	$14	(6)	**
0214	20 00	BRA	$0216	(4)	**
0216	A6 02	LDA	#$02	(2)	
0218	AD 2D	BSR	$0247	(43)	
021A	3A 10	DEC	$10	(6)	
021C	26 E8	BNE	$0206	(4)	
021E	36 14	ROR	$14	(6)	
0220	24 04	BCC	$0226	(4)	
				TOTAL = 96 MC	

*: Path for "1" only.

**: Path for "0" only.

As shown below, $0241 and $0244 are the addresses of the two subroutines called to transmit a "1" and a "0" respectively. Another variable delay subroutine is provided at address $0247, where the delay is equal to: (A) x 10 + 23 MC.

0241	10	00	BSET	0,$00	(7)
0243	81		RTS		(6)
0244	11	00	BCLR	0,$00	(7)
0246	81		RTS		(6)
0247	B7	17	STA	$17	(5)
0249	4A		DECA		(4)
024A	9D		NOP		(2)
024B	26	FC	BNE	$0249	(4)
024D	B6	17	LDA	$17	(4)
024F	81		RTS		(6)

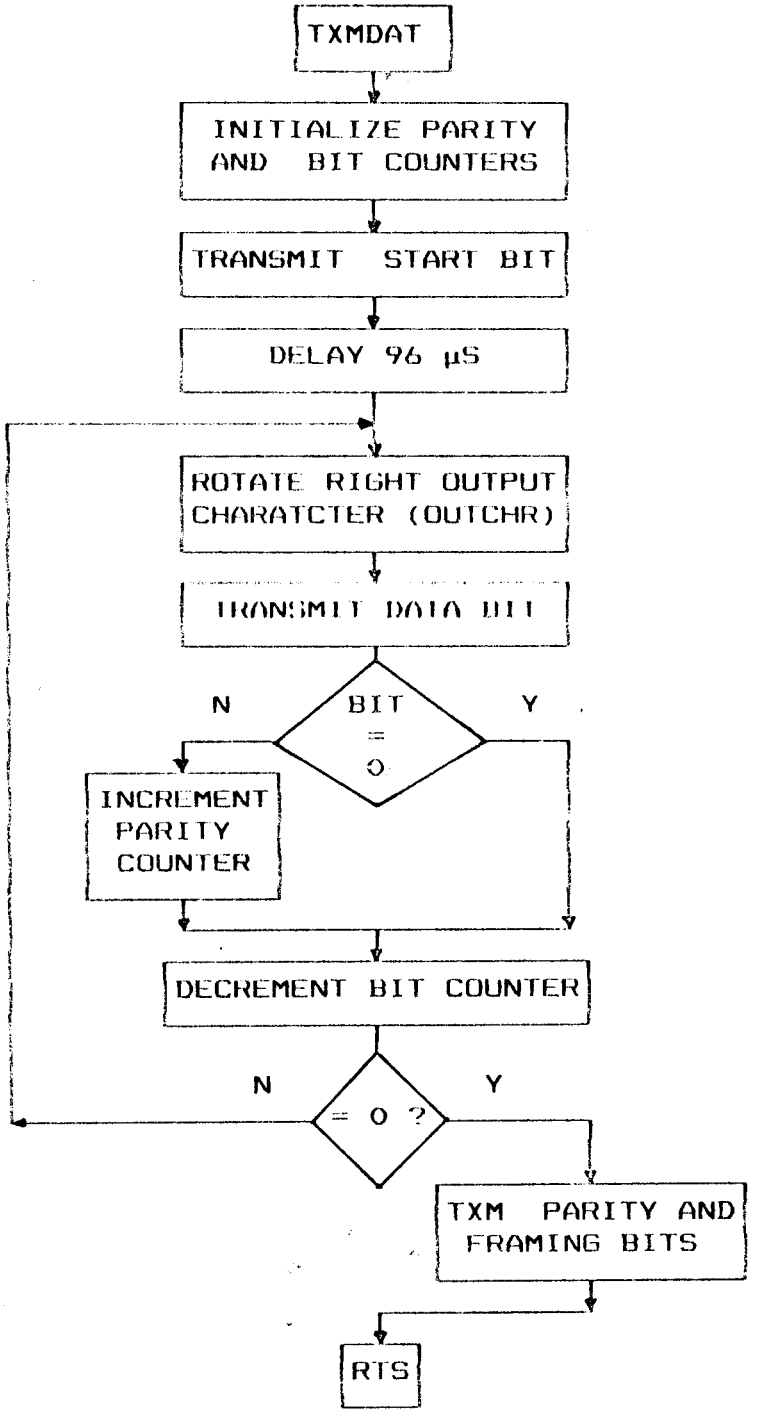


Fig. 6.11 Flowchart of SCU transmit data subroutine.

Data acquisition

This subroutine, see figure 6.12, consists of two parts: one dealing with analog signals, which is executed by the engine slave unit only (SCU #05), and the other part dealing with monitoring data of the driven loads.

Analog data is read via a four-channel analog-to-digital converter incorporated in the engine SCU. Three signals are considered: fuel level, RPM, and motor temperature. Equivalent digital values are processed and stored in memory locations 0032 ad 0033 for transmission.

Monitoring data is read via I/O port lines PA_2 thru PA_7 and is stored in memory location 0031. A "1" represents faulty load condition while a "0" represents normal load operation.

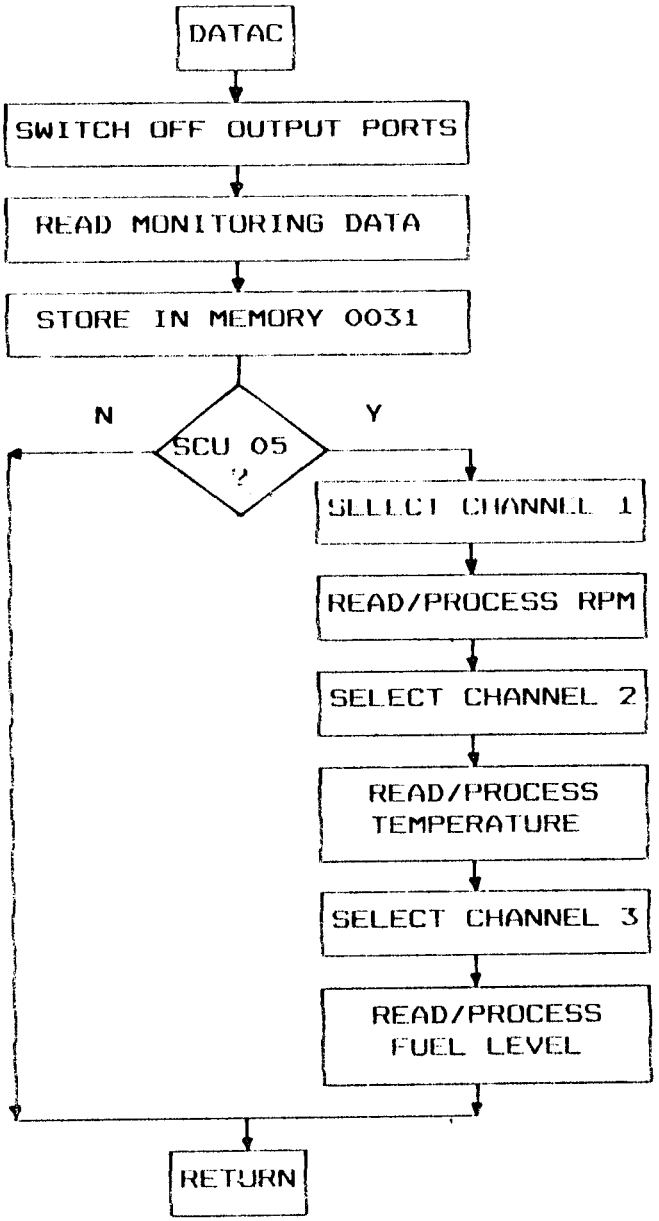


Fig. 6.12 Data acquisition subroutine.

Fail safe operation

This program segment is executed, whenever the SCU detects a bus time-out or an illegal bus state, in order to maintain a minimum set of electrical functions.

The bus time-out is detected whenever the MCU does not broadcast any message within a time-slot of 25 mS during normal operation or 47 mS after a power-up reset. A bus time-out counter is reset every time a message is received by the SCU to avoid false bus time-out detection. The following functions are maintained in the fail safe operating mode:

- Headlight dip
- Hazard light
- Back up lights.

CHAPTER SEVEN

CONCLUSION

. Evaluation

. Conclusion and further work

Conclusion

VII.1 Evaluation

IMS prototype has been installed and tested on a '25L4' minibus electrical harness provided by the National Company of Industrial Vehicles (SNVI-CVI). The prototype consists of one MCU and five SCUs with about 120 I/O signal lines connected to the system.

Figure 7.1 gives a comparison of the total length of conductors used in the conventional and the multiplexed harness respectively. IMS harness length represents about 17% of the total length of the conventional harness.

- System flexibility: within an upper limit of 8 slave control units, no change is necessary in the system protocol of communication in case of addition or deletion of an SCU.

- Low latency time: as required by SAE J-1850 class B standard, latency time varies between 5 and 50 mS.

- Adaptability: IMS protocol can expand easily to other vehicle models with alleviated labor of model development.

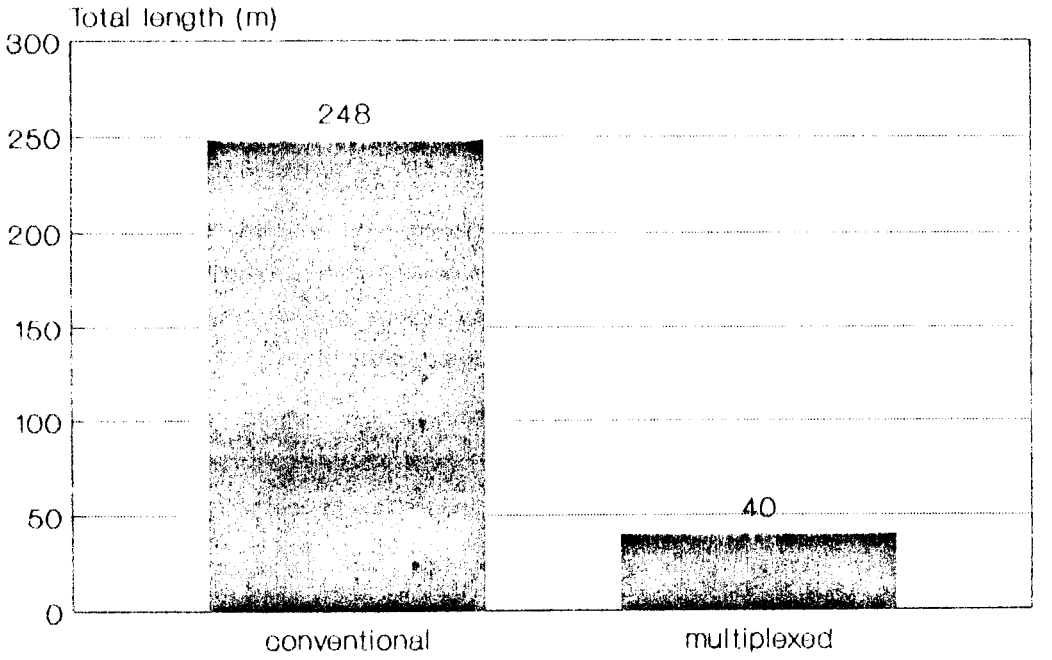


Fig. 7.1 Comparison of conventional and multiplexed harness lengths.

- **Economy:** using the existing popular CPU, MC6802, will keep total system electronics price at commercial grade level. On retail price basis, The current price of IMS prototype is 900.00 DA.

- **Power consumption:** typical IMS prototype electronics power consumption is about 5 watts. This value can be greatly reduced it using CMOS devices.

VII.2 Conclusion and further work

Efforts will undoubtly continue for the foreseeable future to develop techniques to overcome vehicle wiring congestion problem.

In this research work, we developed a microprocessor-based system to reduce vehicle harness and introduced monitoring features that allow the vehicle user to keep track of the proper functioning of the different vehicle parts.

The system protocol offers implementation flexibility and ease of assembly and allows sharing of parametric data between the different processors, which is an important feature for future development.

For off-board diagnosis, an external computer can be connected to the system via a standard RS232 interface.

IMS hardware will be much more simplified if some components were made available during the time of project development. Using the MAX 232 [24] single supply EIA level converter, will considerably reduce the power supply module hardware. The introduction of intelligent power switching devices will improve the hardware reliability as well as reduce the total number of components used per system.

Having at hand the electronic components mentioned above, another prototype can effectively be implemented and tested in actual vehicle environmental conditions.

IMS protocol can also be used in other local area control applications, such as fire detection and monitoring systems. Future candidate application fields of multiplex technology include:

- aircraft industry,
- ships
- lifts, elevators and cranes.

Finally, I would like to mention that, as far as our knowledge is concerned, this is the first project dealing with multiplex technology to be developed at the 'Magister' level throughout the whole country and the Arab World. I hope it will be the first effective contribution to the building of a modern Algerian Car Industry.

REFERENCES

References

[1] Electronics and Wireless World, Vol. 95, Nº 1641, July, 1989.

[2] William M. Ford, "A system to evaluate communication protocols for automotive multiplex", United Technologies Automotive, IEEE Transactions on Industrial Electronics, Vol. IE-30, Nº 2, May 1983, pp. 155-159.

[3] T. Inoue et al., "Protocol for Automotive Local Area Network (PALNET) - A newly Developed In-Vehicle Communication System Based on SAE J1850", SAE paper 890535, pp. 95-109.

[4] Lawrence A. Berardinis "Automakers Move to Multiplexing", Machine Design Vol. 61 Nº 11 June 8, 1989, pp. 102-107

[5] Esmer G. P., Miesterfeld F.O.R., Smisek R.R., "Recommended Practice for Class B Automotive Network" SAE paper 871554.

[6] ... Proceedings of 2ème Symposium PROMETHEUS du 1er Decembre 1987.

[7] SAE paper J1211 Recommended environmental practices for electronic equipment design.

[8] W. Knittel, "Environmental Requirements for Automotive Electronics", IEE Conference Publication, 181, "Automotive Electronics", November 1979, pp. 205-209.

[9] Brian W. Marsden Communication Network Protocols, Chartwell-Bratt (Publishing and Training Ltd.), 1985, England.

[10] M. S. Sloman, "Communications for Distributed Control", Computer Control of Industrial Processes, IEE Control Engineering Series 21, 1984, pp. 118-120.

[11] M.P.U. Data Book, Motorola Inc., 1982.

[12] Single-chip Microcomputer Data 1984/85, Motorola Inc., 1985, reprinted 1987, pp. 3.859-3.882

[13] Single-chip Microcomputer Data 1984/85, Motorola Inc., 1985, reprinted 1987, pp. 3.298-3.389

[14] 1987 D.A.T.A. Book, Power Semiconductor, Edition 21, Electronic Information Series, Vol. 31 Book 50, December 1986.

[15] Data Communication Handbook, Signetics Corp., MOS Microprocessor Division, 1981.

[16] Institute of Electrical and Electronic Engineers, IEEE project 802, Local Area Network Standards, Draft D, November 1982, IEEE Computer Society, Long Beach, Calif.

[17] Richard W. Hamming, Coding and Information Theory, Prentice Hall, 1986 (second edition), Englewood Cliffs, New Jersey NJ07632, pp. 34-50

[18] INTEGRATION, Texas Instruments Ltd., Manton Lane, Bedford, MK41 7PA, 1988.

[19] B. Saby, B. C. Nadd, "Smart Power Devices in Automotive Multiplex Wiring Systems", 6th International Conference on Automotive Electronics, Conference Publication Nº 280, 1987, pp. 195-199.

[20] MEK6802-D5E Microcomputer Evaluation Board, User's Manual, Motorola Inc., 1980.

[21] M68705EV.M Evaluation Module User's Manual, Motorola Inc., 1982.

[22] Class B Data Communication Network Interface, SAE paper J1850, November 1988.

[23] M.P.U. Data Book, Motorola Inc., 1982.

[24] R.S. Catalogue, 1990.

APPENDICES

Appendix A

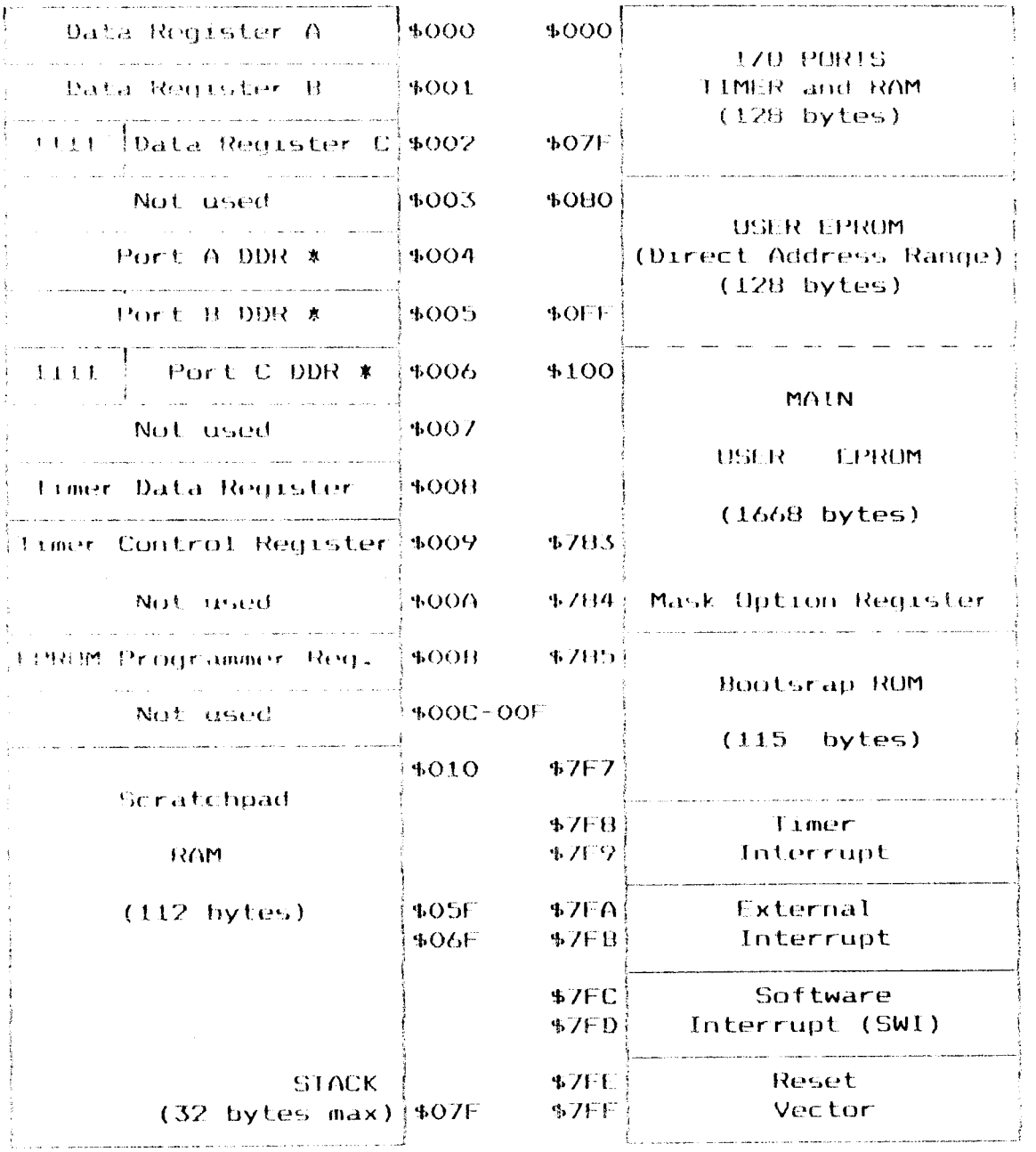
In common with most microprocessors, the 6805 MCU sees memory and interface registers hung on its (internal) data bus as addressable locations.

In the MCU device, these registers are an integral part of the chip, and it seems sensible that this should be the case for the address decoder. The resulting memory map is thus frozen into silicon.

The memory map for the 68705P3 is shown in figure APP-1. The CPU sees 1804 bytes of user addressable storage. Of this, only the first 128 bytes are alterable by software (RAM); the remaining 1796 bytes are for fixed storage (ROM).

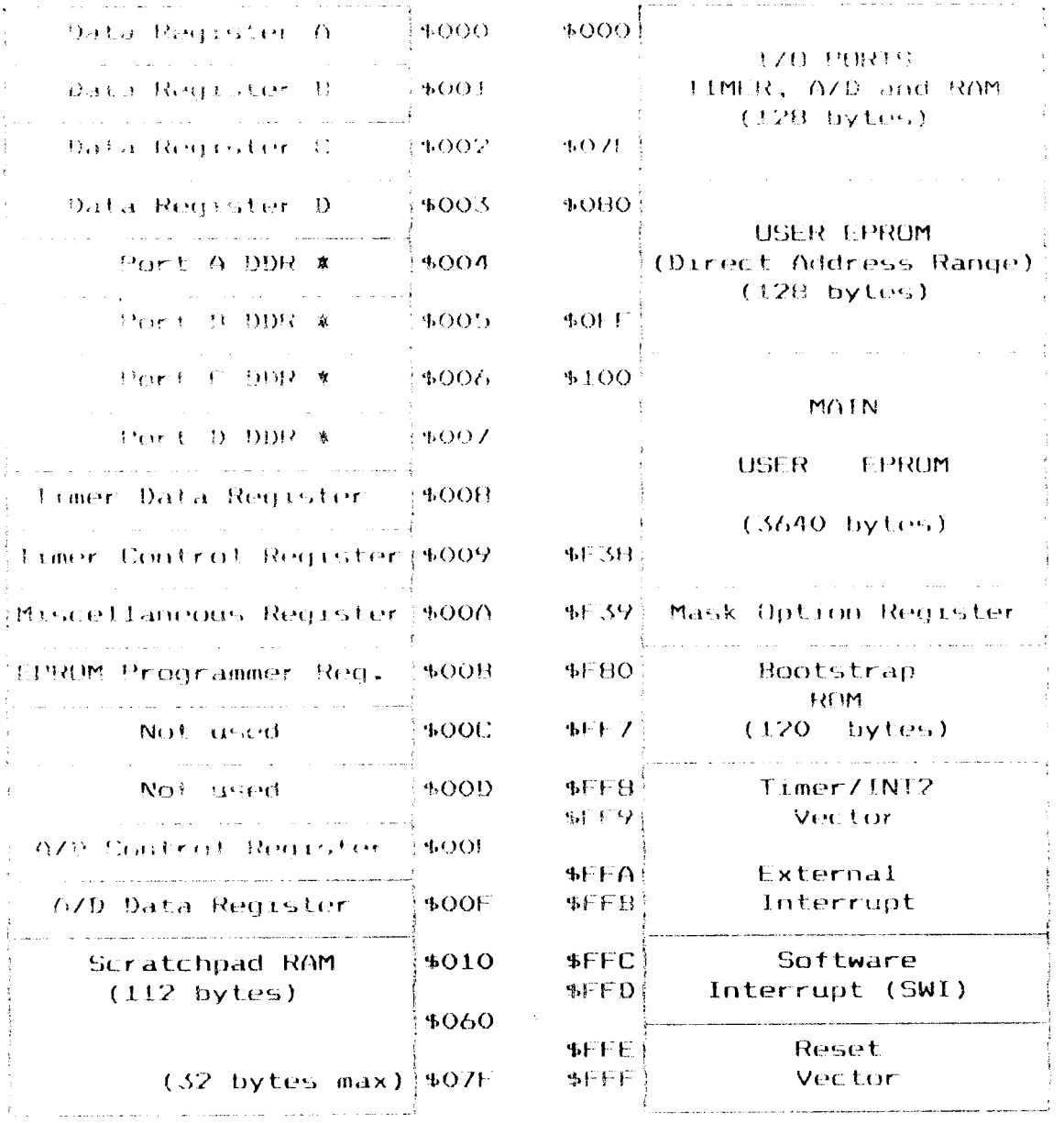
Locations $010-$07F are normal read/write RAM, but the top 32 bytes of this are generally used for stack as regulated by the 5-bit stack pointer.

Figure APP-2 shows the memory map for the 68705R3, where we can notice the presence of on-chip analog-to-digital converter, and the availability of more memory space.



(*) Data Direction Registers (DDRs)
are write only, they read as $FF.

Fig. APP-1 Memory map for the 68705P3.



(*) DDRs are write-only.

Fig. APP-2 Memory map for the 68705R3.

```
Tektronix        M6800 Asm V3.3                              1

00001     0000     BITCNT   EQU    0000
00002     0001     OUTCHR   EQU    0001
00003     0002     INCHAR   EQU    0002
00004     0003     LIMIT    EQU    0003
00005     0004     DRECNT   EQU    0004
00006     0005     INTPAS   EQU    0005
00007     0006     BSAV     EQU    0006
00008     0007     ERRFLG   EQU    0007
00009     0008     SLAVE    EQU    0008
00010     0009     LOCK     EQU    0009
00011     000A     DISPL    EQU    000AH
00012     000B     DATAL1   EQU    000BH
00013     000C     DATAL2   EQU    000CH
00014     000D     FUELEV   EQU    000DH
00015     000E     RFMEMO   EQU    000EH
00016     000F     FLASH    EQU    000FH
00017     0010     DIAGTIC  EQU    0010H
00018     0020     SLVPTR   EQU    0020H
00019     0022     TIMCTR   EQU    0022H
00020     0024     SCRACH   EQU    0024H
00021     0030     CMDTAB   EQU    0030H
00022     0050     SLAVTB   EQU    0050H
00023     0059     DSTPTR   EQU    0059H
00024     E001     SWICH1   EQU    0E001H
00025     E000     SWICH2   EQU    0E000H
00026     E400     SWICH3   EQU    0E400H
00027     E405     LATCH1   EQU    0E405H
00028     F000     LATCH2   EQU    0F000H
00029     E800     PIADRA   EQU    0E800H
00030     E800     PIADRA   EQU    0E800H
00031     E801     PIACRA   EQU    0E801H
00032     E802     PIADRB   EQU    0E802H
00033     E802     PIADRB   EQU    0E802H
00034     E803     PIACRB   EQU    0E803H
00035     EC00     RDDISP   EQU    0EC00H
00036              ;
```

Tektronix M6800 ASM V3.X Page 2

```
                         ;
                         ;*******************************
                         ;***** MAIN PROGRAM *****
                         ;*******************************
                         ;
        F800      >               ORG     0F800H
F800 0F                  MAIN     SEI
F801 7F0000                       CLR     00
F804 CE007F                       LDX     #007FH
F807 6F00                CLEAR    CLR     ,X              ; CLEAR SYSTEM RAM SPACE 00-7F
F809 09                           DEX
F80A 26FB                         BNE     CLEAR
F80C 7F0004                       CLR     ONECNT          ; INITIALIZE PARITY COUNTER
F80F 7F0005                       CLR     INTPAS          ; CLEAR INTERRUPT PASS FLAG
F812 7F0007                       CLR     ERRFLG          ; CLEAR COMM. ERROR INDICATOR
F815 7F0009                       CLR     LOCK            ; CLEAR DEAD-LOCK FLAG
F818 7F000B                       CLR     DATAL1
F81B 7F000C                       CLR     DATAL2
F81E CEFFFF                       LDX     #0FFFFH
F821 DF0A                         STX     DISPL           ; CLEAR OFF DISPLAYS
F823 86FF                         LDA     A     #0FFH
F825 B7E802                       STA     A     PIADRB    ; PIA PORT B AS OUTPUT
F828 4F                           CLR     A
F829 970C                         STA     A     DISPL+2   ; CLEAR FUEL LEVEL INDICATOR
F82B 970D                         STA     A     DISPL+3   ; CLEAR DASHBOARD INDICATORS
F82D 8634                         LDA     A     #34H      ; SELECT PIA OUTPUT REG. B
F82F B7E803                       STA     A     PIACRB    ; AND DISABLE DISPLAY FLASHING
F832 BDFC20       >               JSR     DISPLAY
F835 CE021F                       LDX     #021FH          ; TIMEOUT DELAY = 20 MS
F838 DF22                         STX     TIMCTR          ; INITIALIZE TIMEOUT COUNTER
F83A CE0059                       LDX     #DSTPIR
F83D DF10                         STX     DIAGTIC
```

```
Tektronix        M6800 ASM V3.3                                          Page      3

F83F F6E400                  LDA    B      SWICH3          ; GET NUMBER OF SLAVE UNITS
F842 C4E0                    AND    B      %11100000B
F844 59                      ROL    B
F845 59                      ROL    B
F846 59                      ROL    B
F847 59                      ROL    B
F848 5C                      INC    B
F849 D708                    STA    B      SLAVE
F84B CE0050                  LDX    #SLAVTB
F84E DF50                    STX    SLAVTB
F850 DB51                    ADD    B      SLAVTB+1
F852 D751                    STA    B      SLAVTB+1
F854 DE50                    LDX    SLAVTB
F856 C6FF                    LDA    B      #0FFH           ; END OF SLAVETB FLAG
F858 E700                    STA    B      0,X
F85A 09                      DEX
F85B D608                    LDA    B      SLAVE
                  ;
                  ;**********************************************
                  ;****** BUILT SLAVE UNIT TABLE *****
                  ;**********************************************
                  ;
F85D E700         FILLTB     STA    B      0,X
F85F 6600                    ROR    0,X
F861 6600                    ROR    0,X
F863 6600                    ROR    0,X
F865 6600                    ROR    0,X
F867 6600                    ROR    0,X
F869 09                      DEX
F86A 5A                      DEC    B
F86B 26F0                    BNE    FILLTB
F86D 08                      INX                               ; POINT TO START OF SLAVE TABLE
F86E DF20                    STX    SLVPTR
```

```
Tektronix      M6800 ASM V3.3                                              PAGE   4

00105 F870 8650                    LDA    A      #$0F
00106 F872 9B08                    AND    A      [illegible]
00107 F874 9725                    STA    A      SCRACH+1
00108 F876 5F                      CLR    B
00109 F877 D724                    STA    B      SCRACH
00110 F879 DE24                    LDX    SCRACH
00111 F87B 09                      DEX
00112 F87C A600                    LDA    A      ,X
00113 F87E 9703                    STA    A      LIMIT
00114 F880 DE20                    LDX    SLVPTR
00115                       ;
00116                       ;*********************************
00117                       ;***** MAIN PROGRAM LOOP *****
00118                       ;*********************************
00119                       ;
00120 F882 A600     LOOP           LDA    A      ,X          ; GET NEXT SLAVE ADDRESS
00121 F884 81FF                    CMP    A      #0FFH       ; TEST FOR END OF TABLE FLAG
00122 F886 2605                    BNE    NXT                ; IF NOT CONTINUE
00123 F888 CE0050                  LDX    #SLAVTB            ; OTHERWISE, RESET POINTER
00124 F88B A600                    LDA    A      ,X
00125 F88D 81F0     NXT            CMP    A      #0F0H       ; TEST FOR FAULTY SLAVE
00126 F88F 2603                    BNE    NXT1               ; IF OK CONTINUE
00127 F891 08                      INX                       ; IF FAULTY SKIP TO NEXT ONE
00128 F892 20EE                    BRA    LOOP
00129 F894 9730     NXT1           STA    A      CMDTAB      ; PUT ADDRESS BYTE IN CMDTAB
00130 F896 BDFB31  >               JSR    GETCMD             ; READ DRIVER COMMANDS
00131 F899 BDFAFE  >               JSR    INITX              ; PREPARE TO BROADCAST MESSAGE
00132 F89C A600     SEROUT         LDA    A      ,X          ; [5]  GET FIRST BYTE
00133 F89E 9701                    STA    A      OUTBUF      ; [4]  STORE IN OUT. BUFFER
00134 F8A0 BDFA21  >               JSR    TXMIT              ; [9]  SERIALIZE AND TRANSMIT
00135 F8A3 08                      INX                       ; [4]  INCREMENT CMDTAB POINTER
00136 F8A4 8C0034                  CPX    #0034H             ; [3]  END OF MESSAGE ?
00137 F8A7 26F3                    BNE    SEROUT             ; [4]  IF NOT TRANSMIT NEXT ONE
00138 F8A9 01                      NOP
```

```
00140 F8AA BDFB0C  >  IGNORE   JSR    INITRX          ; YES, PREPARE RECEPTION
00141 F8AD 0E                  CLI                    ; ENABLE CPU INTERRUPT
00142 F8AE 01         WAIT     NOP                    ; [2]  .
00143 F8AF DE22                LDX    TIMCTR          ; [4]  GET TIMEOUT COUNTER
00144 F8B1 09                  DEX                    ; [4]  TEST SCU TIME-OUT
00145 F8B2 270B                BEQ    TIMERR          ; [4]  IF YES, CALL TIMEOUT
00146 F8B4 DF22                STX    TIMCTR          ; [5]
00147 F8B6 2000                BRA    $+2             ; [4]
00148 F8B8 7D0005              TST    INTPAS          ; [6]
00149 F8BB 27F1                BEQ    WAIT            ; [4] TOTAL = 33 MC
00150 F8BD 2036                BRA    CTNUE

00151 F8BF 0F         TIMERR   SEI                    ; AVOID SPURIOUS INTERRUPTS
00152 F8C0 BDFB4F  >           JSR    TIMEOUT         ; FLAG WITH FO FAULTY SCU
00153 F8C3 BDFB60  >           JSR    DIAGNO          ; PUT CODE OF FAULTY SCU IN DIAGTIC
00154 F8C6 BDFB70  >           JSR    CONVERT         ; CONVERT CODE FOR INDICATORS
00155 F8C9 CE0059              LDX    #DSTPTR

00156 F8CC A600                LDA    A       ,X      ; GET THIS CODE AND PUT IN FLASH
00157 F8CE 81FF                CMP    A       #0FFH   ; RETEST AGAIN
00158 F8D0 2723                BEQ    CTNUE
00159 F8D2 970F                STA    A       FLASH
00160 F8D4 201F                BRA    CTNUE
```

```
00162                    ;
00163                    ;*************************
00164                    ; ANALYZE SLAVE FEEDBACK *
00165                    ;*************************
00166                    ;
00167 F8D6 0F                    SEI
00168 F8D7 7F0005                CLR     INTPAS
00169 F8DA 9607                  LDA     A       ERRFLG          ; CHECK MESSAGE INTEGRITY
00170 F8DC 16                    TAB                             ; MAKE EXTRA COPY OF ERROR
00171 F8DD 8401                  AND     A       #00000001B      ; FALSE START BIT ERROR
00172 F8DF 26C9                  BNE     IGNORE                  ; YES, IGNORE
00173 F8E1 17                    TBA
00174 F8E2 C402                  AND     B       #00000010B      ; PARITY ERROR
00175 F8E4 2726                  BEQ     PROCEED
00176 F8E6 8404                  AND     A       #00000100B      ; FALSE START BIT
00177 F8E8 270B                  BEQ     CTNUE                   ; IF MESSAGE OK, CONTINUE
00178 F8EA 7C0009                INC     LOCK                    ; UPDATE LOCK COUNTER FOR DEADLOCK
00179 F8ED 9609                  LDA     A       LOCK
00180 F8EF 9108                  CMP     A       SLAVE
00181 F8F1 2E1F                  BGT     BUSERR
00182 F8F3 2017                  BRA     PROCEED                 ; CORRUPTED MESSAGE, BROADCAST NEXT
00183 F8F5 0F            CTNUE   SEI
00184                    ; MESSAGE RECEIVED OK
00185                    ; UPDATE DISPLAYED DATA AND DIGNOSIC INFORMATION
00186 F8F6 7F0009                CLR     LOCK
00187 F8F9 CE0034                LDX     #0034H
00188 F8FC BDFBAC  >             JSR     DASHBRD                 ; DISPATCH RECEIVED DATA
00189 F8FF BDFC20  >             JSR     DISPLAY
00190 F902 A600                  LDA     A       ,X              ; UPDATE DISPLAYS
00191 F904 B7E802                STA     A       PIADRB
00192 F907 A601                  LDA     A       1,X
00193 F909 B7E800                STA     A       PIADRA
00194 F90C DE20          PROCEED LDX     SLVPTR
00195 F90E 08                    INX                             ; UPDATE SLAVE TABLE POINTER
00196 F90F 7EF882  >             JMP     LOOP                    ; READY TO MESSAGE BROADCASTING
```

```
00198 F912 01          BUSERR   NOP
00199                  ; EMERGENCY MODE
00200 F913 8638                 LDA    A       #38H
00201 F915 B7E803               STA    A       PIACRB          ; CB2 SET HIGH TO SWITCH BUZZER ON
00202                  ; TURN ON ALL SCU INDICATORS
00203 F918 86FF                 LDA    A       #0FFH
00204 F91A 970C                 STA    A       DATAL2
00205                  ; CLEAR OTHER DASHBOARD INDICATORS
00206 F91C 8600                 LDA    A       #00
00207 F91E 970B                 STA    A       DATAL1
00208 F920 BDFC20   >           JSR    DISPLAY
00209 F923 20FE        NOSEND   BRA    NOSEND
00210 F925 3F                   SWI                            ; STOP
00211                  ; NO MORE SEND SYSTEM BUS IS SHORT-CIRCUITED TO GROUND
00212                  ;
00213                  ;*************************************
00214                  ;***** INTERRUPT SERVICE  ******
00215                  ;************ ROUTINE **********
00216                  ;
00217       FA00    >           ORG    0FA00H
00218                  ;
00219 FA00 01                   NOP                            ; [2]
00220 FA01 01                   NOP                            ; [2]
00221 FA02 7DFA02   >           TST    $                       ; [6]
00222 FA05 8604                 LDA    A       #04             ; [2]
00223 FA07 B7E801               STA    A       PIACRA          ; [5] DISABLE INTERRUPT
00224 FA0A CE0035               LDX    #0035H                  ; [3]
00225 FA0D BDFA73   >  SERIN    JSR    RECEIV                  ; [9]
00226 FA10 7D0007               TST    ERRFLG                  ; [6]
00227 FA13 2606                 BNE    EXIT                    ; [4]
00228 FA15 08                   INX                            ; [4]
00229 FA16 8C0039.              CPX    #0039H                  ; [3]
00230 FA19 26F2                 BNE    SERIN                   ; [4]
00231 FA1B 86FF        EXIT     LDA    A       #0FFH
00232 FA1D 9705                 STA    A       INTPAS
00233 FA1F 0E                   CLI
00234 FA20 3B                   RTI
00235                  ;
```

```
Tektronix      M6800 ASM V3.3                                        Page    8

                                ;
                                ;***********************************
                                ;***** TRANSMIT        DATA *****
                                ;********** SUBROUTINE **********
                                ;
FA21 BDFAEB   >   TXMIT     JSR     ZEROUT               ; [11]/[10]
FA24 C605                   LDA     B        #05         ; [2]
FA26 BDFAF5   >             JSR     DLYB                 ; [53] DELAY
FA29 760001       AGAIN1    ROR     OUTCHR               ; [6]
FA2C 2408                   BCC     ZERO                 ; [4]
FA2E BDFAE5   >             JSR     ONEOUT               ; [11]/[10] * PATH FOR 1 ONLY
FA31 7C0004                 INC     ONECNT               ; [6] *
FA34 2008                   BRA     CONT                 ; [4] * DUMMY INSTRUCTION
FA36 BDFAEB   >   ZERO      JSR     ZEROUT               ; [11]/[10] ** PATH FOR 0 ONLY
FA39 7DFA39   >             TST     $                    ; [6] ** DUMMY INSTRUCTION
FA3C 2000                   BRA     CONT                 ; [6] ** DUMMY INSTRUCTION
FA3E C601         CONT      LDA     B        #01         ; [2]
FA40 BDFAF5   >             JSR     DLYB                 ; [29] DELAY
FA43 2000                   BRA     $+2                  ; [4]
FA45 7A0000                 DEC     BITCNT               ; [6] ONE BYTE TRANSMITTED?
FA48 26DF                   BNE     AGAIN1               ; [4]
FA4A 760004                 ROR     ONECNT               ; [6]
FA4D 2405                   BCC     EVENPR               ; [4] TRANSMIT PARITY BIT
FA4F BDFAE5   >   ODDPR     JSR     ONEOUT               ; [11]/[10]
FA52 2005                   BRA     FRWD                 ; [4]
FA54 BDFAEB   >   EVENPR    JSR     ZEROUT
FA57 2000                   BRA     FRWD
FA59 C606         FRWD      LDA     B        #06         ; [2]
FA5B BDFAF5   >             JSR     DLYB                 ; [59]
FA5E BDFAE5   >             JSR     ONEOUT               ; [11]/[10] TRANSMIT STOP BIT
FA61 8608                   LDA     A        #08         ; [2]
FA63 9700                   STA     A        BITCNT      ; [4]
FA65 4F                     CLR     A                    ; [2]
FA66 9704                   STA     A        ONECNT      ; [4] RESET PARITY COUNTER
FA68 C601                   LDA     B        #01         ; [2]
```

```
00273 FA6A 5A          DLYLP   DEC     B                       ; [2]
00274 FA6B 26FD                BNE     DLYLP                   ; [4]
00275 FA6D 2000                BRA     $+2                     ; [4]
00276 FA6F 7DFA6F  >           TST     $                       ; [6]
00277 FA72 39                  RTS                             ; [5]
00278                  ;
00279                  ;*************************************
00280                  ;***** RECEIVE       DATA ******
00281                  ;******** SUBROUTINE ********
00282                  ;
00283 FA73 B6E800      RECEIV  LDA     A       PIADPA          ; [4] TEST FALSE START BIT
00284 FA76 49                  ROL     A                       ; [2]
00285 FA77 2559                BCS     FSBERR                  ; [4]
00286 FA79 860A                LDA     A       #10             ; [2]
00287 FA7B BDFAF1  >           JSR     DLYA                    ; [74] DELAY
00288 FA7E B6E800      SERIAL  LDA     A       PIADPA          ; [4] GET SERIAL DATA
00289 FA81 49                  ROL     A                       ; [2]
00290 FA82 2405                BCC     ZEROIN                  ; [4]
00291 FA84 7C0004      ONEIN   INC     ONECNT                  ; [6] * PATH FOR 1 ONLY
00292 FA87 2005                BRA     FRWARD                  ; [4] *
00293 FA89 7DFA89  >   ZEROIN  TST     $                       ; [6] ** PATH FOR 0 ONLY
00294 FA8C 2000                BRA     FRWARD                  ; [4] ** DUMMY INSTRUCTION
00295 FA8E 760002      FRWARD  ROR     INCHAR                  ; [6]
00296 FA91 8605                LDA     A       #05             ; [2]
00297 FA93 BDFAF1  >           JSR     DLYA                    ; [44]
00298 FA96 2000                BRA     $+2                     ; [4]
00299 FA98 7A0000              DEC     BITCNT                  ; [6]
00300 FA9B 26E1                BNE     SERIAL                  ; [4]
00301 FA9D B6E800              LDA     A       PIADPA          ; [4] TEST EVEN PARITY
00302 FAA0 49                  ROL     A                       ; [2]
00303 FAA1 2507                BCS     ODDPAR                  ; [4]
00304 FAA3 760004      EVENPA  ROR     ONECNT                  ; [6]
00305 FAA6 2530                BCS     PARERR                  ; [4]
00306 FAA8 2007                BRA     NOERR                   ; [4]
```

```
Tektronix      M6800 ASM V3.3                                    Page    10

00308 FAAA 760004      ODDPAR  ROR     ONECNT                  ; TEST PARITY ERROR
00309 FAAD 2429                BCC     PARERR
00310 FAAF 2000                BRA     NOERR
00311 FAB1 9602        NOERR   LDA     A       INCHAR          ; [4]
00312 FAB3 A700                STA     A       ,X              ; [6]
00313 FAB5 8A08                LDA     A       #08             ; [2]
00314 FAB7 9700                STA     A       BITCNT          ; [4] RESET BIT COUNTER
00315 FAB9 C603                LDA     B       #03             ; [2]
00316 FABB BDFAF5  >           JSR     DLYB                    ; [41]
00317 FABE 01                  NOP                             ; [2]
00318 FABF B6E800              LDA     A       PIADRA          ; [4] TEST FRAMING
00319 FAC2 49                  ROL     A                       ; [2]
00320 FAC3 2419                BCC     FRMERR                  ; [4]
00321 FAC5 4F                  CLR     A                       ; [2]
00322 FAC6 9707                STA     A       ERRFLG          ; [4]
00323 FAC8 7F0004              CLR     ONECNT                  ; [6] RESET PARITY COUNTER
00324 FACB 8602                LDA     A       #02             ; [2]
00325 FACD BDFAF1  >           JSR     DLYA                    ; [26] DELAY
00326 FAD0 01                  NOP                             ; [2]
00327 FAD1 39                  RTS                             ; [5]
00328 FAD2 9607        FSBERR  LDA     A       ERRFLG          ; UPDATE COMM. STATUS REGISTER
00329 FAD4 8A01                ORA     A       #00000001B
00330 FAD6 200A                BRA     RTRN
00331 FAD8 9607        PARERR  LDA     A       ERRFLG
00332 FADA 8A02                ORA     A       #00000010B
00333 FADC 2004                BRA     RTRN
00334 FADE 9607        FRMERR  LDA     A       ERRFLG
00335 FAE0 8A04                ORA     A       #00000100B
00336 FAE2 9707        RTRN    STA     A       ERRFLG
00337 FAE4 39                  RTS
```

```
                 ;
                 ;********************************
                 ;***** ONEOUT SUBROUTINE *****
                 ;********************************
                 ;
FAE5 8638        ONEOUT   LDA   A     #38H        ; [2] + [9] for JSR
FAE7 B7E801               STA   A     PIACRA      ; [5]
FAEA 39                   RTS                     ; [5] total = 21 MC
                 ;
                 ;********************************
                 ;***** ZEROUT SUBROUTINE *****
                 ;********************************
                 ;
FAEB 8630        ZEROUT   LDA   A     #30H        ; [5] + [9] for JSR
FAED B7E801               STA   A     PIACRA      ; [5]
FAF0 39                   RTS                     ; [5] total = 21 MC
                 ;
                 ;********************************
                 ;***** DELAY SUBROUTINES *****
                 ;********************************
                 ;
FAF1 4A          DLYA     DEC   A                 ; [2] + [9] for JSR
FAF2 26FD                 BNE   DLYA              ; [4]
FAF4 39                   RTS                     ; [5] total = (A)*6 + 14 MC
                 ;
FAF5 D706        DLYB     STA   B     BSAV        ; [4] + [9] for JSR
FAF7 5A          DLYBLP   DEC   B                 ; [+2]
FAF8 26FD                 BNE   DLYBLP            ; [+4]
FAFA D606                 LDA   B     BSAV        ; [3]
FAFC 01                   NOP                     ; [2]
FAFD 39                   RTS                     ; [5] total = (B)*6 + 23 MC
```

```
                   ;
                   ;*******************************************
                   ;***** INITIALIZE TRANSMISSION *****
                   ;*******************************************
                   ;
FAFE CE0030        INITX   LDX   #0030H          ; INITIALIZE CMD TAB POINTER
FB01 8608                  LDA   A    #08
FB03 9700                  STA   A    BITCNT     ; INITIALIZE BIT COUNTER
FB05 7F0004                CLR   ONECNT          ; RESET PARITY COUNTER
FB08 7F0007                CLR   ERRFLG          ; CLEAR COMMUNICATION STATUS REGISTER
FB0B 39                    RTS
                   ;
                   ;*****************************************
                   ;***** INITIALIZE RECEPTION *****
                   ;*****************************************
                   ;
FB0C 8608          INITRX  LDA   A    #08
FB0E 9700                  STA   A    BITCNT     ; INITIALIZE BIT COUNTER
FB10 7F0004                CLR   ONECNT          ; RESET FLAGS
FB13 7F0005                CLR   INTPAS
FB16 7F0007                CLR   ERRFLG
FB19 CE021F                LDX   #021FH          ; RESET TIMEOUT COUNTER. DELAY=20 MS
FB1C DF22                  STX   TIMCTR
FB1E 8639                  LDA   A    #39H
FB20 B7E801                STA   A    PIACRA     ; ENABLE INTERRUPT
FB23 867F                  LDA   A    #7FH
FB25 B7E801                STA   A    PIACRA
FB28 863D                  LDA   A    #3DH
FB2A B7E801                STA   A    PIACRA
FB2D B6E800                LDA   A    PIACRA     ; CLEAR CA1 INTERRUPT
FB30 39                    RTS
```

```
00403                    ;
00404                    ;*************************************
00405                    ;***** GET COMMAND SUBROUTINE *****
00406                    ;*************************************
00407                    ;
00408 FB31 9724          GETCMD   STA     A       SCRACH
00409 FB33 DF20                   STX     SLVPTR
00410 FB35 CE0030                 LDX     #CMDTAB
00411 FB38 B6E001                 LDA     A       SWICH1
00412 FB3B 43                     COM     A
00413 FB3C A701                   STA     A       1,X
00414 FB3E B6E000                 LDA     A       SWICH2
00415 FB41 43                     COM     A
00416 FB42 A702                   STA     A       2,X
00417 FB44 B6E400                 LDA     A       SWICH3
00418 FB47 43                     COM     A
00419 FB48 A703                   STA     A       3,X
00420 FB4A 9624                   LDA     A       SCRACH
00421 FB4C DE20                   LDX     SLVPTR
00422 FB4E 39                     RTS
00423                    ;
00424                    ;*************************************
00425                    ;***** TIMEOUT SUBROUTINE *****
00426                    ;*************************************
00427                    ;
00428 FB4F CE021F        TIMEOUT  LDX     #021FH                 ; RESET TIMEOUT COUNTER
00429 FB52 DF22                   STX     TIMCTR
00430 FB54 DE20                   LDX     SLVPTR                 ; PUT FLAG IN SLAVTB FOR FAULTY SLA
00431 FB56 E600                   LDA     B       ,X
00432 FB58 17                     TBA                            ; MAKE EXTRA COPY OF FAULTY SLAVE
00433 FB59 CAF0                   ORA     B       #0F0H          ; MAKE SURE THAT FAULT FLAG IS : F0
00434 FB5B C4F0                   AND     B       #0F0H
00435 FB5D E700                   STA     B       ,X
00436 FB5F 39                     RTS
```

```
00438                    ;
00439                    ;*************************************
00440                    ;***** DIAGNOSTIC SUBROUTINE *****
00441                    ;*************************************
00442                    ;
00443 FB60 DE10          DIAGNO   LDX     DIAGTIC
00444 FB62 46                     ROR     A
00445 FB63 46                     ROR     A
00446 FB64 46                     ROR     A
00447 FB65 46                     ROR     A
00448 FB66 A700                   STA     A       ,X
00449 FB68 08                     INX
00450 FB69 86FF                   LDA     A       #0FFH           ; END-OF-DIAGNOSTIC TABLE FLAG
00451 FB6B A700                   STA     A       ,X
00452 FB6D DF10                   STX     DIAGTIC
00453 FB6F 39                     RTS
00454                    ;
00455                    ;*************************************
00456                    ;***** CODE CONVERSION SUBROUTINE *****
00457                    ;*************************************
00458                    ;
00459 FB70 01            CONVERT  NOP
00460 FB71 8100                   CMP     A       #00             ; TEST FOR ILLEGAL SCU ADDRESS
00461 FB73 2732                   BEQ     RESULT
00462 FB75 8101                   CMP     A       #01             ; SCU 01 FAULTY
00463 FB77 272E                   BEQ     RESULT
00464 FB79 8102                   CMP     A       #02             ; SCU 02 FAULTY
00465 FB7B 272A                   BEQ     RESULT
00466 FB7D 8103                   CMP     A       #03             ; SCU 03 FAULTY
00467 FB7F 2604                   BNE     RESUL1
00468 FB81 8604                   LDA     A       #04
00469 FB83 2022                   BRA     RESULT
00470 FB85 8104          RESUL1   CMP     A       #04             ; SCU 04 FAULTY
00471 FB87 2604                   BNE     RESUL2
00472 FB89 8608                   LDA     A       #08
00473 FB8B 201A                   BRA     RESULT
```

```
00475 FB8D 8105       RESUL2   CMP    A      #05          ; SCU 05 FAULTY
00476 FB8F 2604                BNE    RESUL3
00477 FB91 8610                LDA    A      #10H
00478 FB93 2012                BRA    RESULT
00479 FB95 8106       RESUL3   CMP    A      #06          ; SCU 06 FAULTY
00480 FB97 2604                BNE    RESUL4
00481 FB99 8620                LDA    A      #20H
00482 FB9B 200A                BRA    RESULT
00483 FB9D 8107       RESUL4   CMP    A      #07          ; SCU 07 FAULTY
00484 FB9F 2604                BNE    RESUL5
00485 FBA1 8640                LDA    A      #40H
00486 FBA3 2002                BRA    RESULT
00487 FBA5 8680       RESUL5   LDA    A      #80H         ; SCU 08 FAULTY
00488 FBA7 9A0C       RESULT   ORA    A      DATAL2
00489 FBA9 970C                STA    A      DATAL2
00490 FBAB 39                  RTS
00491                 ;
00492                 ;*********************************
00493                 ;***** DASHBOARD INDICATORS *****
00494                 ;*********************************
00495                 ;
00496                 ; CALLED WITH INDEX REGISTER=0034H
00497                           ; EXTRACT DATA FROM SLAVE FEEDBACK
00498                           ; AND SEND TO CORRESPONDING MEMORY LOCATIONS
00499                           ; FOR DISPLAY SUCH AS HEAD LAMP, TURN SIGNAL
00500                           ; FUEL LEVEL, BELT CONTINUITY, ...
00501 FBAC 01         DASHBRD  NOP
00502 FBAD E600                LDA    B      ,X           ; GET MESSAGE ORIGIN SLAVE ADDRESS
00503 FBAF C103                CMP    B      #03          ; IS MESSAGE FROM SCU #03
00504 FBB1 260E                BNE    CTNUE1               ; IF NOT CONTINUE
00505 FBB3 08                  INX                         ; INDEX REGISTER = 0035H
00506 FBB4 A600                LDA    A      ,X           ; TEST NIL MESSAGE FROM SCU #03
00507 FBB6 8100                CMP    A      #00
00508 FBB8 2627                BNE    CTNUE3
```

```
00510 FBBA 08                    INX
00511 FBBB A600                  LDA     A       ,X
00512 FBBD 8100                  CMP     A       #00
00513 FBBF 275D                  BEQ     RETURN
00514 FBC1 C101          CTNUE1  CMP     B       #01             ; IS MESSAGE FROM SCU #01
00515 FBC3 260D                  BNE     CTNUE2
00516                    ; DISPATCH DATA COMING FROM SCU 01
00517 FBC5 A600                  LDA     A       ,X
00518 FBC7 46                    ROR     A                       ; RIGHT HEADLIGHT MAIN/DIP
00519 FBC8 46                    ROR     A                       ; FLASHER AND BACK UP LIGHTS
00520 FBC9 843F                  AND     A       #00111111B
00521 FBCB 43                    COM     A
00522 FBCC 940B                  AND     A       DATAL1
00523 FBCE 970B                  STA     A       DATAL1          ; UPDATE LATCH1 MEMORY LOCATION
00524 FBD0 204C                  BRA     RETURN
00525 FBD2 C102          CTNUE2  CMP     B       #02             ; IS MESSAGE FROM SCU 02
00526                    ; DISPATCH DATA COMING FROM SCU 02
00527 FBD4 A600                  LDA     A       ,X
00528 FBD6 46                    ROR     A
00529 FBD7 46                    ROR     A                       ; HEADLIGHT, FLASHER, AND BACK UP
00530 FBD8 841F                  AND     A       #00011111B      ; LEFT SIDE
00531 FBDA 43                    COM     A
00532 FBDB 940B                  AND     A       DATAL1
00533 FBDD 970B                  STA     A       DATAL1
00534 FBDF 203D                  BRA     RETURN
00535 FBE1 01            CTNUE3  NOP
00536                    ; DISPATCH DATA COMING FROM SCU 3
00537 FBE2 CE0035                LDX     #0035H                  ; RESET DATA TABLE POINTER
00538 FBE5 46                    ROR     A
00539 FBE6 46                    ROR     A                       ; BACK UP, STOP LIGHT AND FLASHER
00540 FBE7 8418                  AND     A       #00011000B      ; RIGHT SIDE
00541 FBE9 43                    COM     A
00542 FBEA 940B                  AND     A       DATAL1
00543 FBEC 970B                  STA     A       DATAL1
00544 FBEE 202E                  BRA     RETURN
```

```
00546 FBF0 C104        CTNUE4  CMP     B       #04             ; IS MESSAGE COMING FROM SCU 04
00547 FBF2 260B                BNE     CTNUE5
00548                  ; DISPATCH DATA COMING FROM SCU 04
00549 FBF4 46                  ROR     A
00550 FBF5 46                  ROR     A                       ; TAIL LICENCE, FLASHER, BACK UP
00551 FBF6 8438                AND     A       #00111000B      ; LEFT SIDE
00552 FBF8 43                  COM     A
00553 FBF9 940B                AND     A       DATAL1
00554 FBFB 970B                STA     A       DATAL1
00555 FBFD 201F                BRA     RETURN
00556 FBFF C105        CTNUE5  CMP     B       #05
00557 FC01 261A                BNE     CTNUE6
00558                  ; DISPATCH DATA COMING FROM SCU 05
00559 FC03 A600                LDA     A       ,X
00560 FC05 49                  ROL     A                       ; HALT CONTINUITY, OIL PRESSURE,
00561 FC06 49                  ROL     A                       ; BRAKE, RPM, TEMPERATURE, FUEL
00562 FC07 84E0                AND     A       #11100000B
00563 FC09 43                  COM     A
00564 FC0A 940B                AND     A       DATAL1
00565 FC0C 970B                STA     A       DATAL1
00566 FC0E A600                LDA     A       ,X
00567 FC10 841F                AND     A       #00011111B
00568 FC12 970D                STA     A       FUELEV
00569 FC14 A601                LDA     A       1,X
00570 FC16 B7EC00              STA     A       RDUISP
00571 FC19 A602                LDA     A       2,X
00572 FC1B 970E                STA     A       RPMEMO
00573 FC1D 01          CTNUE6  NOP                             ; DATA FROM OPTIONAL SCUS
00574 FC1E 01          RETURN  NOP
00575 FC1F 39                  RTS
```

```
00577                          ;
00578                          ;*****************************
00579                          ;***** DISPLAY SUBROUTINE *****
00580                          ;*****************************
00581                          ;
00582 FC20 DF24      DISPLAY   STX     SCRACH
00583 FC22 CE000A              LDX     #DISPL
00584 FC25 A600                LDA     A       ,X
00585 FC27 B7EC00              STA     A       RDDISP          ; RIGHT DIGITS DISPLAY
00586 FC2A 960E                LDA     A       RFMEMO
00587 FC2C B7E802              STA     A       PIADRB          ; LEFT DIGITS DISPLAY
00588 FC2F 960D                LDA     A       FUELEV
00589 FC31 B7E800              STA     A       PIADRA          ; FUEL LEVEL DISPLAY
00590 FC34 A601                LDA     A       1,X
00591 FC36 B7E405              STA     A       LATCH1          ; 8218 LATCH1 DASHBOARD INDICATORS
00592 FC39 A602                LDA     A       2,X
00593 FC3B B7F000              STA     A       LATCH2          ; 8212 LATCH2 SCU FAULT INDICATORS
00594 FC3E DE24                LDX     SCRACH
00595 FC40 39                  RTS
00596                          END
```

A N N E X E

Liste et composition du jury en vue de la soutenance de la thèse de Magister en Electronique Appliquée par Mr MEGHRICHE KAMAL.

PRESIDENT / - Dr TEDJINI BAILICHE HACENE (Maitre de conférence à l'U.S.T.H.3)

RAPPORTEUR / - Dr AL-LAMI BASSIM (Maitre de conférence à l'INELEC)

EXAMINATEURS / - Dr BOURDOUCEN HADJ (I.N.E.L.E.C)
- Dr HARICHE KAMEL (I.N.E.L.EC)
- Dr DIAF MOUSSA (chargé de recherche Université de Tizi-Ouzou)