MAGNETIC BUBBLE MEMORY SYSTEM IMPLEMENTATION AND TESTING

by

Balraj S. Joll

A thesis
presented to the University of Manitoba
in partial fulfillment of the
requirements for the degree of
Master of Science
in
Department of Electrical Engineering

Winnipeg, Manitoba, 1981

MAGNETIC BUBBLE MEMORY SYSTEM IMPLEMENTATION AND TESTING

BY

BALRAJ S. JOLL

A thesis submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

I hereby declare that I am the sole author of this thesis.

I authorize the University of Manitoba to lend this thesis to other institutions or individuals for the purpose of scholarly research.

B. Joll

Balraj S. Joll

I further authorize the University of Manitoba to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

B. Joll

Balraj S. Joll

The University of Manitoba requires the signatures of all persons using or photocopying this thesis. Please sign below, and give address and date.

ABSTRACT

This thesis presents an overview of the state of the art in magnetic bubble memories (MBM) and the results of a study of the reliability of bubble memories as a storage medium from the systems aspect. The interface of MBM in microprocessor systems was also investigated.

Reliability of MBM was evaluated in terms of device sensitivity to various bit patterns. Bit patterns were chosen to examine the following failure modes: (i) collapse, (ii) strip-out, and (iii) failure of a domain to cross a gap region. Tests, based on the above bit patterns, were conducted to evaluate the effects of device control and sensing elements on data integrity, and the effects of failure mechanisms in the major and minor loops of the TIB-0203.

From the tests it was determined that the observed error rate during any single test was less than 0.6×10^{-11} and the overall rate was less than 4.11×10^{-11}. Both error rates are below the error rate of 10^{-9} specified by the manufacturer. Other tests were also conducted to observe the effects on the implemented system due to prolonged operation. Results indicate that there was no measurable degradation in device operation and that the implemented bubble memory system and bubble module are both highly reliable.

ACKNOWLEDGEMENTS

The author would like to take this opportunity to thank the many individuals and organizations that have contributed towards this research. Foremost is Dr. W. Kinsner, for his guidance in directing the field of study and for the discussions leading to an understanding of the subject material. The author would also like to express his appreciation for the technical assistance of Jack Sill and the Industrial Applications of Microelectronic Centre Inc.

During the course of this research, contributions of components from G. Cox and Texas Instruments, and the assistance of John Hentic of Burrougs Business Machines were greatly appreciated. Other colleagues who have contributed by providing helpful suggestions during the experimental work and/or during the writing of this thesis are, A. W. DeGroot, A. Weirich, J. Reimer, V. Shkawrytko, and G. Smith.

The author also wishes to express his indebtedness to Mary-Ann Tyler, Mary Napier, and Toni Allardyce for their help in the typing of this thesis, and to technologists A. McKay and K. Podaima.

Finally the author would like to thank the National Research Council for providing the financial support for this endeavor.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter I

INTRODUCTION

In the mid 1970s, the access times of semiconductor memories were significantly lower than those of fixed head disk and tape storage devices. Semiconductor memories exhibited an access time of less than 1μs whereas disk and tapes have access times greater than 5ms. In 1977 this "access gap " was bridged with the introduction of charge-coupled devices (CCD) and magnetic bubble memories (MBM) [1,2]. Charge coupled devices and MBM have become pronounced in the marketplace. This can be attributed to the following factors: (i) the low latency time of CCD memories, (ii) the nonvolatility of MBM, (iii) the small access times of both devices, and (iv) the low power requirements of MBMs.

The objective of this research was to study the reliability of MBM as a storage medium from the systems aspect, and to provide insight into MBMs and the incorporation of bubble memories in microprocessor systems. Reliability of MBM was evaluated in terms of device sensitivity to various bit patterns. Patterns were selected to simulate worst case operating conditions. Tests were also performed to determine MBM system reliability during a prolonged period of operation.

This section presents an introduction to bubble memories. This includes a discussion of the stability of magnetic domains and a description of device architecture, as well as current device trends. This is followed by a description of the hardware and software associated with a

bubble memory/microprocessor system used in conducting system testing. Finally the testing and test results evaluating the bubble memory system reliability are presented.

1.1 MAGNETIC BUBBLES

Magnetic bubbles are cylindrical magnetic domains formed in single crystal thin films of ferrites [3], garnets [4,5], or in amorphous materials [6,7]. The domains are formed when the film is subjected to an external magnetic field applied perpendicular to the film surface. Without the external field , the domains form serpentine domains of equal and opposite magnetization such as to minimize the total energy of the bubble platelet. This is the minimum energy configuration. Both types of domains are illustrated in Fig. 1.1. Bubbles and serpentine domains were first observed in orthoferrites in 1959, and by 1967 a working bubble memory had been demonstrated [8]. The presence and absence of a domain was used to represent a logical "1" and logical "0" respectively. The use of magnetic domains is not limited to purely memory functions as the use of bubbles in performing bubble logic has been postulated and tested [9,10].

By 1970, the use of garnets became prevalent, which resulted in increased storage densities of MBM [8]. This allowed a reduction in bubble diameter from approximately 150μm to 3-5μm bubbles utilized by present devices. Garnets and amorphous materials have the potential for sustaining lum bubbles.

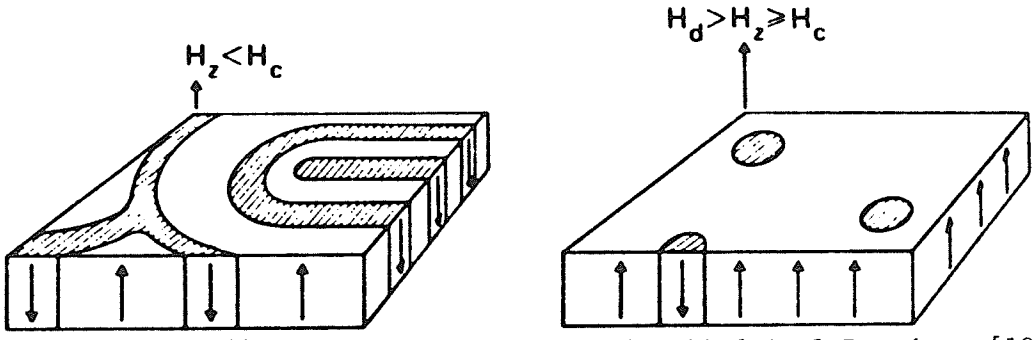


FIGURE 1.1: Serpentine and Cylindrical Domains [10]

The stripe domain and cylindrical domain phenomena have been analyzed from the stability perspective. Stripe domain analysis was first performed by C. Kooy and U. Enz [11] and later by A.H. Bobeck [12]. The cylindrical domain analysis was carried out by A.A. Thiele [13,14]. The work conducted by Thiele is the basis for the design and material selection in present devices. To gain insight into the behavior of bubbles when subjected to numerous forces, the work of Thiele is outlined below.

1.2 STABILITY ANALYSIS

Consider a magnetic domain configuration used by Thiele [13] in performing the stability analysis as shown in Fig. 1.2. The domain shape is represented by the following quasi-orthogonal coordinates.

$$r_b(\theta) = \sum_{n=0}^{\infty} r_n \cos[n(\theta - \theta_n)] \qquad (1)$$

where r_n and θ_n are coefficients. In order to account for small arbitrary variations from a perfectly circular domain Eq. (1) may be written as

$$r_b(\theta) = r_o + \Delta r_o + \sum_{n=1}^{\infty} \Delta r_n \cos[n(\theta - \theta_n - \Delta\theta_n)] \qquad (2)$$

where

$$|r_o| >> |\Delta r_o| + \sum_{n=1}^{\infty} n\,|\Delta r_n|$$

This constraint insures that the variations are small compared to the domain radius. To facilitate analysis the following assumptions are made:

a) The wall energy (σ_W) is independent of the wall configuration,

b) The domain wall is of negligible width,

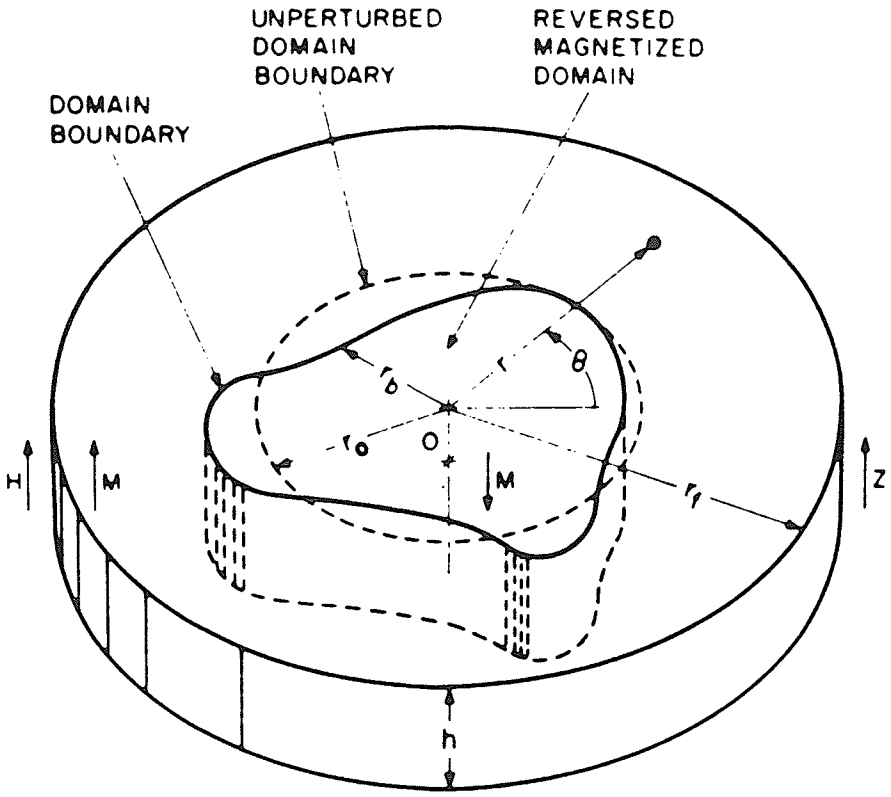


FIGURE 1.2: Cylindrical Domain Configuration [8]

c) The domain is in an infinite platelet of uniform thickness, and

d) The magnetization is along the z axis.

The following three energy terms are important when considering domain stability: the wall energy (E_W), the energy due to the externally applied field (E_H), and the internal magnetostatic energy (E_M). The wall energy is due to the exchange energy and anisotropy energy. The total energy for the ideal model is assumed to be the sum of the above three energies, and can be written as

$$E_T = E_W + E_H + E_M \tag{3}$$

The individual components are given by

$$E_W = \int_s \sigma_W ds = h\,\sigma_W \int_0^{2\pi} \{r_b^2(\theta) + [\frac{dr_b(\theta)}{d\theta}]^2\}^{1/2}\, d\theta \tag{4}$$

$$E_H = -\int_v \bar{M}\cdot\bar{H} dv = hHMs\int_0^{2\pi} r_b^2(\theta)\, d\theta \tag{5}$$

$$E_M = \frac{1}{2}\int_v \rho\psi dv = \frac{1}{2}\int_v\int_{v'} \frac{\frac{\partial M_Z}{\partial Z}\frac{\partial M'_Z}{\partial Z'}}{|\bar{r} - \bar{r}'|}\, dv'\, dv \tag{6}$$

where h = plate thickness.
s = wall area.
M = magnetization.
H = applied field (positive when tending to collapse the domain).
ψ = scalar magnetic potential.
$\rho = \nabla\cdot M$ = magnetic pole density.
M_s = saturation magnetization.
r_0 = domain radius.
σ_W = energy per unit wall area.

Using a Taylor series expansion in expressing the variation in total energy due to variations in r and θ we obtain:

$$\Delta E_T = \sum_{n=0}^{\infty} \left[\left(\frac{\partial E_T}{\partial r_n} \right)_0 \Delta r_n + \left(\frac{\partial E_T}{\partial \theta_n} \right)_0 \Delta \theta_n \right]$$

$$+ \frac{1}{2} \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} \left[\left(\frac{\partial^2 E_T}{\partial r_n \partial r_m} \right)_0 \Delta r_n \Delta r_m + 2 \left(\frac{\partial^2 E_T}{\partial r_n \partial \theta_m} \right)_0 \Delta r_n \Delta \theta_n \right.$$

$$\left. + \left(\frac{\partial^2 E_T}{\partial \theta_n \partial \theta_m} \right)_0 \Delta \theta_n \Delta \theta_m \right] + \text{higher order terms} \qquad (7)$$

By making appropriate substitutions, Thiele showed that the total energy variation is:

$$\Delta E = \left[2\pi h \sigma_w + 4\pi h 4 M_s r_o - (2\pi h^2)(4\pi M_s)\, F(a) \right] \Delta r_o$$

$$+ \frac{1}{2} \left[4\pi h M_s H - (4\pi h)(4\pi M_s^2) \frac{dF(a)}{da} \right] (\Delta r_o)^2$$

$$+ \frac{1}{2} \sum_{n=1}^{\infty} \left\{ \frac{\pi}{r_o} h\, \sigma_w n^2 + 2\pi h M_s H - (2\pi h)(4\pi M_s^2) \frac{dF}{da} \right.$$

$$\left. + (h)(4\pi M_s^2)\, a \left[L_n(1/a)^2 - L_n(0) \right] \right\} (\Delta r_n)^2$$

$$+ \text{higher order terms} \qquad (8)$$

where

$$F(a) = \frac{2}{\pi} a^2 \left[\frac{(1 + a^2)^{1/2}}{a} E \left(\frac{a^2}{1 + a^2} \right) - 1 \right]$$

$$E(K) = \int_0^{\pi/2} (1 - K \sin^2 \alpha)^{1/2}\, d\alpha$$

$$L_n(1/a) = \frac{4}{a} \int_0^{\pi/2} \frac{\sin^2 n\beta}{(1 + 1/a^2 \sin^2\beta)^{1/2}} d\beta$$

$$L_n(0) = L_n(1/a)\Big|_{a=\infty}$$

and the aspect ratio $a = 2r_0/h$.

Upon normalization, Eq (8) becomes

$$\frac{\Delta E}{2(4\pi M_s^2)(\pi h^3)} = \underbrace{\left[\frac{\ell}{h} \right]}_{(1)} + \underbrace{\frac{d}{h} \frac{H}{4\pi M_s}}_{(2)} - \underbrace{F(a)}_{(3)} \left] \frac{\Delta r_o}{h} \right.$$

$$+ \frac{1}{2} \left\{ -\left(\frac{2h}{d} \right) \left[\frac{\ell}{h} - S_o(a) \right] \left(\frac{\Delta r_o}{h} \right)^2 \right.$$

$$+ \sum_{n=2}^{\infty} (n^2 - 1) \left(\frac{h}{d} \right) \left[\frac{\ell}{h} - S_n(a) \right] \left(\frac{\Delta r_n}{h} \right)^2 \Bigg\}$$

$$+ \text{higher order terms} \qquad (9)$$

where ℓ, the characteristic length, and the stability functions S_0 and S_n are defined as follows:

$$\ell = \frac{\sigma_w}{4\pi M_s} \qquad (10)$$

$$S_o(a) \equiv F(a) - a \frac{\partial F(a)}{\partial A} \qquad (11)$$

and

$$S_n(a) \equiv \frac{-1}{n^2 - 1} \left\{ S_o(a) + \frac{a^2}{2\pi} \left[L_n(1/a)^2 - L_n(0) \right] \right\} \quad n \geq 2 \qquad (12)$$

In Eq. (9) terms (1), (2), and (3) correspond to the forces acting on the magnetic bubble due to the wall energy, applied field, and internal magnetostatic energy respectively. The remaining terms in Eq. (9) are called "stability coefficients" corresponding to the second order variations in energy. The force and stability functions are plotted in Fig. 1.3.

The following conditions are required for bubble stability:

a) The sum of the forces must equal zero, and

b) The second order variations in energy must be positive.

The two constraints, above, imply that for a stable bubble the following conditions must be satisfied:

$$\ell/h + a\, H/M_s - F(a) = 0 \tag{13}$$

$$S_n(a) < \ell/h < S_o(a) \tag{14}$$

Both equations (13) and (14) and Fig. 1.3 can be used to determine material parameters for bubble memories. For example, consider a particular material with a characteristic length 1. Choosing h=2.51 results in 1/h = 1/2.51 = 0.4. Figure 1.3 shows that for 1/h = 0.4, the stability region is between aspect ratios of 1.6 to 5.2 as determined by the bias field and saturation magnetization. The stability region is limited to within S_0 and S_2 since these two functions represent a transition from a circular domain to a stripe domain respectively. Now consider the application of a bias field $H = 0.17\ M_s$, a substitution of known parameters into Eq. (13) results in:

$$F(a) = 0.17a + 0.4 \tag{15}$$

Equation (15) has the following two solutions, a=1 and a=3.5, as illustrated in Fig. 1.3. The constraint that $1.6 < a < 5.2$ implies that the stable domain will have an aspect ratio of 3.5 . The use of stability design curves may be reversed to determine the characteristic length and saturation magnetization of the material [13].

The stability discussion alone gives insight into the complex configuration of a magnetic domain. The use of design curves enable direct analysis of the effects on the domain due to variations in material characteristics. Thiele's efforts have laid the groundwork for most of the devices currently being used.

1.3 FIELD ACCESS DEVICES AND FUNCTIONAL ELEMENTS

In field access devices the domain motion is the result of a field gradient produced by a rotating in-plane field interacting with a permalloy structure [15]. In considering MBM devices, the following six essential functions are required for an operational memory: (i) storage, (ii) propagation, (iii) generation, (iv) detection, (v) annihilation, and (vi) transfer.

1.3.1 Bubble Storage

Information is represented in MBM by the presence and absence of magnetic domains representing a logical "1" and a logical "0" respectively. In functional devices, a domain is kept stable by the application of an external field applied perpendicular to the storage film surface. The external field is generated by permanent magnets.

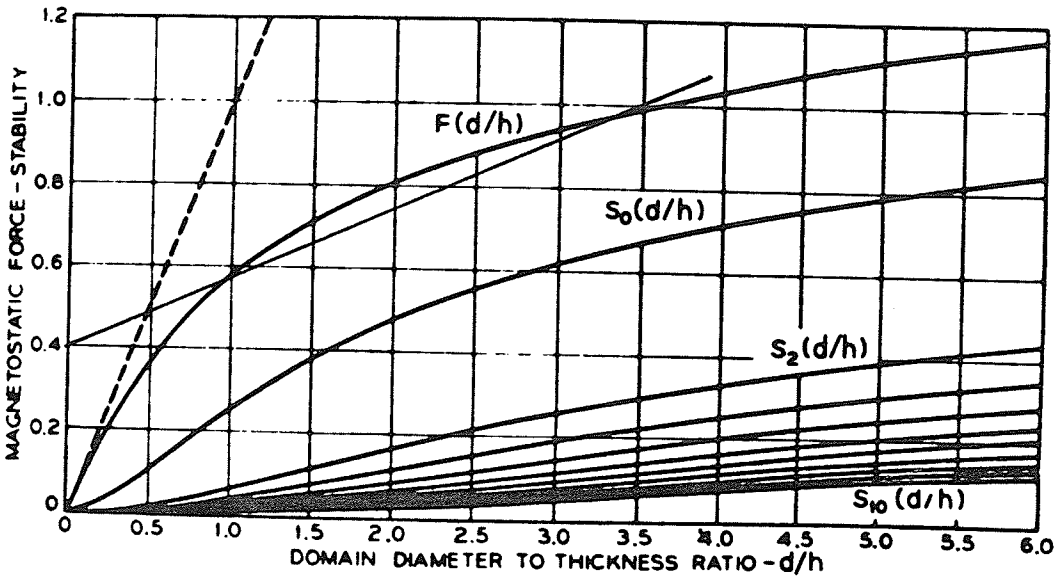


FIGURE 1.3: Force and Stability Functions [8]

1.3.2 Bubble Propagation

Propagation of bubbles is achieved through the interaction of an in-plane rotating field with the permalloy overlay structure such that attractive and repulsive poles are created. The principle of operation is that a rotating in-plane magnetic field produces a travelling wave that propagates a bubble. The wave is produced when the magnetization of the permalloy is changed sequentially by the in-plane field applied along the long axis. The permalloy magnetization produces field wells which hold the bubble during propagation [16,17]. Figure 1.4 illustrates domain motion using various propagating cicuits due to an application of a rotating in-plane field. Each propagating element is traversed in one period of the in-plane field.

First generation bubble devices employed the "T - bar" propagating pattern. This structure evolved through the various illustrated patterns to the presently used asymmetric chevrons. Asymmetric chevrons are discussed in Sec. 3.1. The evolution was an attempt to,(i) define more stable positions for bubbles (in the case of the "Y - bar", "X - bar" and modified "Y - bar" propagation pattern), and (ii) simplify the drive circuits by using propagation patterns not requiring a rotating in-plane field. Other propagation techniques include: angelfish propagation [18], parallel bar propagation [19], channel bar propagation [20], single conductor propagation [21,22], contiguous disk propagation [23], and current sheet propagation [24].

The angelfish propagation circuit utilizes an oscillating bias field to modulate the bubble diameter causing an inchworm-like motion. The parallel bar circuit uses an alternating transverse field interacting

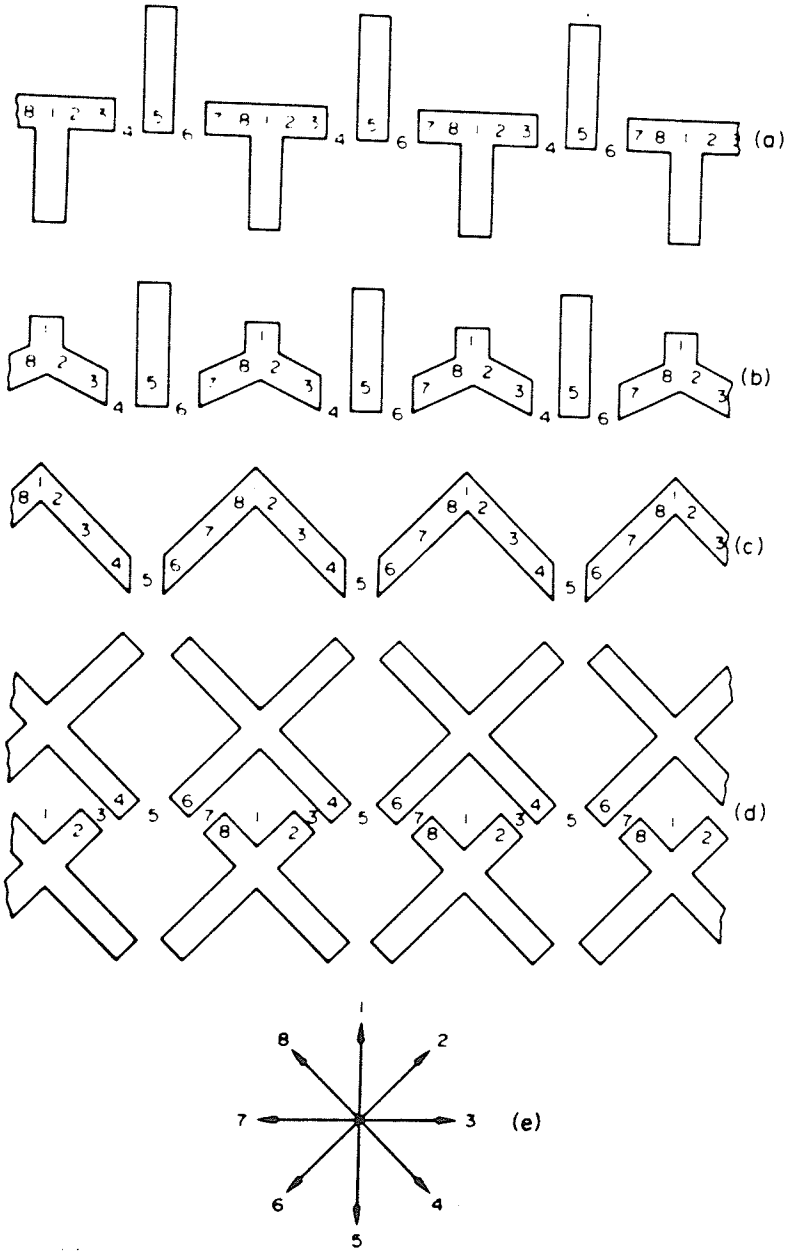
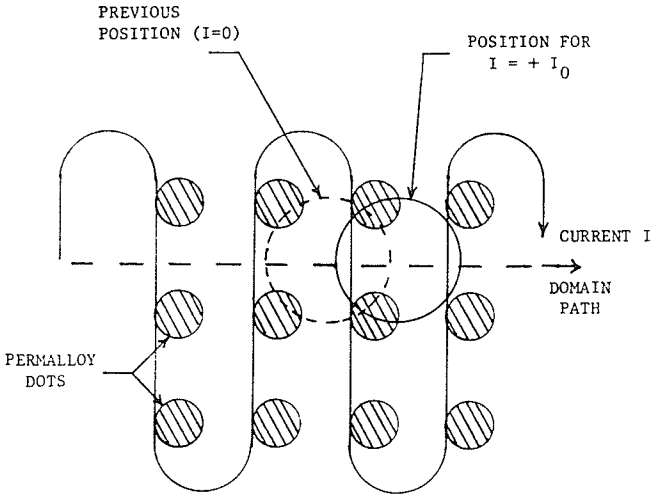


Figure 1.4: Bubble Domain Propagation Tracks [17]

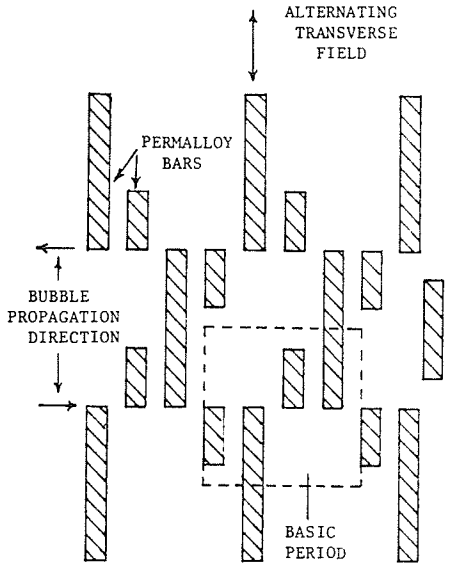
with parallel permalloy bars of different lengths to instigate propagation. Propagation in the channel bar circuit is due to a combination of forces stemming from a modulated channel, made by physical etching, and a permalloy bar magnetized by a transverse magnetic field. The latter technique uses current conductors to supply the field gradient required for bubble propagation. Permalloy may be used in conjunction with the current conductors to define stable positions for a bubble during propagation. The various propagation techniques are illustrated in Fig. 1.5.

The contiguous disk propagation circuit uses a gapless propagating pattern. The propagation pattern is created by implanting He+ ions into the domain carrying garnet film [25]. Figure 1.6 illustrates an interpretation of a domain pattern around an isolated hole. A charged wall is formed upon application of an in-plane field. Charged walls are domain walls formed when in-plane magnetization diverges and converges around an unimplanted hole [26,27,28]. Propagation via charged walls is illustrated in Fig. 1.7. Contiguous disk devices provide higher storage since smaller, coarser propagation elements can be realized using present lithographic capabilities. Complete functional memory elements have been described [29,30] and working contiguous disk memories have been demonstrated.

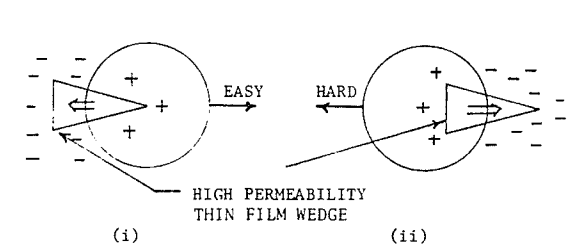
The current sheet propagation circuit departs from the conventional field access devices in that propagation is achieved using dual current conducting sheets. In field access devices, motion results from the attraction of a domain to the magnetic dipole formed by the interaction of an in-plane field with the permalloy overlay. For current access devices, an application of a current diverging around a slotted current

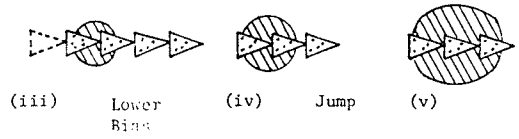


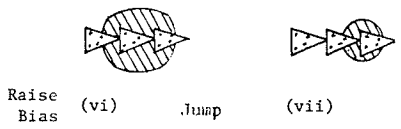
(a) Single Conductor Propagation



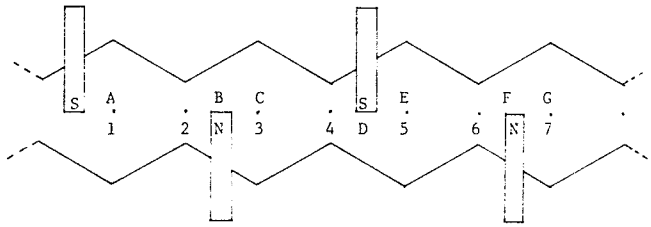
(b) Parallel-Bar Propagation

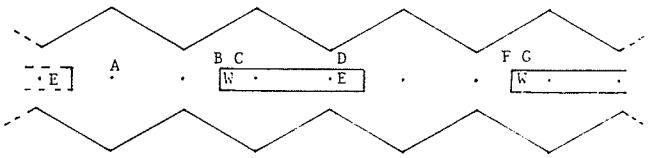


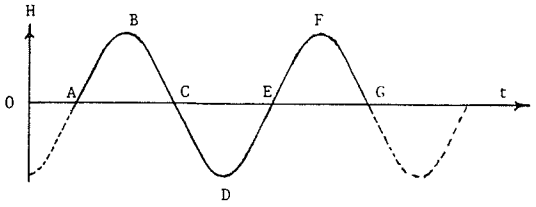




(c) Angelfish Propagation







(d) Channel-Bar Propagation

Figure 1.5: Other Propagating Circuits

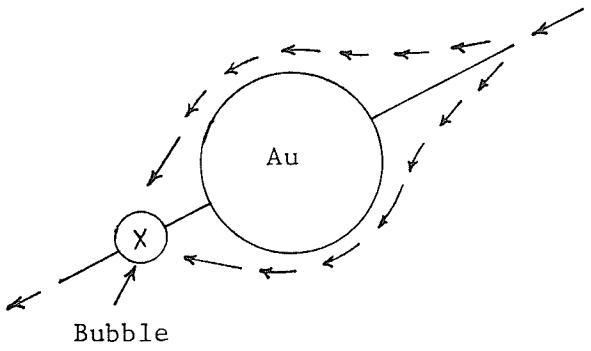


Figure 1.6: Domain Pattern Around an Implanted Hole [27]

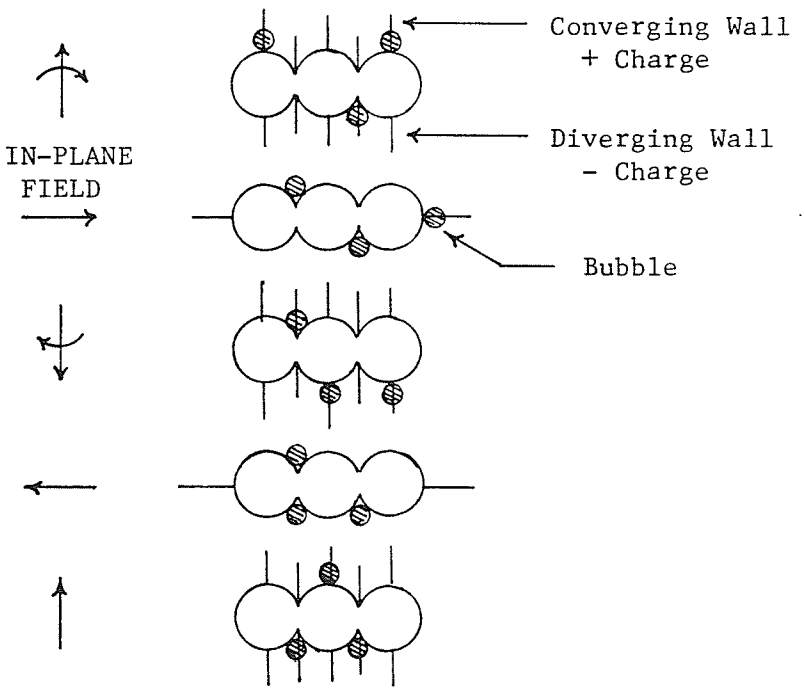


Figure 1.7: Propagation Using Charged Walls [27]

sheet will create similar magnetic dipoles as induced in the permalloy bar, as shown in Fig. 1.8. Figure 1.9 illustrates the use of two current sheets separated by an insulator to propagate bubbles. Bubbles are propagated from positions 1 through 4 when currents I1 through I4 are applied. There are three forseeable advantages of the current-access bubble memory: (i) elimination of speed limiting drive coils, (ii) increased storage density as a result of using smaller bubbles (due to relaxed lithographic requirements), and (iii) design flexibility introduced by eliminating the need to manipulate the entire bubble storage array.

1.3.3 Bubble Generation

Two methods are currently used to generate bubbles, i.e., to perform a write operation. One method utilizes a "seed" bubble from which a bubble is cut with the aid of permalloy patterns and/or current loops [31,32]. The second method generates a new bubble by nucleation with the aid of a current conductor and/or permalloy-conductor interactions. Nucleation is the reversal of the magnetization due to a local field concentration. One other method not used by commercial devices, due to its complexity, is microwave generation of bubbles [33]. This method relies on the ferrimagnetic resonance, that is set up by the frequency of the r.f. field, to generate bubbles.

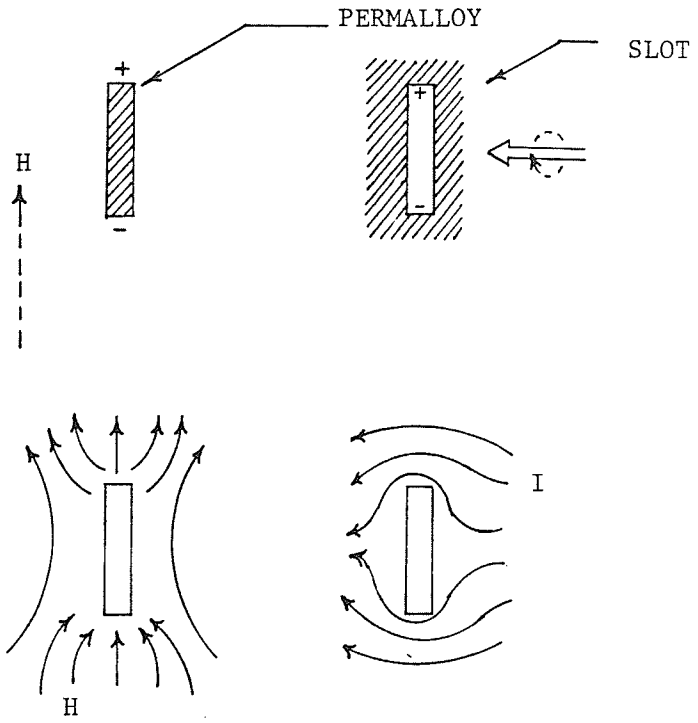


Figure 1.8: Dipole Field from Permalloy Bar and Current Conductor [24]

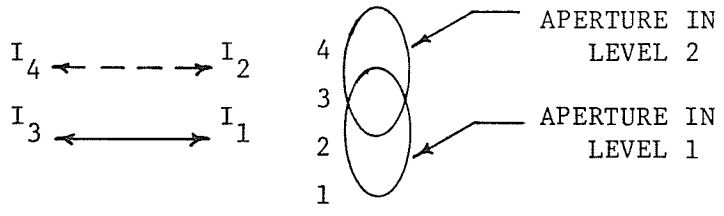


Figure 1.9: Dual Conductor Bubble Stepping Circuit [24]

1.3.4 Bubble Detection

Bubble detectors can be classified as either passive or active [20] . The passive detectors include the inductive loop [34], and the piezoelectric-magnetostrictive [35] detectors. The active detectors include the optical [36,37], galvanomagnetic [38], magneto-resistive and acoustic detectors [39].

The inductive loop technique is the simplest detection method achieved by placing a conductor loop over a domain which senses the flux change as the domain is collapsed. The piezoelectric-magnetostrictive detector achieves detection by monitoring an electrical output from a piezoelectric crystal strained by a magnetostrictive underlay which is subjected to the field of a bubble. In considering the active detectors, the following three methods are used for optical detection: (i) the Bitter method, (ii) the Kerr effect, and (iii) the Faraday effect. The galvanomagnetic (or Hall effect) detector is based on measuring the voltage induced in a semiconductor material, due to the bubble stray field interacting with moving charge carriers in an adjacent semiconductor. The acoustic detector uses surface acoustic waves to induce an electric signal in a conductor coupled to a magnetostrictive material magnetized by the stray field from a magnetic domain. The magneto-resistance phenomenon is also used for domain detection. This method senses the voltage change due to an increase in resistance of the detector material, when a magnetic domain magnetizes the material in the same direction as the current flowing through the detector. Figure 1.10 illustrates a chevron detector currently used by most commercial devices. Magneto-resistive detectors are preferred for the following reasons:

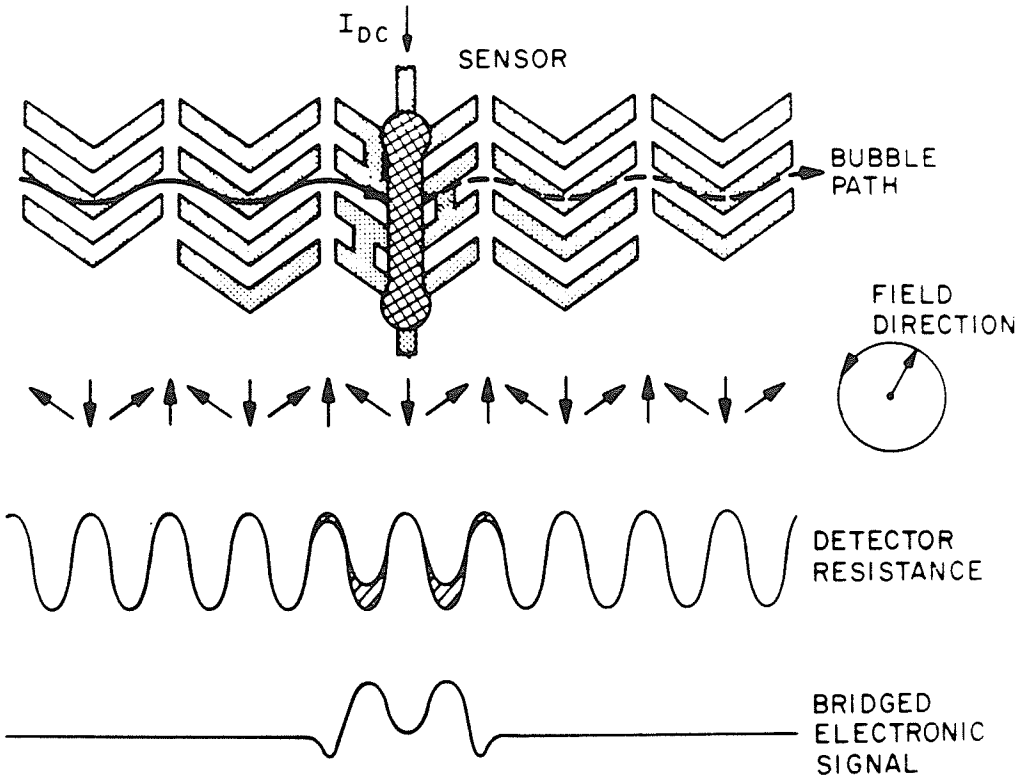


FIGURE 1.10: Chevron Detector [18]

a) The detector is made of the same material and at the same time as the permalloy propagation path, and

b) Chevron propagation elements lend themselves naturally to bubble amplification, thereby producing large output signals without any external sources.

1.3.5 Bubble Annihilators

Bubble annihilators and bubble generators are very similar in nature. The following three methods are used for bubble annihilation: (i) the hairpin annihilator, (ii) disk annihilator, and (iii) guardrail annihilator. The latter type is currently used in commercial devices. The guardrail annihilator simply propagates bubbles through a guardrail (protecting the operating device area from being contaminated by stray bubbles) into a waste area referred to as "bubble heaven". The hairpin annihilator uses a current loop to locally increase the bias field to the point of collapse of a magnetic domain (S_0 curve on the stability curve, Fig. 1.3). A disk annihilator functions in reverse to the seed generator in that bubbles are trapped onto a permalloy area with no means of re-entering the propagation path.

1.3.6 Bubble Transfer

The bubble transfer switch is the last functional element required in conventional MBM devices. The switch enables the routing of a bubble from one propagation path to another. Figure 1.11 illustrates the dollar sign transfer switch. Transfer is accomplished by means of an applied current in conjunction with the rotating field. The application of a

current makes point 2 (see Fig. 1.11), repulsive and creates a pole at point 3, attracting the bubble. At this point the in-plane field dominates bubble motion. This process performs a transfer from point 2 to point 4.

Several functional elements required for operational memories have been described. These elements play a key role in memory organization since the complexity and interconnection of these functions must be well defined. With the fundamental building blocks of MBM established, various organizations are outlined below.

1.4 MEMORY ORGANIZATION

Magnetic bubble memories have inherently slower access times in comparison to semiconductor memories. This can be attributed to the shift register type of architecture, whereas semiconductor memories have access times in the range of 50ns.

By using innovative structures, some of the inherent speed limitations of shift registers can be overcome. The shift register memory shown in Fig. 1.12 has evolved from the parallel shift register memory illustrated in Fig. 1.13, to the major/minor loop architecture of Fig. 1.14. The evolution process resulted from a need to minimize the number of drive lines, interconnecting pins, and peripheral circuits, while trying to decrease the access and cycle times.

Minor loops cannot be placed closer than eight bubble diameters because of limitations imposed by geometric constraints on the propagation circuit. Since the minimum bubble separation is approximately four bubble diameters (a distance at which bubble-bubble interaction becomes

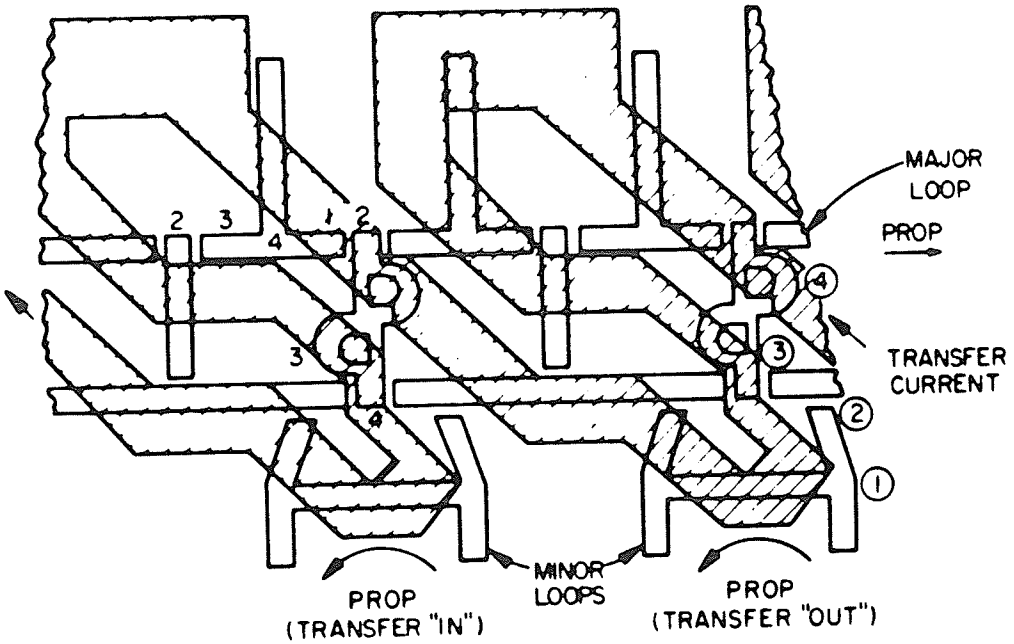


FIGURE 1.11: Dollar Sign Transfer Gate [18]

negligible), the constraints imposed on the major/minor loop configuration does not lend itself to an organization exhibiting an optimum output data rate.

To further increase the data rate by a factor of two, a block replicate architecture was devised. The architecture is illustrated in Fig. 1.15. Present commercial devices use one of the following three organizations: (i) serial loop [40], (ii) major/minor loop [41,42], or (iii) the block replicate architecture [43,44]. At present all commercial devices are serial in nature. Other non-serial approaches have been considered [45].

1.4.1 Serial Loop Operation

The serial loop memory illustrated in Fig. 1.12 is inherently simpler than either the major/minor loop or block replicate architecture. The serial loop memory is ideally suited for applications which are not time critical, but require the large storage capacity and/or simpler interfacing requirements attainable with this organization. Information is stored as a continuous string, resulting in a long cycle time. To improve cycle times, a major/minor loop architecture was devised.

1.4.2 Major/minor Loop Architecture

The major/minor loop architecture, illustrated in Fig. 1.14, is composed of a number of parallel minor loops used for information retention, and a common major loop for input/output functions. An important feature of this organization is that the number of bits stored in any minor loop is equal to the number that can be stored in the major loop.

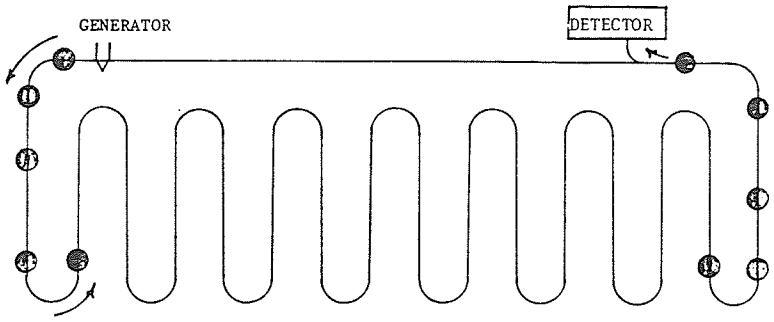


Figure 1.12: Serial Shift Register Memory Organization

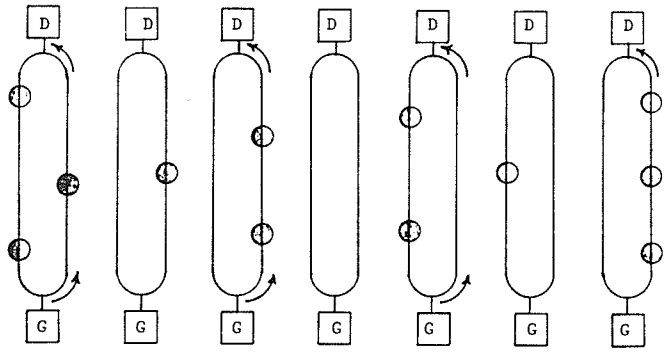


Figure 1.13: Parallel Shift Register Memory Organization

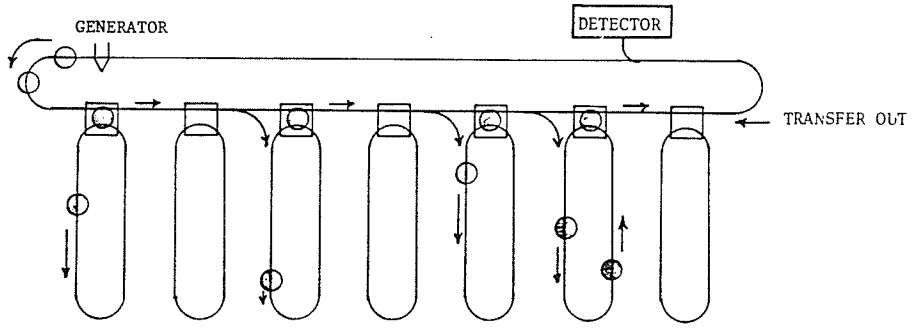


Figure 1.14: Major/Minor Loop Memory Organization

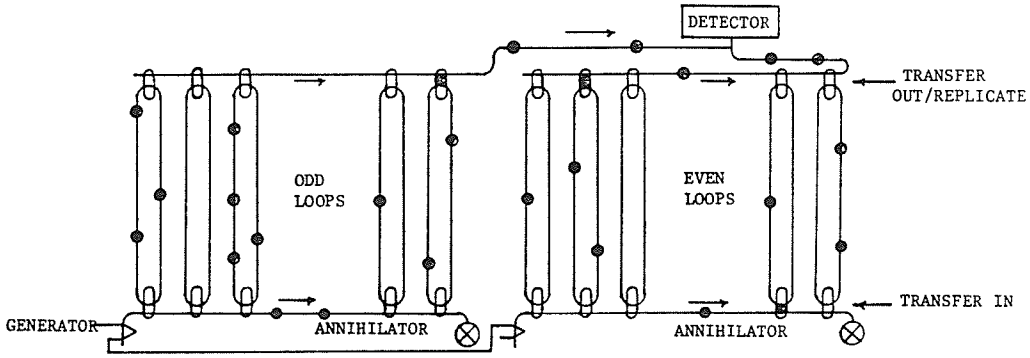


Figure 1.15: Block Replicate Memory Architecture

The above constraint provides the means for accurate data tracking as outlined below. To write data into the minor loops, data is generated and propagated serially on the major loop. A complete page or block of data is transferred into the minor loops when all the bits in the major loop are aligned with the tops of the minor loops. A bubble memory controller is used to provide control signals(eg. generate), and maintain tracking information for the MBM modules [46]. To initiate a read operation, the desired page of data is rotated in the minor loops until the block is aligned such that a parallel transfer into the major loop can be performed. After transferring the data into the major loop, the bits are rotated over a detector where the data can be annihilated or replicated. This results in either a destructive or nondestructive read operation. The size constraints of the major and minor loops (i.e., the length of the major loop must equal the length of the minor loop), insure that all paging information is retained when data bits are rotated back into the minor loops.

The speed improvements of the major/minor loop architecture in comparison to the serial loop architecture are illustrated by the following example. Consider both devices with a storage capacity of 64K bits and a rotating field frequency of 100 kHz. The average access time for the serial loop memory is 320 ms. Assume for the major/minor loop device a minor loop size of 1K bit, i.e., 64 minor loops. The average access time may be computed as follows:

Average time to locate desired page

= no. of bits/loop x [1/(2 x field frequency)]

= 5.12ms

Time to read first data bit

= no. of bit positions to the detector

x [1/(2 x field frequency)]

= n x 50μs n < 10

Access time = Average time + read time = 5.12ms

The above example illustrates that a significant increase in data rates can be achieved by using novel memory organizations.

The physical geometric limitations of device layout result in every second bit position of the major loop being left vacant. This drawback results in a data rate of one half the field frequency. To further increase the data rate a block replicate architecture is used.

1.4.3 Block Replicate Structure

In the block replicate structure, data is generated on both input tracks as shown in Fig. 1.15. The physical layout insures the transfer of odd and even bits into the odd and even storage loops respectively. During a read operation, data bits are interleaved such that the odd or even bits fill the vacant bit positions. This interleaving results in the full utilization of one detector. One should also note that in the illustrated structure the input track has been separated from the output track in order to further increase the average access time.

1.5 REDUNDANCY

The MBM is a serial device, hence any defect in a propagation path would render that path unusable. Devices based on the major/minor loop architecture are not rendered unusable by defects in the minor loop. To increase device yield, a MBM can be manufactured with redundant minor

loops which would compensate for any defective loops that may arise. This insures that the device storage capacity is still maintained and compensation for defective loops can be performed by the bubble memory controller.

During the bubble memory evolution process there has been a trend to decrease memory access times and increase storage capacities. This has been achieved, primarily through the use of novel memory organizations. Presently research is being conducted in the area of coilless propagation techniques. The drive coils are the primary elements impeding the operation of bubble memories at field frequencies greater than 200 kHz [47]. The speed limitation is due to both skin-effect losses in the coil windings and eddy-current losses in the metal package components. The following three devices are currently being considered to either increase storage density and/or eliminate drive coils: (i) bubble lattice, (ii) contiguous disk, and (iii) the dual conductor memory.

1.6 BUBBLE LATTICE MEMORIES

The bubble lattice memory structure offers the largest storage capacity. In this structure bubbles are stored in a compact hexagonal array as shown in Fig. 1.16. The distance between bubbles is determined by the mutual repulsive forces between them. Storage of information is achieved by encoding data in the wall states [48] rather than by the presence or absence of a domain. It has been found that, depending on the wall configuration, a domain will either propagate in a straight line or at an angle to the applied field. This difference leads to distinguishable states for data storage. Propagation of bubbles may be

achieved via field gradients produced by current carrying conductors [49]or by in-plane field interacting with a permalloy overlay [50]. Figure 1.17 illustrates the organization of a column accessed bubble lattice device. The translation of domains is achieved by field gradients interacting with a fraction of the array. This interaction in turn affects the entire array due to bubble-bubble interactions.

The increased storage capacity of bubble lattice devices is attributed to the following two factors:

a) The bubble-to-bubble spacing of 4 bubble diameters is eliminated, and

b) no restrictions, imposed by permalloy propagation patterns, are placed on the location of domains.

Other approaches to increasing the storage density of magnetic bubble memory are the contiguous disk memory, and the dual conductor memory. The propagation structures are described in Sec. 1.3.2.

Bubble lattice devices and contiguous disk memories are two techniques to improve the storage capacities of MBM. However, the approach in MBM design is not simply to increase the storage density, but also to attempt to increase the speed of the MBM. Currently, speed is limited by the drive coils. An alternative to drive coils is the dual conductor approach. Presently, only field access permalloy devices are available commmercially. It is conceivable that the next generation of bubble devices will utilize the contiguous disk propagation structure offering increased storage capacity. Recently an experimental device with a storage capacity of 11.5 megabits was demonstrated [51]. It can be concluded that contiguous disk devices offer at least an order of magnitude

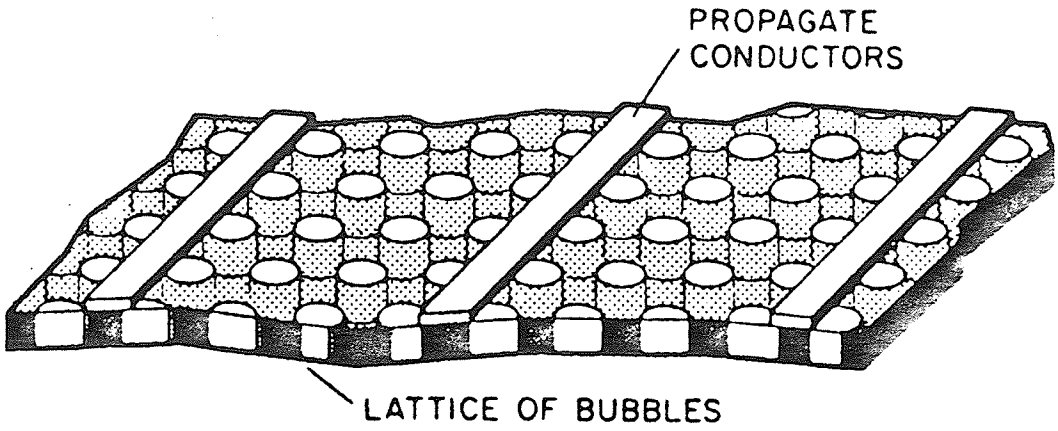


FIGURE 1.16: Bubble Lattice Structure [53]

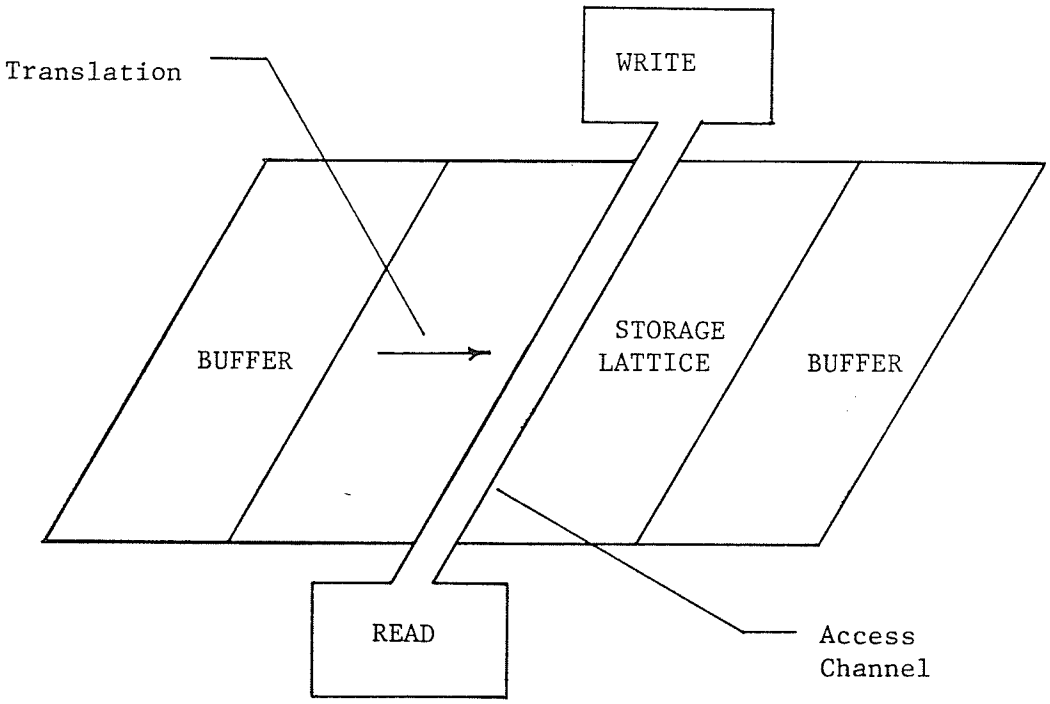


Figure 1.17: Column Accessed Bubble Lattice Structure [53]

greater storage density over field-access permalloy devices, and the impact of current-access devices is unforeseeble. Current access devices presently have power dissipation and electromigration problems in functional elements, which have to be resolved, before the viability can be examined.

Chapter II

SYSTEM DESCRIPTION

The high density and nonvolatility of magnetic bubble memories have enabled this technology to grow to its present state. Currently bubble memories have been successfully used for flight recorders [52], repertory dialers [53], and intelligent terminals [54]. Bubble memory technology is a comparatively young technology, hence designing MBM systems is not fully understood. To gain insight into designing with magnetic bubble memories and evaluating system reliability a bubble memory system was built. The following two chapters outline the hardware and software aspects of the system outlined below.

2.1 HARDWARE

A stand-alone bubble memory system is shown in Fig. 2.1. [55]. The system consists of a microprocessor with its program ROM and scratchpad RAM, internal buffer memory, serial asynchronous interface, bubble memory controller, function timing generator and bubble driving circuitry, and up to eight bubble memory modules.

A Motorola MC6802 microprocessor is the heart of the bubble memory system. The microprocessor and its associated circuitry provide all the necessary control and timing signals to the bubble memory controller. With a microprocessor acting as a system controller the system can be made stand-alone. The stand-alone system shown allows independence from

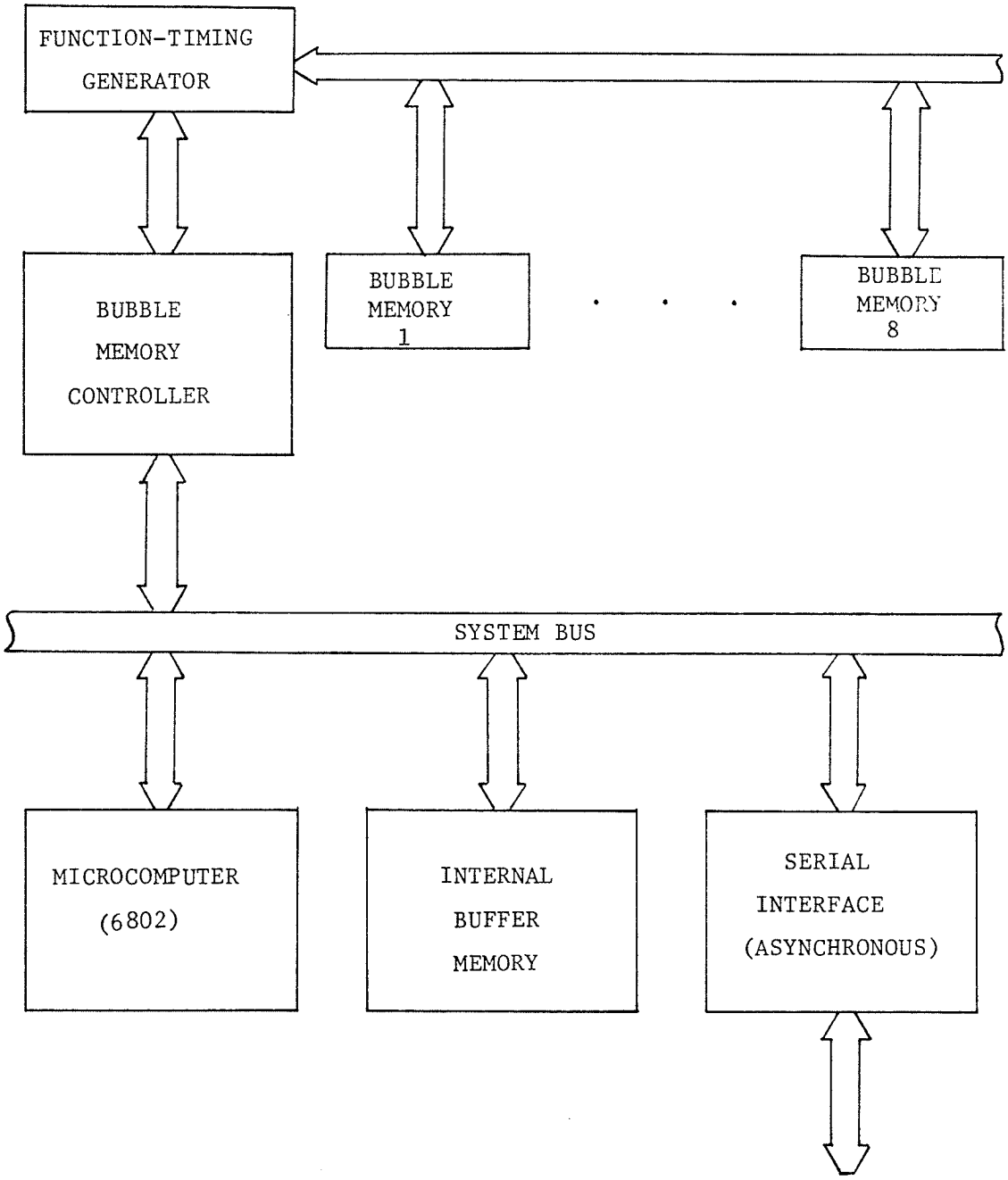


Figure 2.1: System Organization

any external architecture by providing a standard RS232 asynchronous interface. Hence, any external computer with an asynchronous interface can view the bubble memory system as an extension of its own memory. Communication is by means of a predefined protocol.

The MC6802 based system offers a maximum storage capacity of 92K bytes using eight TIB-0203 92K bit bubble memory modules. The processor controls all data transfers between the outside world and the bubble memory modules. Data transfer between the central processor and the bubble memory modules is through an intelligent bubble memory controller. The bubble memory controller in turn provides all the control signals (generate, replicate, transfer-in/out, etc.,) to the function timing generator and bubble driving circuits.

2.2 SOFTWARE

To facilitate design and understanding, a 4K byte monitor was written. The interface monitor is intended to demonstrate and test the operation of the microprocessor, and the bubble memory module within a microprocessor system. Interactive communications is achieved through a terminal with a RS232 interface. For development purposes, a terminal was connected to the asynchronous interface illustrated in Fig. 2.1. During stand-alone operation the terminal is not used and interaction with the system is established through a communication protocol.

During the design phase, various features were incorporated to facilitate system upgrading. Features include 10K bytes of RAM, direct memory access (DMA) capability for the MC6802, and a means of displaying internal status conditions via a peripheral interface adapter(PIA). By having

a 10K byte buffer memory and DMA facility, the system capabilities are greatly increased, i.e., the system architecture is more flexible. The capabilities also include synchronous communication, by means of a synchronous data link controller (SDLC), to interface to the external world. By using DMA to buffer data to the SDLC, the effective data rates from the bubble memory to and external computer can be increased past the 50 kHz data rate of the bubble memory module. During the DMA transfer mode, the microprocessor is used to fill buffer RAM and perform any intermediate processing of data that is desired by the user. Processing can include data compression or expansion through coding techniques. This would increase the effective storage capacity of the system, and/or allow algorithmic conditioning of data.

A stand-alone bubble memory system with capabilities of storing 92K bytes of data, interfacing to an external computer through a standard communication interface, and performing data conditioning, has been designed and implemented. The functional description of the hardware and software are presented in the following two chapters.

Chapter III

HARDWARE DESCRIPTION

A bubble memory system based on the TIB0203 bubble memory module and its associated circuits was designed and constructed. The elements of the system are described starting from the nucleus of the system (the bubble memory module). This description is followed with discussions of the controller, function timing generator(FTG), support circuits, and the computer. Detailed schematics of the hardware are included in Appendix A.

3.1 BUBBLE MEMORY MODULE

The bubble memory system architecture is centered around a TIB0203 bubble module [56,57]. A 14-pin dual-in-line package houses the bubble memory chip, the permanent magnet structure used to provide the necessary bias field, and two orthogonal drive coils used to generate the in-plane field. The package casing serves a dual purpose: (i) to house the above elements, and (ii) to provide a magnetic shield from external fields which can affect device operation. The module has a total storage capacity of 92304 bits using a major/minor loop architecture. The major/minor loop architecture illustrated in Fig. 3.1, consists of 157 minor loops and one major loop, each with a total storage capacity of 641 bits. This results in a total capacity of 100637 bits. The module has 13 redundant loops, resulting in a usuable capacity of 92304 bits. Con-

trol elements for generation, replication, annihilation and detection are located on the major loop.

Control functions such as generate, transfer, replicate and annihilate are performed by applying current pulses to the appropriate bubble circuit elements. Propagation of magnetic bubbles is achieved by a rotating in-plane field interacting with a permalloy overlay structure. Propagation of bubbles in the TIB0203 is by means of asymmetric chevrons as illustrated in Fig. 3.2. Asymmetric chevrons are used to provide unidirectional propagation with well defined attractive and repulsive poles. Another advantage of this propagation pattern is that it provides a very stable rest position for the domain, under static conditions. Detection is accomplished by means of dual magneto-resistive detectors. A dual detector structure is used in a differential mode, so as to reduce common-mode noise. The output signal is of the order of a few millivolts.

3.2 BUBBLE MEMORY CONTROLLER

A bubble memory controller handles all transactions between the microcomputer and the bubble memory modules. The TMS5502 is a MOS/LSI device with an internal architecture as shown in Fig. 3.3. The controller performs the following functions:

a) Major/minor loop handling. This handling involves keeping track of data during input/output transfer, generation and detection;

b) Bubble memory I/O handling. This involves performing parallel-to-serial and serial-to-parallel data conversions to and from the bubble module. The controller also ensures that the correct number of bytes have been transferred in or out of the bubble module; and

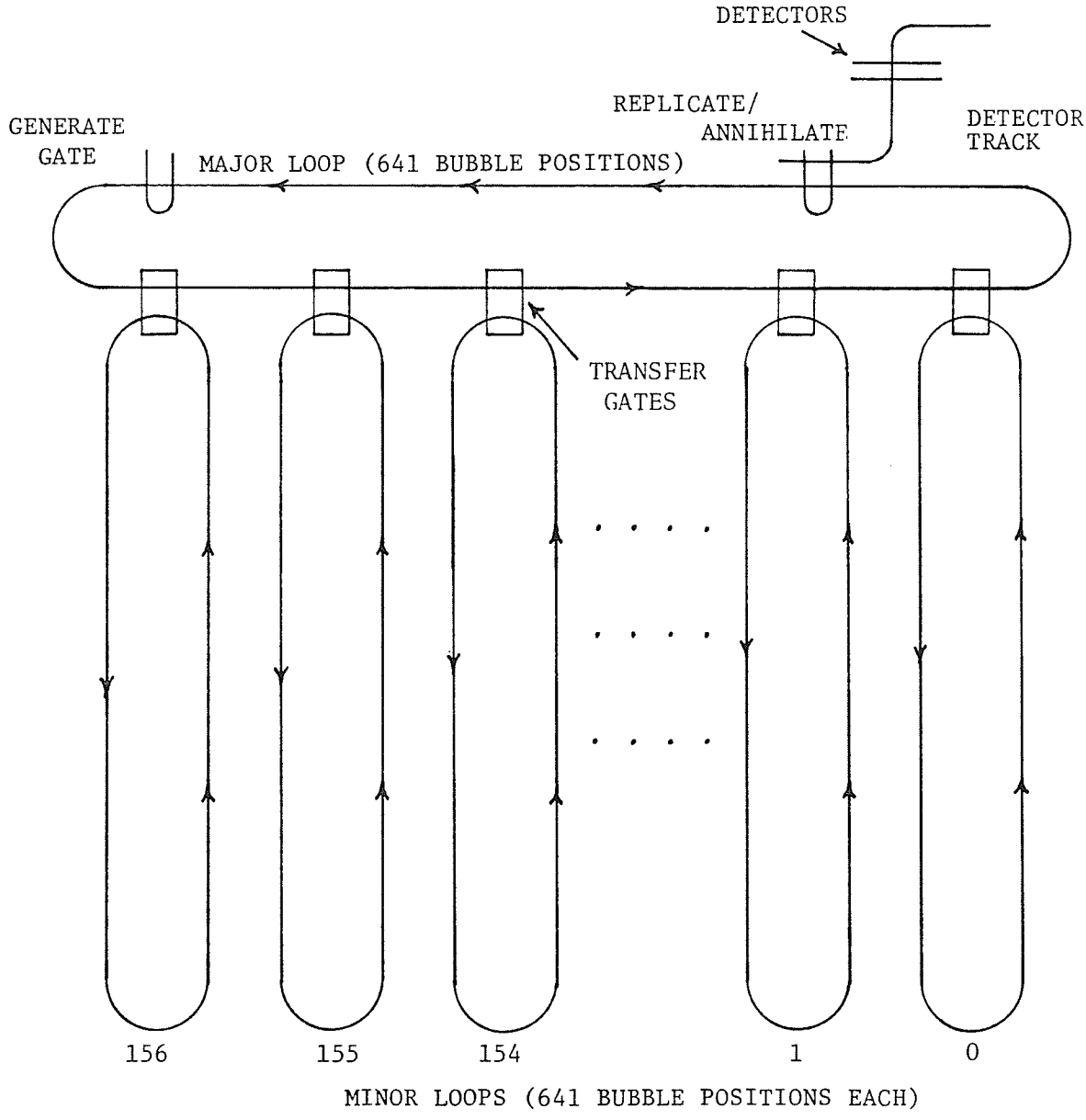


Figure 3.1: Bubble Module Organization

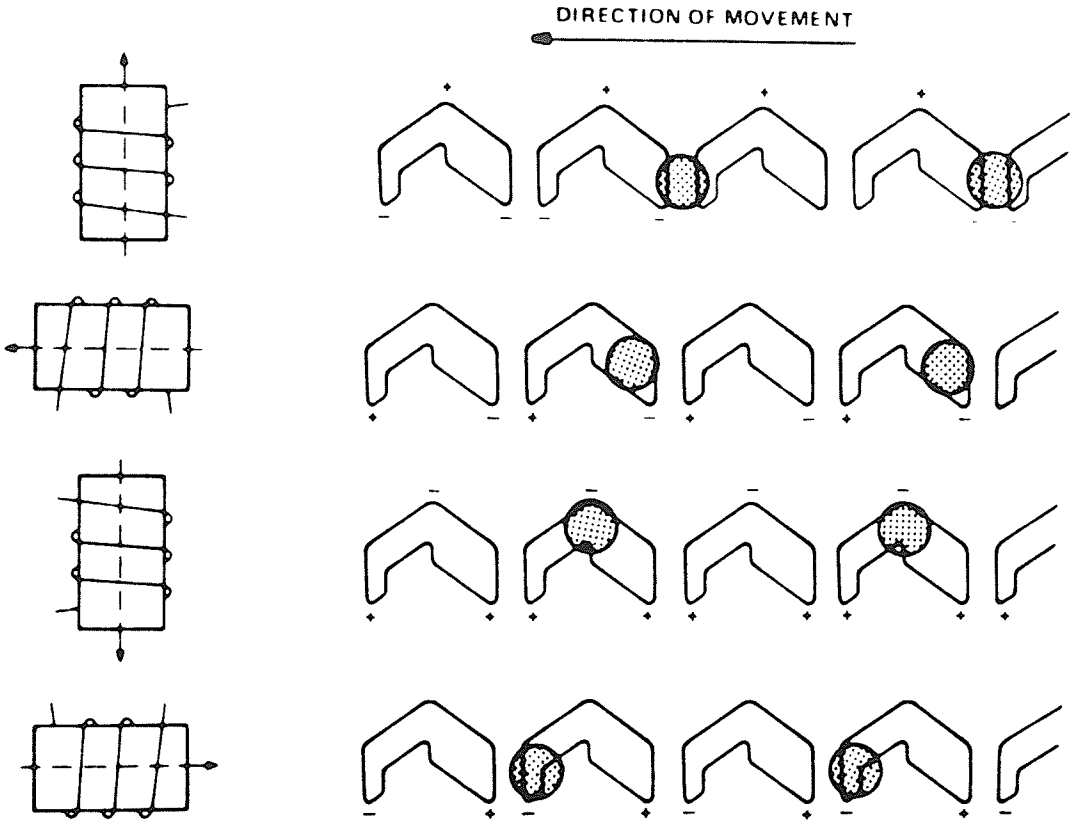


FIGURE 3.2: Asymmetric Chevron Propagating Pattern [50]

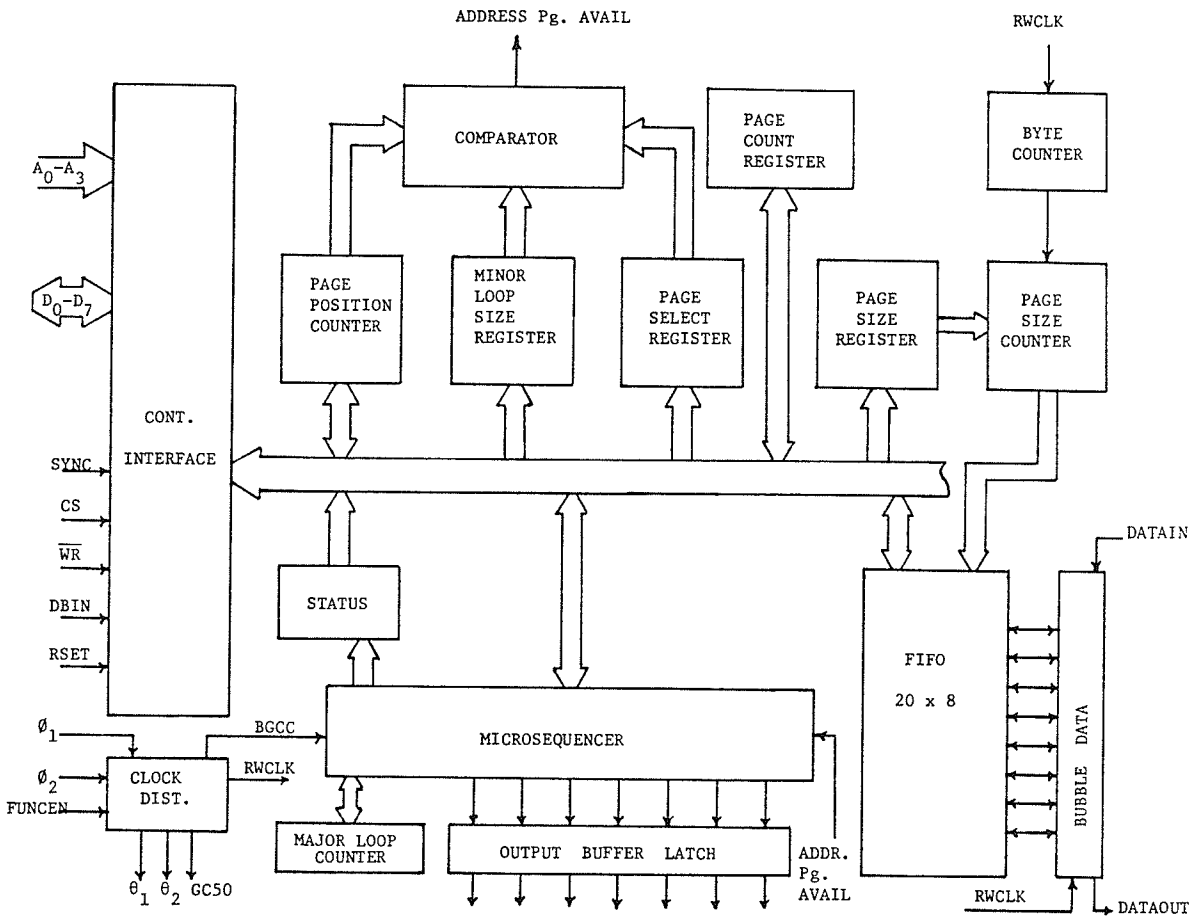


Figure 3.3: Bubble Controller Organization

c) Processor-controller I/O handling. The controller provides an intelligent interface between the processor and bubble modules.

The TMS5502 is a memory mapped device with 15 internal registers, occupying consecutive addresses. Each register performs a specific function, such as loading a command or loading the loop-size register , reading a status condition, or initiating a read/write operation.

The major/minor loop architecture of the bubble module lends itself to a page-oriented input or output of data. The page size for the TIB0203 is 18 bytes. This amounts to 144 bits which, together with the 13 redundant bits, match the number of minor loops (157 loops). A page position counter is used to ensure absolute tracking of page position. The page select register operates in conjunction with the page position counter during a read/write operation. The operation is such that, when a particular page is desired, an addressed page available status bit is set when the two registers match.

A 20 byte first-in first-out (FIFO) buffer memory is used to store a page of data during either a read or write operation. Depending on the mode of operation (single or multi-page) the FIFO becomes either a page oriented status and/or interrupt driven interface or a byte oriented interrupt driven interface between the system processor and controller. Transfers between the FIFO and the bubble module are accomplished through an 8-bit bubble data register which performs the necessary parallel-to-serial conversions. Serial shifting of the data stream is synchronized by a 50 kHz clock.

The following two modes of operation are available for data transfers between the controller and bubble memory modules: (i) single page mode,

or (ii) multi-page mode. The single page mode operation requires 12.8ms per operation. Each operation consists of transferring 18 bytes or one page of data to or from the bubble module. This operation may be either status driven or interrupt driven. The status bit is set or the interrupt is generated when the FIFO is full or empty, depending on the direction of data transfer. The multi-page mode requires 6.4ms for each operation. In this mode, data transfers are interrupt timed in that an interrupt is generated on a byte by byte basis. It can be noted that the multi-page operation provides for increased data throughput . The increase in data transfer rates is achieved by using a different storage technique. In the single-page mode, pages are stored in sequential order, whereas in the multi page mode, pages are stored 322 bit positions apart, thereby reducing access times.

The presence of redundant loops neccesitates a procedure for invalidating defective minor loops. This is achieved with external circuitry as illustrated in Fig. 3.4. The hardware is used to generate a gated signal which is provided to the byte counter, the function of which is to trace the bits shifted in or out of the bubble data register. A 512x8 bit PROM is used to maintain defective loop mapping information. Each output corresponds to a particular bubble device. A counter is used to clock the PROM in synchronism with the rotating field frequency. The appropriate PROM output is then gated to the byte counter to invalidate defective minor loops.

The bubble memory controller serves the following two functions:

a) To provide an interface between the system processor and the bubble modules, and

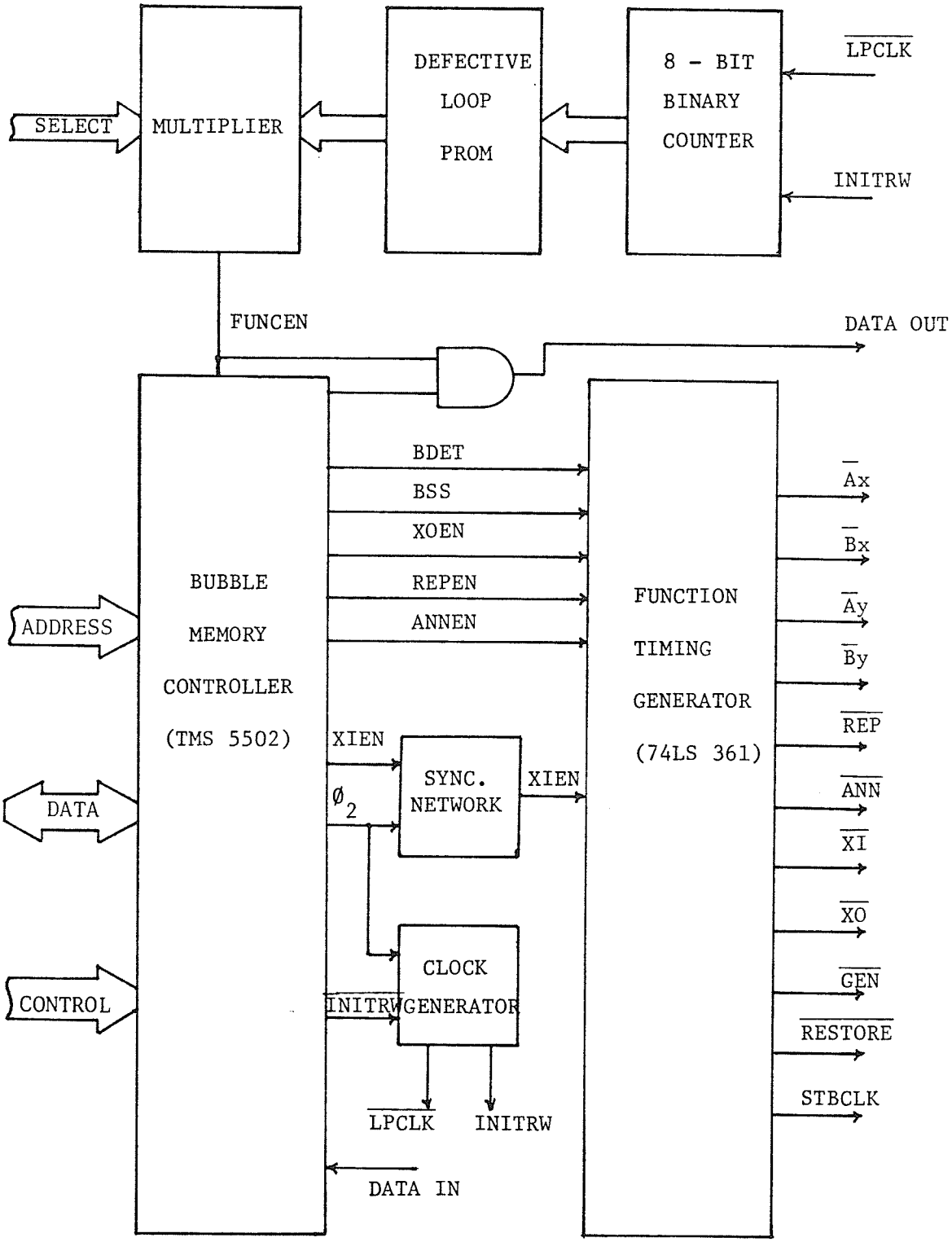


Figure 3.4: Function Timing Generator

b) To provide control signals required by the function timing generator (described later) to generate precise control timing signals required to enable the functional elements of the bubble module.

3.3 CONTROLLER INTERFACE REQUIREMENTS

The TMS5502 BMC has been designed to interface easily with the 9900 and 8080A microprocessors. An examination of the timing diagrams shown in Fig. 3.5 reveals a relationship between the two phase clocks required by the controller and the system clock of the MC6802 [58]. The relationship shown in Fig. 3.5 shows that the controller can be directly interfaced to the 6802 if clock synchronization between the two devices can be achieved. Figure 3.6 illustrates the synchronization technique and the generation of the required clock and timing signals.

An Intel clock generator and driver (8224) [59] provides the master clock signals. The feature that makes this device ideally suited for this application is its ability to generate the high-level 2 MHz $\phi 1$ and $\phi 2$ required by the BMC and a 2 MHz TTL level signal synchronized to the high-level $\phi 2$. The 2 MHz TTL clock is converted to a 4 MHz clock, required by both the function-timing generator and the 6802. Synchronization between the processor and controller is achieved by inverting the 4 MHz clock required by the 6802 through a CMOS gate. This technique provided the exact delay required to establish synchronization.It was found that the controller operated properly with clock signals deviating by as much as 50 ns from the exact delay.

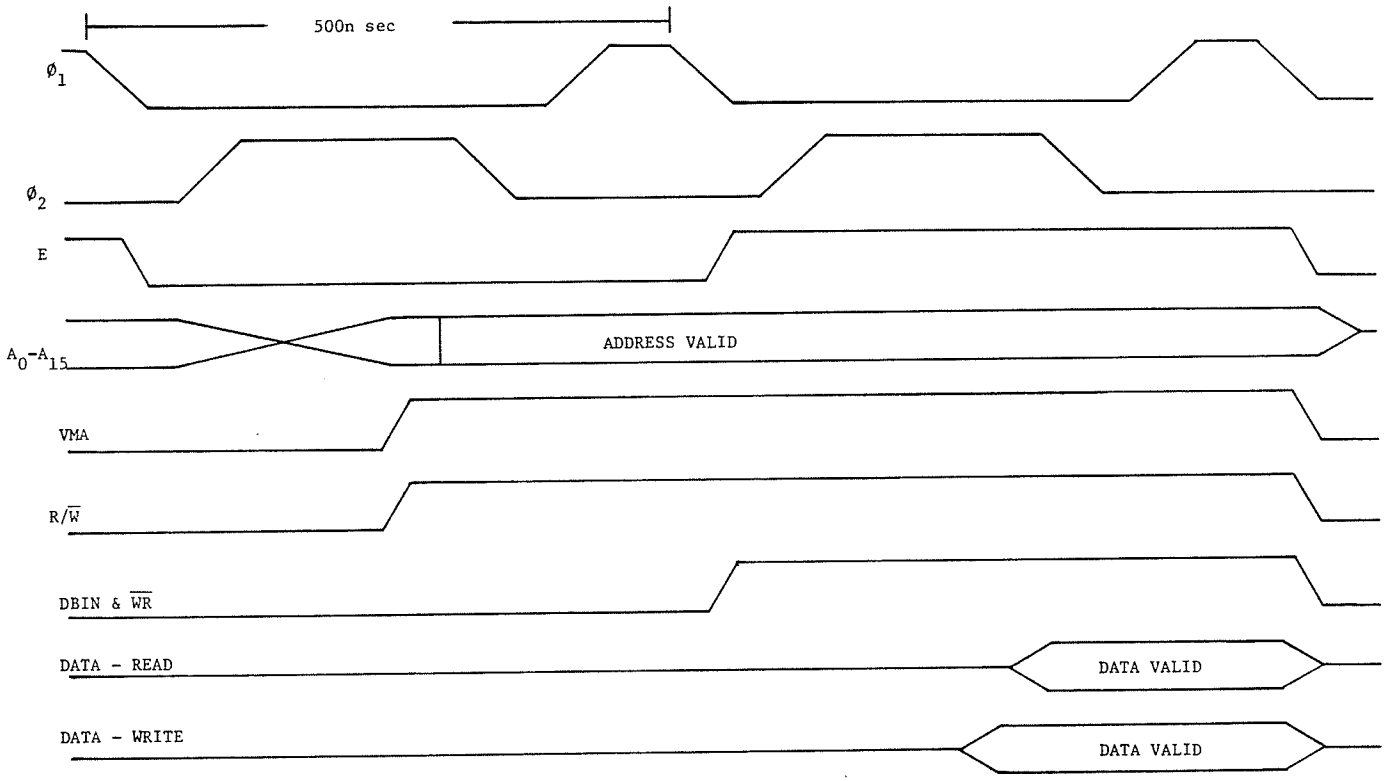


Figure 3.5: Timing Diagram of the 6802 and 5502

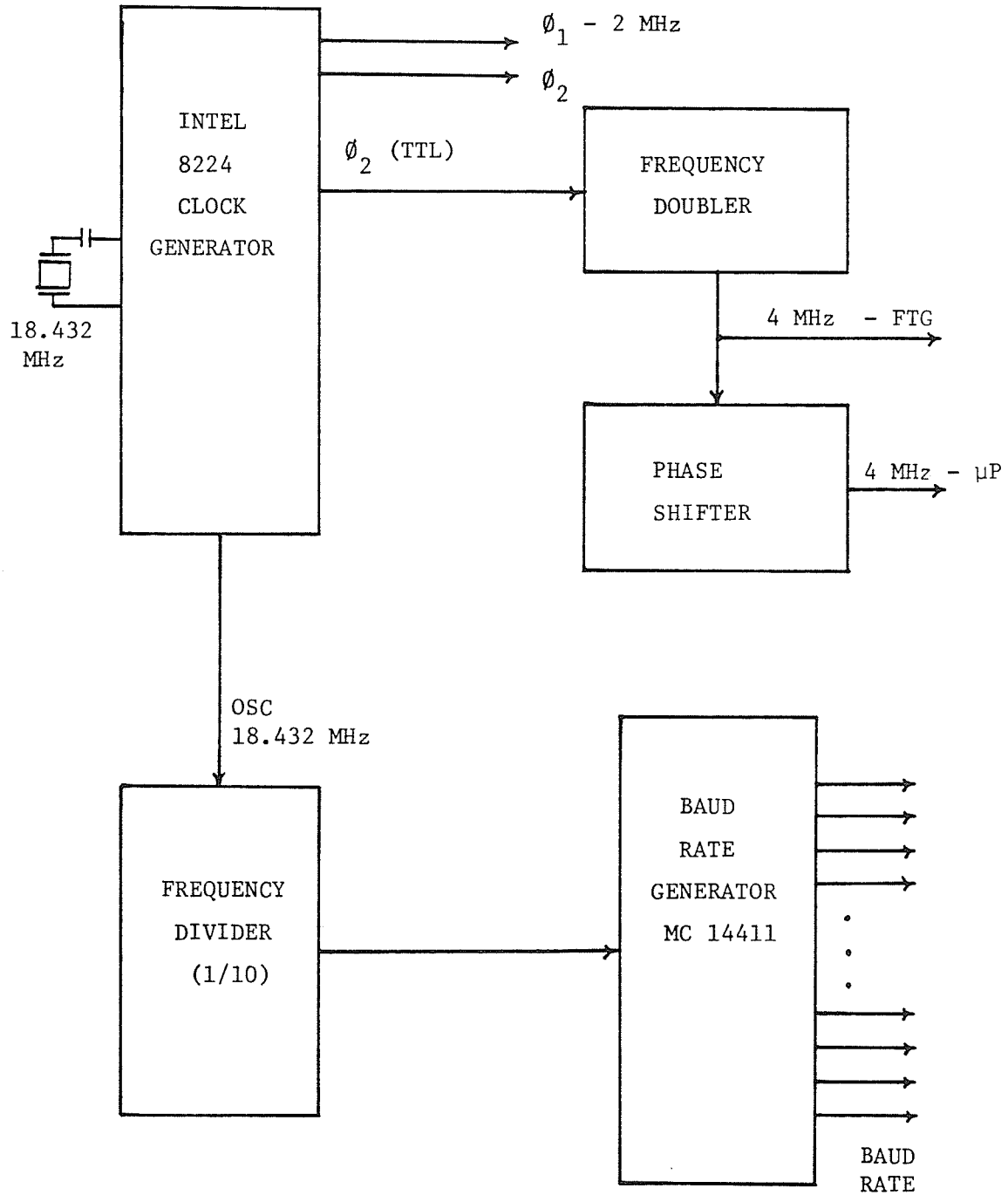


Figure 3.6: Generation of Clock and Timing Signals

3.4 FUNCTION TIMING GENERATOR (FTG)

The function-timing generator (74LS361) is a 22-pin DIP monolithic integrated circuit. The bubble memory system depends on the FTG (illustustrated in Fig. 3.4), to initiate and synchronize the operation of various interface integrated circuits. The FTG provides the precise control timing necessary to operate the function driver, coil driver, and sense amplifier elements shown in Fig 3.7. The function timing generator also provides data protection during power transient conditions by forcing the coil drive outputs to a logic high when the supply voltage drops below approximately 3.5 volts.

3.5 BUBBLE SUPPORT CIRCUITS

The bubble memory module requires a variety of circuits to generate control pulses and amplify bubble detector output signals. To provide the necessary control signals a sense amplifier, bubble memory function driver, and coil driver have been developed as integrated circuits available [60].

The SN75281 sense amplifier circuit is illustrated in Fig. 3.8. It consists of an internally ac-coupled amplifier, a threshold sense circuit, a D-type flip-flop, a three state output, and an RC biasing network. The operation of the circuit shown in Fig. 3.8 is such that the large common-mode signal produced by the rotating in-plane field, coupled with high frequency noise, and an offset voltage is passed through the RC network. A signal of typically three millivolts peak to peak results. This signal is converted to a TTL-compatible signal before being passed to the BMC. Application of the RESTORE input results in the input

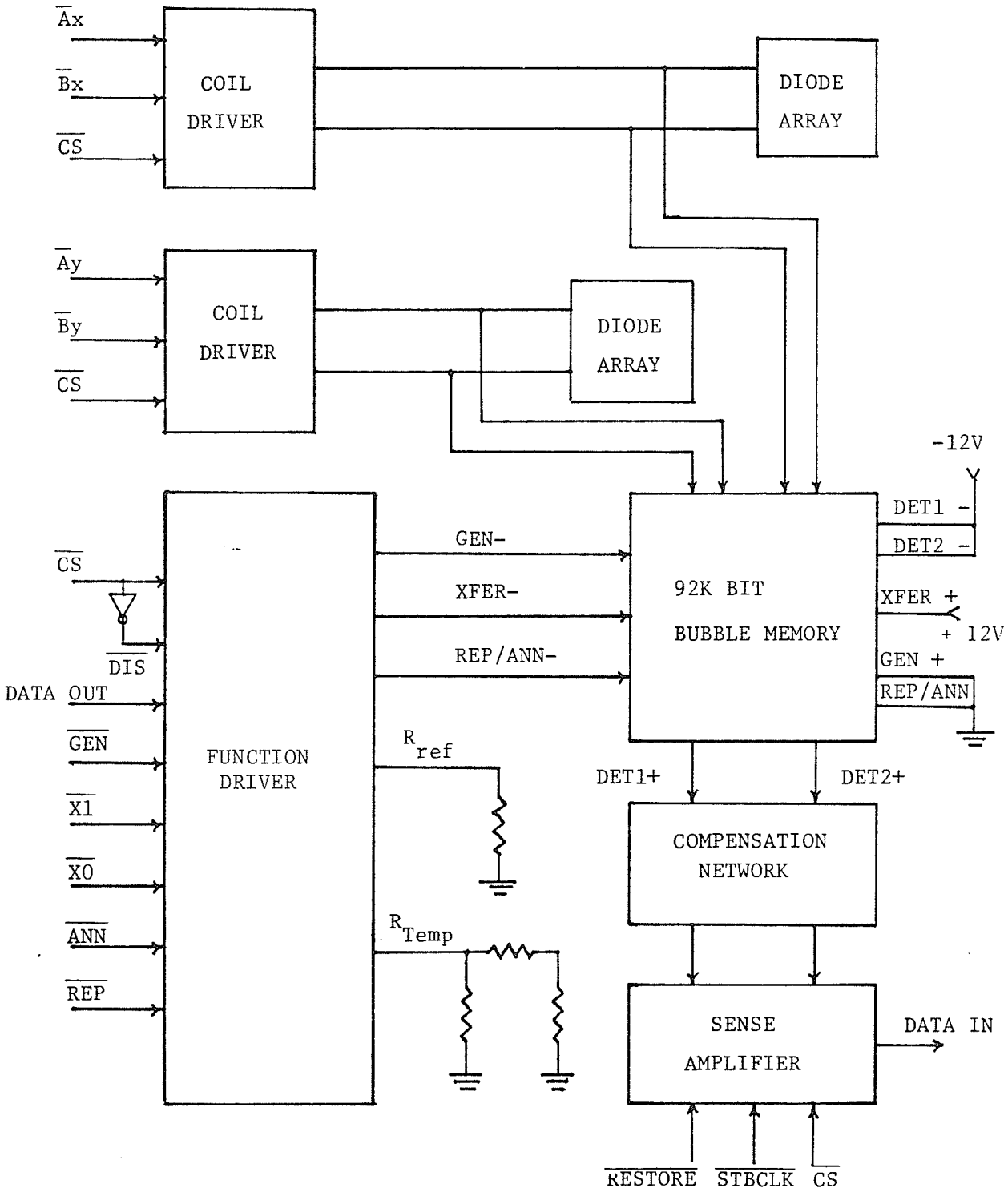


Figure 3.7: Bubble Module with Drive Circuits

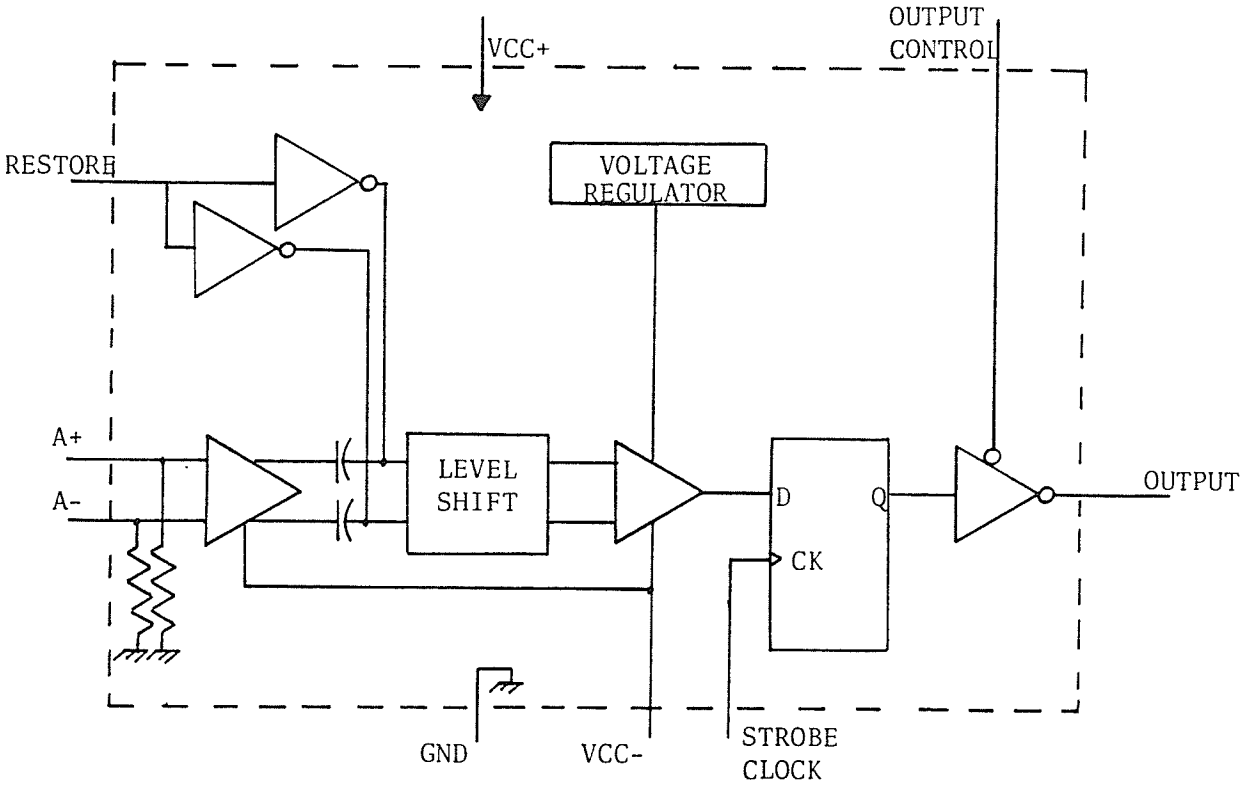


Figure 3.8: Sense Amplifier Internal Architecture

signal being clamped and amplified before it is strobed into the D-flip-flop. Gating of the output is via the output control input.

The function driver (SN75380) shown in Fig. 3.9 provides the constant current pulses required for generation, replication, annihilation, and transfer-in/transfer-out operations. Note that transfer in and transfer out are parallel operations. The parallel operation is achieved as follows: the minor loop data is transferred into vacant bit locations in the major loop and the major loop data is transferred into the minor loop. Figure 3.9 also illustrates that the same output is used for both the replicate and annihilate functions. These functions use current pulses of different amplitude or timing, to either transfer bubbles in or out of the major loop, or annihilate bubbles. The function driver also provides a reference output that is used to provide temperature tracking of the generate current. Temperature tracking is required since it is easier to generate bubbles at higher temperatures. This implies that the current must be limited with increasing temperatures.

A pair of bubble memory coil drivers (SN75382) in conjunction with Schottky diode arrays are used to produce the 100 kHz rotating in-plane magnetic field. The configuration and drive waveforms are shown in Fig. 3.10. The triangular drive waveform is generated by applying a step voltage to the driving coils. The triangular waveform in turn is integrated to produce a current ramp. When the step voltage is removed, the current is commutated by two diodes and allowed to ramp down to zero. When the current reaches zero, an opposite polarity step is applied to produce a current ramp of opposite polarity. Note that the frequency of the rotating field is twice that of the bubble-in and bubble-out rates.

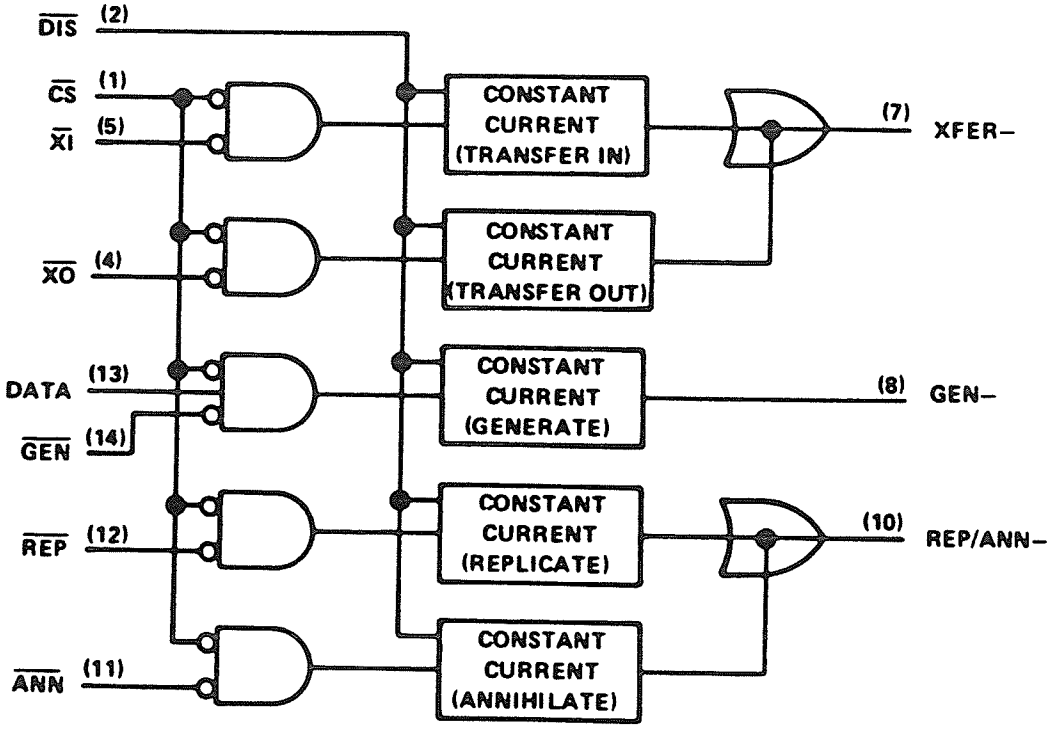


FIGURE 3.9: Function Driver Block Diagram [60]

3.6 INTERNAL MICROCOMPUTER

The internal computer is based on the 6802 microprocessor. The system architecture is standard and consists of a 4K byte monitor and 128 bytes of scratchpad RAM. The RAM is internal to the microprocessor and 32 bytes may be retained under standby power conditions. The 6802 microprocessor does not have DMA capabilities. DMA in a halt steal mode can be achieved by placing three-state buffers on the address and data buses. These buffers are enabled by the bus available signal of the 6802. In this mode of operation, the processor is halted for one machine cycle during which one DMA transfer is performed.

3.7 ASYNCHRONOUS INTERFACE

An EIA RS232C interface standard is used for data transmission. The interface uses the 6850 asynchronous adapter (ACIA) for data transfers between the communication channel and the microprocessor. Baud rate timing is switch selectable from 110 to 9600 baud. The timing is obtained from a baud rate generator, the MC14411, as shown in Fig. 3.5.

3.8 POWER SUPPLY REQUIREMENTS

A fundamental requirement for the system is that the power supply regulation must be maintained for approximately 13 milliseconds after power failure or shutdown. This allows time for any data in the major loops to be returned to the minor loops and time to rotate the minor loops to address zero. In this way data integrity and paging information are maintained.

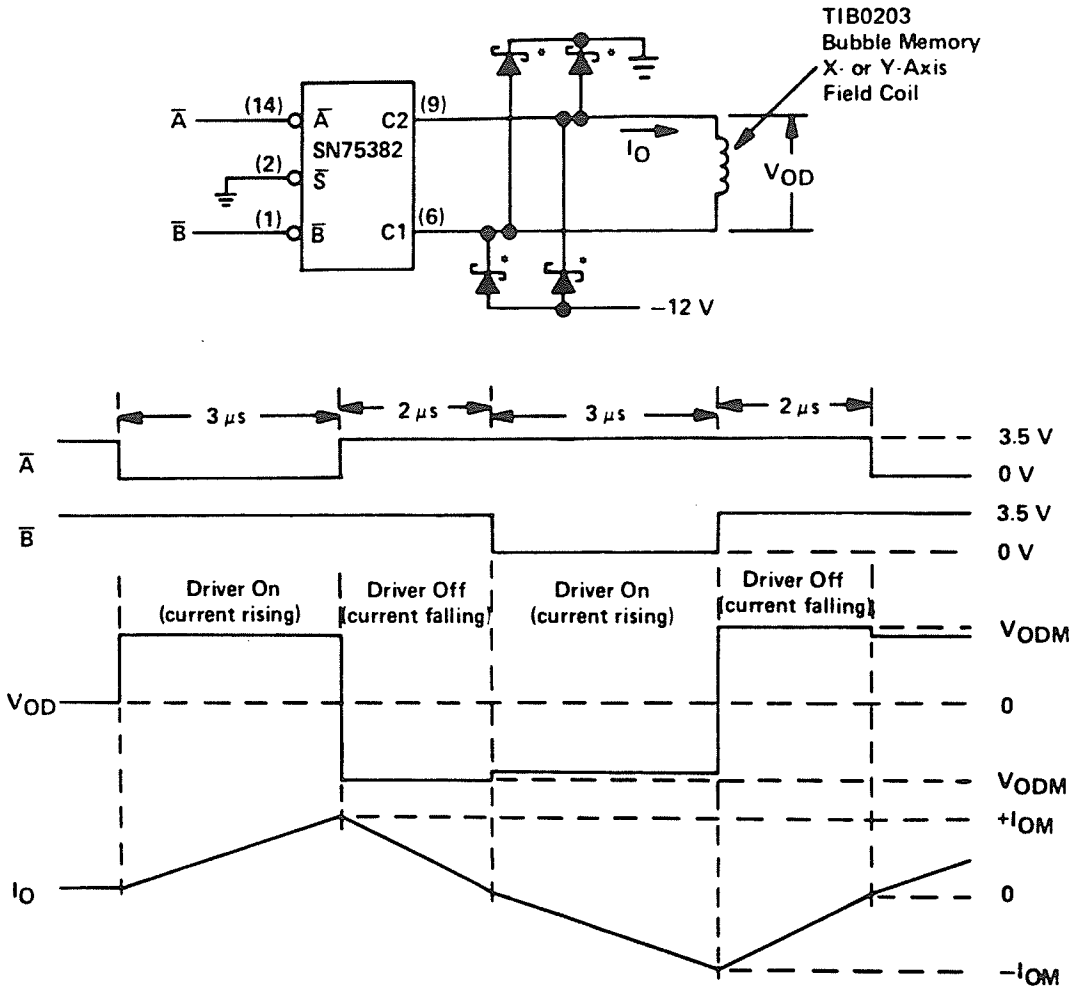


Figure 3.10: Generation of Bubble Drive Fields [60]

The necessary power supply was designed and built. A schematic of the power supply is included in Appendix A. It is capable of maintaining power for up to 2 seconds. This flexibility was built in to allow for complete data integrity if the system is modified to use the buffer RAM as discussed earlier.

The power supply requirements calculated using typical device requirements are summarized as follows:

+5v supply total current requirements = 4.225 A
-5v supply total current requirements = 0.201 A
+12v supply total current requirements = 0.346 A
-12v supply total current requirements = 0.230 A

3.9 SYSTEM DESIGN CONSIDERATIONS

When designing a bubble memory system using the family of circuits discussed, attention must be paid to the following items:

a) The bubble memory controller provides a gated 50 kHz clock (GC50) intended for clocking the defective loop map PROM. However, there are timing problems in GC50 due to design faults in the controller. The timing problem can be corrected by one of the following two approaches:

   i) the timing can be corrected as illustrated in Fig. 3.4, or

   ii) the GC50 signal can be gated to the $\phi 2$ output from the controller by means of a NOR gate.

b) A delay must be provided before reading the status bit indicating that a byte has been read or written. The delay should be approximately one machine cycle.

c) One must ensure that all chip select inputs to the bubble support circuits and the multiplex channel select inputs (Fig. 3.4) are

latched during the entire transaction between the bubble module and controller.

d) A nominal duration for transfer-in enable is 10 us. Following an access to absolute page 640_{10} however, the transfer in remains active into the next transfer. The following two approaches may be used to correct this condition:

   i) Issue an initialize command to the controller after accessing page 640_{10}, or

   ii) Avoid using page 640_{10}, resulting in a total storage capacity of 92160 bits.

e) Coil drivers are highly susceptible to noise spikes on the -12 volt supply line. A voltage spike in the order of -14 volts will destroy the device. The placement of a 12 volt transient suppressor along with the capacitors recommended by the manufacturer on the supply line will sufficiently minimize noise spikes.

3.10 BUBBLE MEMORY MODULE LAYOUT CONSIDERATIONS

The following guidelines are effective in reducing noise induced into the sense amplifier and detector circuits, when laying out bubble modules.

a) Use large ground planes to minimize noise effects such as switching noise and high frequency noise generated by the coil drive and function drive circuits.

b) Use separate ground returns for the coil drive and sense amplifier circuits to reduce induced noise.

c) Use high frequency decoupling and low frequency filtering on all supply lines. Filter capacitors should be placed at the board edge and at coil driver supplies.

d) Use wide coil drive lines to reduce IR drops created from large peak currents.

e) Keep detector signal lines between the bubble memory and the sense amplifier as short, close together, equal in length, and as distant from all other signal lines as is practical.

A bubble memory system was designed and implemented using the TIB-0203 92K bit magnetic bubble memory module. The system architecture is such that the total storage capacity is 92K bytes. A MC6802 microprocessor in conjunction with a bubble memory controller is used to provide an intelligent interface to the bubble modules. The controller provides the necessary timing and control signals to the bubble support circuits(FTG, sense amplifier, function driver, etc.,). The microprocessor is also the heart of the microcomputer which controls the interface between the bubble modules and the external communications port. Hence, with the aid of the microcomputer, an external terminal can communicate with the bubble memory for either testing purposes or as a memory expansion to the external system.

Chapter IV

SOFTWARE DESCRIPTION

The bubble memory system software is a 4K byte interactive monitor written in 6800 assembly code. The interactive monitor is intended to demonstrate and test the operation of the bubble memory module within a 6802 based system. Flowcharts and the assembly listing of the software are included in Appendix B.

The monitor has functions found in standard CPU monitors. The software is stored in four 2708 EPROMS residing at memory locations F000-FFFF Hex. The stack area and scratchpad RAM area required by the monitor consist of 128 bytes of RAM starting at address 0000 Hex. The locations of the various memory mapped system elements are shown in Fig. 4.1. Interactive communication is achieved through a terminal with an EIA RS232C interface.

4.1 POWER-UP INITIALIZATION

Figure 4.2 illustrates the sequence of events after either a power-up or hardware reset. After a reset, the ACIA, PIA, BMC, and bubble modules are initialized. The ACIA is configured for a transmission frame composed of 7 data bits, 1 parity bit, and 1 stop bit. The PIA initialization configures the ports as outputs except for the LSB of port A. The PIA is intended for monitoring power-fail and for outputing status conditions. Initialization of the BMC involves setting up the minor loop

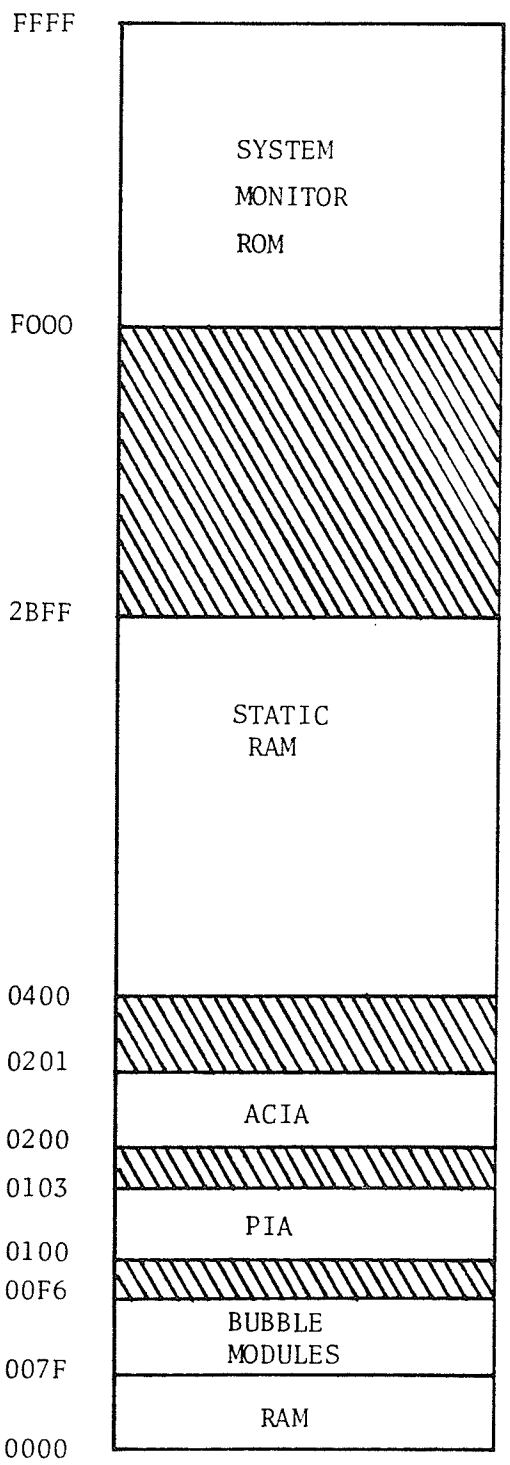


Figure 4.1: System Memory Map

size and page size registers, and issuing a controller reset. The bubble module must also be initialized on power up. This operation is performed to destroy any bubbles that may have been present in the major loop. Flowcharts for the above operations are included in Appendix B.

4.2 COMMAND DESCRIPTIONS

In writing the software, a modular approach was taken. This approach allowed the greatest degree of flexibility in building up a complete monitor. This concept is illustrated in Fig. 4.2 with the implementation of the command structure. The modular approach is easier to implement than some type of coding scheme, particularly when the number of commands are limited. This approach also makes any software expansion easier to implement. Table 4.1 shows the full command list.

Each command is executed by entering a single letter or number followed by any required parameters. Commands interacting with the bubble module are self explanatory in operation. A more detailed command description is presented below with the discussion of the tests left to the next chapter.

4.2.1 C, Continuous bubble I/O

Syntax:

```
C[ONTINUOUS BUBBLE I/O ]
BUBBLE I/O IN ASCII OR HEX?
ENTER <A> OR <H> <enter desired letter>
START PAGE: <starting page address>
END PAGE: <end page address>
DATA PATTERN: <enter ASCII character or Hex byte>
```

The data pattern entered is stored into the bubble memory starting at the page specified in Hex. The verify subcommand (described later) is

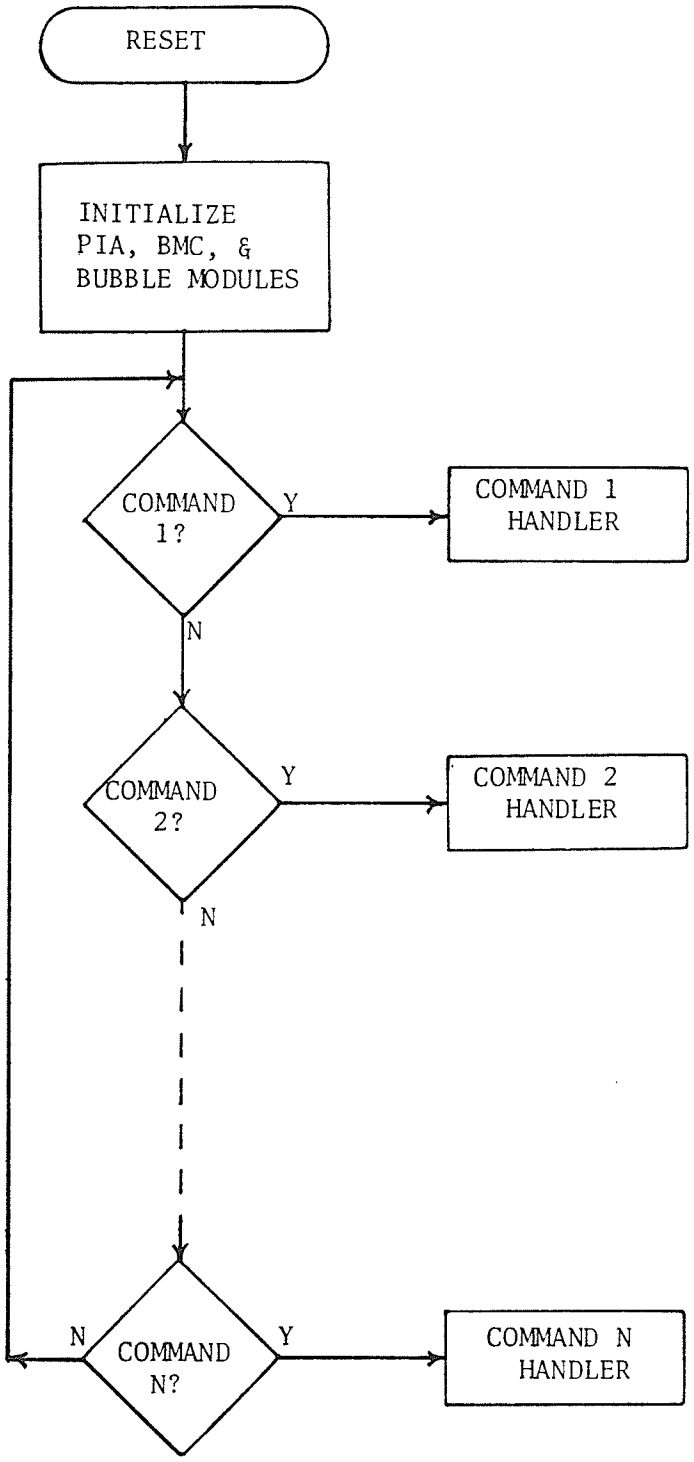


Figure 4.2: Monitor Main Program

TABLE 4.1

System Command List

Input	Operation
C	Continuous bubble I/O
D	Memory dump to terminal
E	Examine memory location
F	Fill bubble memory with a selected 8 bit data pattern
G	Execute user program
H	Help facility - displays available system commands
I	Initialize BMC
L	Display contents of bubble module (ASCII or HEX)
R	Read desired page from bubble memory (ASCII or HEX)
S	Set up for data input mode (half duplex)
T	Transfer from RAM-to-bubble or bubble-to-RAM
V	Verify number of r/w generations
1	Test 1, generator and detector testing
2	Test 2, major loop failure mode testing
3	Test 3, major loop nearest neighbour testing
4	Test 4, minor loop failure mode testing
5	Test 5, minor loop nearest neighbour testing
6	Fill selected minor loop entirely with "1s" or "0s"

also available under this command as shown in Fig. 4.3. The "C" command is used as a hardware debugging aid when the bubble control signals have to be monitored to determine proper device operation.

4.2.2 D, Dump memory

Syntax:

D[UMP] <starting address> <end address>

The "D" command dumps the contents of memory starting from the specified address. The starting and end address are specified in Hex. The data is displayed both in Hex and ASCII.

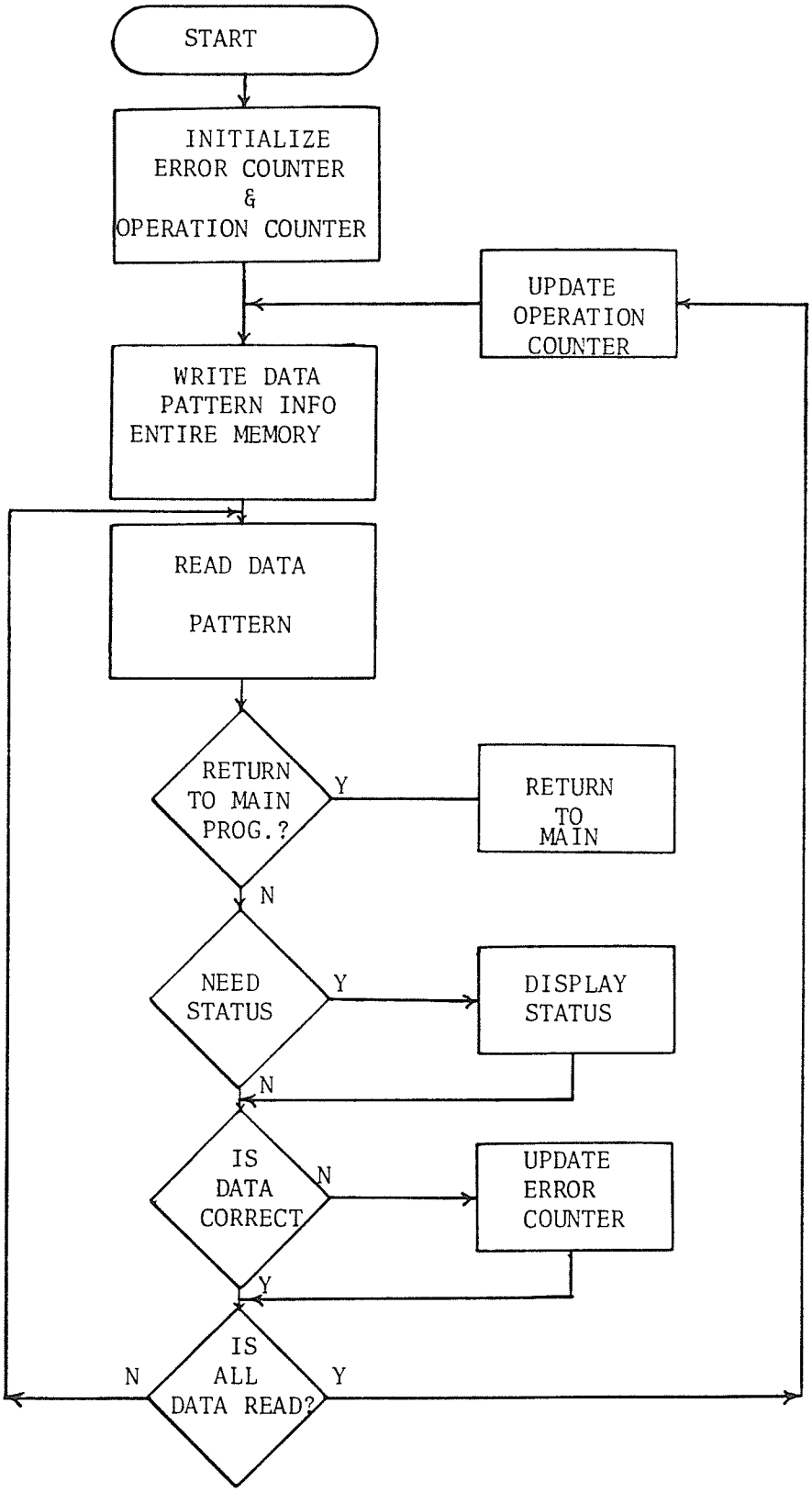


Figure 4.3: "Continue" Command Structure

4.2.3 E, Examine memory location

Syntax:

E[XAM] <starting address>

The "E" command is used for examining memory in the microprocessors work space. The following three subcommands are available under the EXAM command: (i) next, (ii) back, and (iii) change. A carriage return is used to examine the next location, and entering the letter "B" allows examining of the previous location. Entering a "C" results in the following output:

XXXX XX <enter new data in Hex>

where XXXX and XX represent the address and data respectively. The main program can be re-entered by entering a space or the "ESC" character Figure 4.4 illustrates the subcommand structure.

4.2.4 F, Fill bubble memory

Syntax:

```
F[ILL ]
BUBBLE I/O IN ASCII OR HEX?
ENTER <A> OR <H> <enter desired letter>
START PAGE: <starting page address>
END PAGE: <end page address>
DATA PATTERN: <enter ASCII character or Hex byte>
```

The "F" command is used to initialize a block of bubble memory with a user specified data pattern starting from the specified address. The data pattern may be entered in ASCII or Hex to allow input flexibilty . During execution of the fill operation the following message **WORKING** will be displayed.

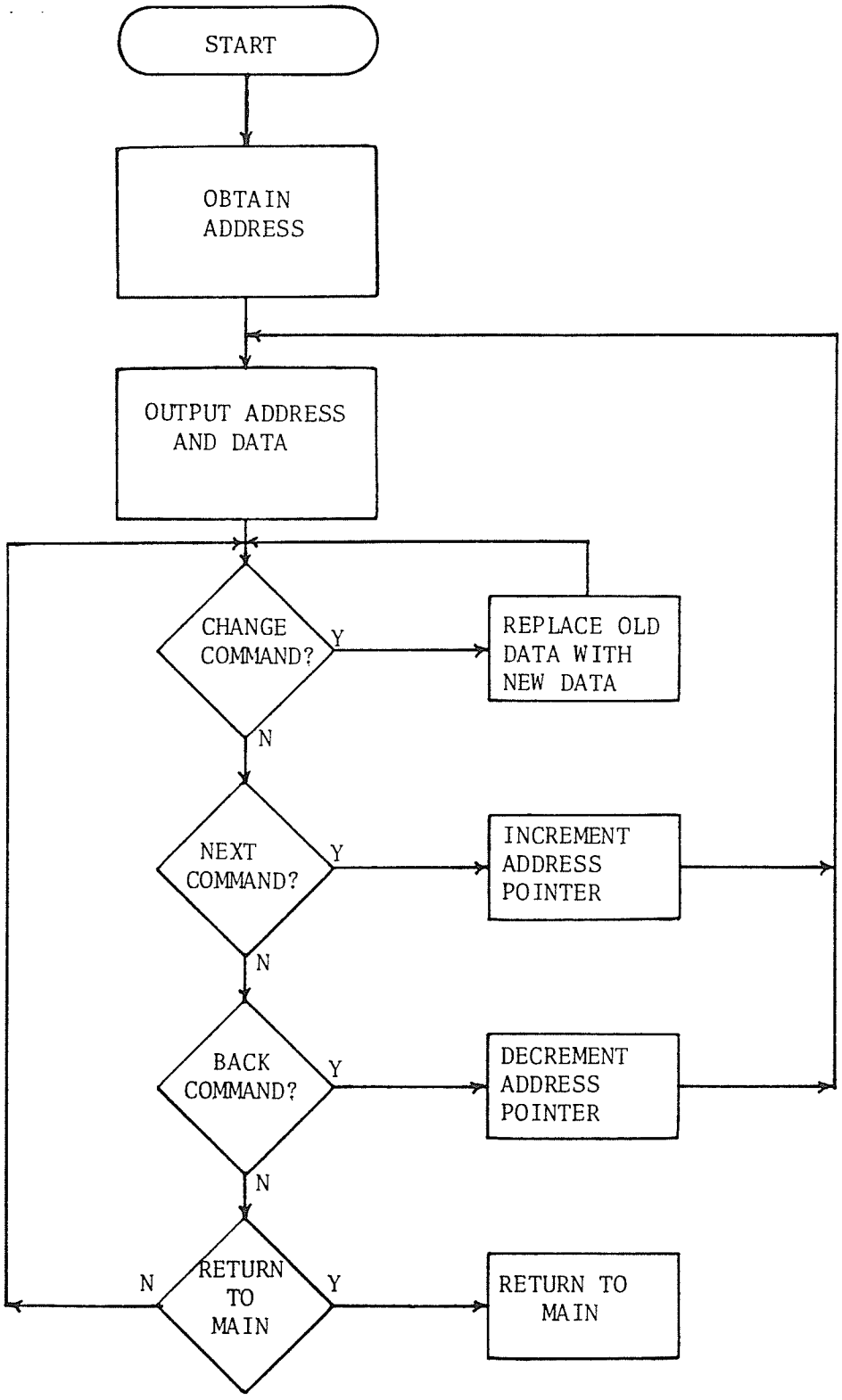


Figure 4.4: "Exam" Command Structure

4.2.5 G, Execute user program

Syntax:

```
G[O FROM: ] <starting address>
```

This command is used to initiate execution of a program starting from a specified address.

4.2.6 L, Display contents of bubble module

Syntax:

```
L[IST ]
BUBBLE I/O IN ASCII OR HEX?
ENTER <A> OR <H> <enter desired letter>
START PAGE: <starting page address>
END PAGE: <end page address>
```

This command lists a block of data specified by the starting and end addresses. The output is either in ASCII or Hex.

4.2.7 R, Read desired page from bubble memory

Syntax:

```
R[EAD ]
BUBBLE I/O IN ASCII OR HEX
ENTER <A> OR <H> <enter desired letter>
START PAGE: <starting page address>
```

The "R" command is similar to the "E" command in operation. This command is used to examine the contents of the bubble module a page at a time. The following three subcommands are available: (i) next, (ii) back, and (iii) change. Entering an "N" will output the next page and a "B" will display the previous page. The display is wrap-around, i.e., the next page after 640_{10} is page zero. The change command acts on one page of data (18 bytes) at a time. Data can be changed by entering the

new data or can be left unchanged by entering an up-arrow character. After the entire page has been changed or partially changed, the new page is written, read, and re- displayed for visual verification. The command structure is illustrated in Fig. 4.5.

4.2.8 S, Set up for data input mode

Syntax:

```
S[DATA INPUT MODE ]<W input data /> DATA TRANSFERRED
```

This allows for unconditional data input from a terminal or external source. The block of data is preceeded with a "W" and ended with a "/" character. This configures the communication channel to half duplex mode.

4.2.9 T, Transfer from RAM-to-bubble or bubble-to-RAM

Syntax:

```
T[RANSFER ]
FROM RAM TO BUBBLE -ENTER <B> Or
ENTER <R> IF TRANSFER FROM BUBBLE TO RAM==><enter desired letter>
```

If the response is "B" the output prompt is as follows;

```
START ADDRESS: <enter start address>
END ADDRESS: <enter end address>
TRANSFER PAGE NO.: <enter transfer page address>
```

and if the response is "R" the prompt becomes;

```
START PAGE: <enter start page address>
END PAGE: <enter end page address>
TRANSFER RAM ADDRESS: <enter transfer address>
```

This command is used to copy a block of data from system memory to bubble memory or from bubble memory to system memory. The data block defined by the start and end address is transferred starting from the specified address.

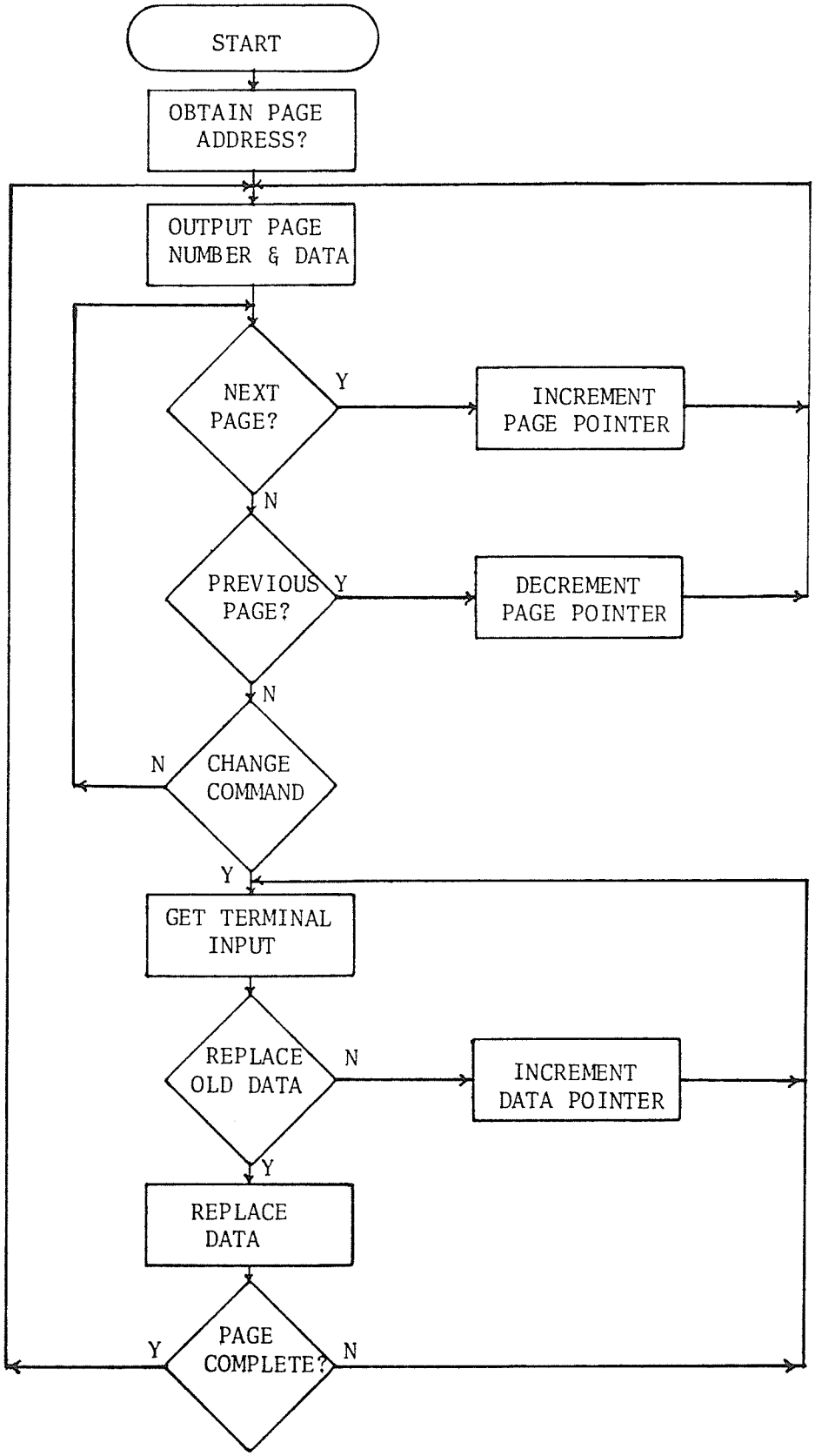


Figure 4.5: "Read" Command Structure

4.2.10 V, Verify number of R/W operations

Syntax:

```
V
NO. OF HARD ERRORS = XX
NO. OF SOFT ERRORS = XXXX
NO. OF R/W OPERATIONS = XXXXXXXX
```

This command is used for monitoring system status during a test procedure or during the continuous bubble I/O operation. Hard and soft errors are defined in chapter 5. Note that all numeric operands are in Hex.

4.3 SOFTWARE FEATURES

Features of the software include error checking and speed enhancement during bubble input/output operations. The following types of error checking are performed: (i) checking for invalid commands in both command structure levels, (ii) checking for valid Hex inputs, i.e., to ensure that the entry is between 0 and F, and (iii) checking that a bubble page address does not exceed 640_{10}. If the bubble page address exceeds 640_{10} the controller remains permanently active, and must be reset. This reset results in a loss of paging information. Speed enhancements are due to both software and hardware considerations. The speed enhancement was obtained by memory mapping the BMC to an address of 80-FF Hex. This facilitates using the direct addressing mode of the 6802. The use of direct addressing over extended addressing allows the software to execute faster since most operations require one less machine cycle.

The software also features a means of returning to the main program without using a hardware reset. Entering the "ESC" character will return control to the main program. This avoids the initialization procedure performed during power up or hardware reset.

A 4K byte interactive monitor serves as the operating system for the bubble memory system. Interactive communication is achieved through a terminal and a standard asynchronous communication interface. The monitor is essentially a modified CPU monitor with enhancements to accomodate the internal organization of the bubble memory module.

Chapter V

SYSTEM TESTING PROCEDURES

Magnetic bubble memories have been commercially available since 1977. This young technology requires various testing schemes and test equipment to determine its reliability [61,62,63]. This chapter discusses the testing that was performed to evaluate the effects of bit patterns on data integrity, and the effects of prolonged operation on device control signals. Bit pattern testing was based on failure mechanisms discussed by Cheng and Davis [64]. The following three failure modes are possible in bubble devices: (i) collapse, (ii) strip-out, and (iii) failure to cross a gap region. The results of the above tests are presented in the next chapter.

5.1 PATTERN SENSITIVITY

The three failure modes mentioned above are bit pattern sensitive. Device function sensitivity is due to the following three factors:

a) Forces on bubbles due to local magnetic fields produced by adjacent bubbles,

b) Forces on bubbles due to fields produced by current control elements, and

c) Interaction of bubbles with detector elements.

Since a bubble memory has a limited number of functions (generate, replicate, transfer in/out, etc.,) localized to small areas of the device, a limited number of bit patterns can be used for sensitivity testing. This limitation minimizes the number of test patterns. In examining pattern sensitivity, the following factors have to be considered:

a) Bubble-to-bubble interaction is strongest within a radius of four bubble diameters. Since the field strength is reduced within four bubble diameters, only short bit patterns are required for approximating the worst case condition, and

b) Since device functions are localized and dependent on each other, specific bit patterns can be used for multiple tests. For example, the write operation affects primarily the generate and swap-gate elements. A read operation can be used to test the replicate gate and magneto-resistive sensor elements. The propagation path, used for data storage can be tested by propagating a string of bubbles across each propagation element.

5.1.1 Pattern Sensitivity of Propagation Elements

The three failure modes (collapse, strip-out, and failure to cross a gap region) are sensitive to different types of data patterns. Failure of propagation elements is dependent on the following three factors:

a) The shape of the permalloy pattern,

b) The strain induced by conductor crossing elements, and

c) The type of defect in the propagation element.

The shape of the permalloy element determines stable bubble positions during propagation. The strain due to conductor crossing elements can

result in deformation of the propagating element. This deformation could result in non-uniform bubble propagation. Defects in the propagation element can result in strong localized fields, induced by the rotating in-plane field, which could result in propagation failure.

Collapse, strip-out, and failure to cross a gap region are each possible under different bit pattern combinations. A bubble is more prone to collapse when surrounded by other bubbles since the bubble-to-bubble interaction is then at a maximum. A strong bubble-to-bubble interaction results in an increased bias field as seen by the bubble located in the center. Propagation failure can also occur when a bubble is displaced from the propagation path by forces exerted on the bubble by other bubbles behind it. This is a repelling force due to the non-uniform field distribution as seen by the leading or trailing bubble. Failure due to strip-out can occur in a region where a bubble is isolated from other bubbles.

Bit patterns most suitable for testing all three failure modes are of the form 1111111100001000100 and its complement. A bit pattern of the form 11011010101010 and its complement should be used to test failure due to bubble-to-bubble interaction. These bit patterns were used by Cheng and Davis. To promote failure, several such patterns should be staggered around each storage loop.

5.1.2 Pattern Sensitivity at Generation Elements

It was stated earlier that bubble-to-bubble interaction increases appreciably if the bubbles are seperated by less than four bubble diameters. To determine the effect of the generator element on data bits, the bit pattern 11110001000101 should be used.

5.1.3 Pattern Sensitivity of Detector Elements

Detection of domains is by means of the magneto-resistive effect. This method relies on the stray field of a bubble to alter the net resistance of the detector. Since neighbouring bubbles can contribute to the stray field, the same bit pattern used for testing the effects of the generator element can be used to exercise the detectors.

5.1.4 Pattern Sensitivity at Swap and Replicate Gates

Pattern sensitivity at the swap and replicate gates is dependent on the interaction of bubbles on the input track with that of the bubbles in the storage loop. To fully test the two operations the following combinations are sufficient:

a) Swap a bubble with a bubble,

b) Swap a bubble with no-bubble, and

c) Complement of both (a) and (b).

5.2 TIB-0203 DEVICE TESTING

On the basis of the above discussion, the following five tests were performed to determine bubble memory reliability in a system environment:

a) Effects of the generate element on bit pattern and the effects of the same bit patterns on the output signal,

b) Major loop failure mode testing,

c) Major loop nearest neighbour testing,

d) Minor loop failure mode testing, and

e) Minor loop nearest neighbour testing.

5.2.1 Test 1:Generator and Detector Effects

The object of this test was to determine the effects of the generate current on the bit pattern in the vicinity of the generate element. The bit pattern generated is also suitable for determining the effects on the detector output signal due to bubbles in the neighbourhood of the detector. A bit pattern of the type 1110001000101 was used as a basis for this test. Since the TIB-0203 is a page oriented device with a page length of 18 bytes, a bit pattern comprised of the following fundamental pattern F1150EEA00 Hex was used.

This test is invoked by entering a "1" after the prompt is displayed. The test is performed by writing an 18 byte data pattern constructed from a repetitive combination of the 5 byte data pattern shown above into page zero. A read cycle is used to perform the following two functions:

a) Provide a means of verifying the data generated, and

b) To see the effects of the bit pattern on the output signal.

Figure 5.1 shows the flowchart of the test software. The results of the test and all subsequent tests are presented in the next chapter.

5.2.2 Test 2:Major Loop Failure Mode Testing

This test was performed to determine the reliability of the major loop. Based on the previous discussion of failure modes, an 18 byte data pattern consisting of the fundamental data pattern FF08800F77 Hex was used. The testing scheme is similar to that of test one, in that the data pattern is continuously written and read to determine any errors during propagation. This and subsequent tests would only be meaningful

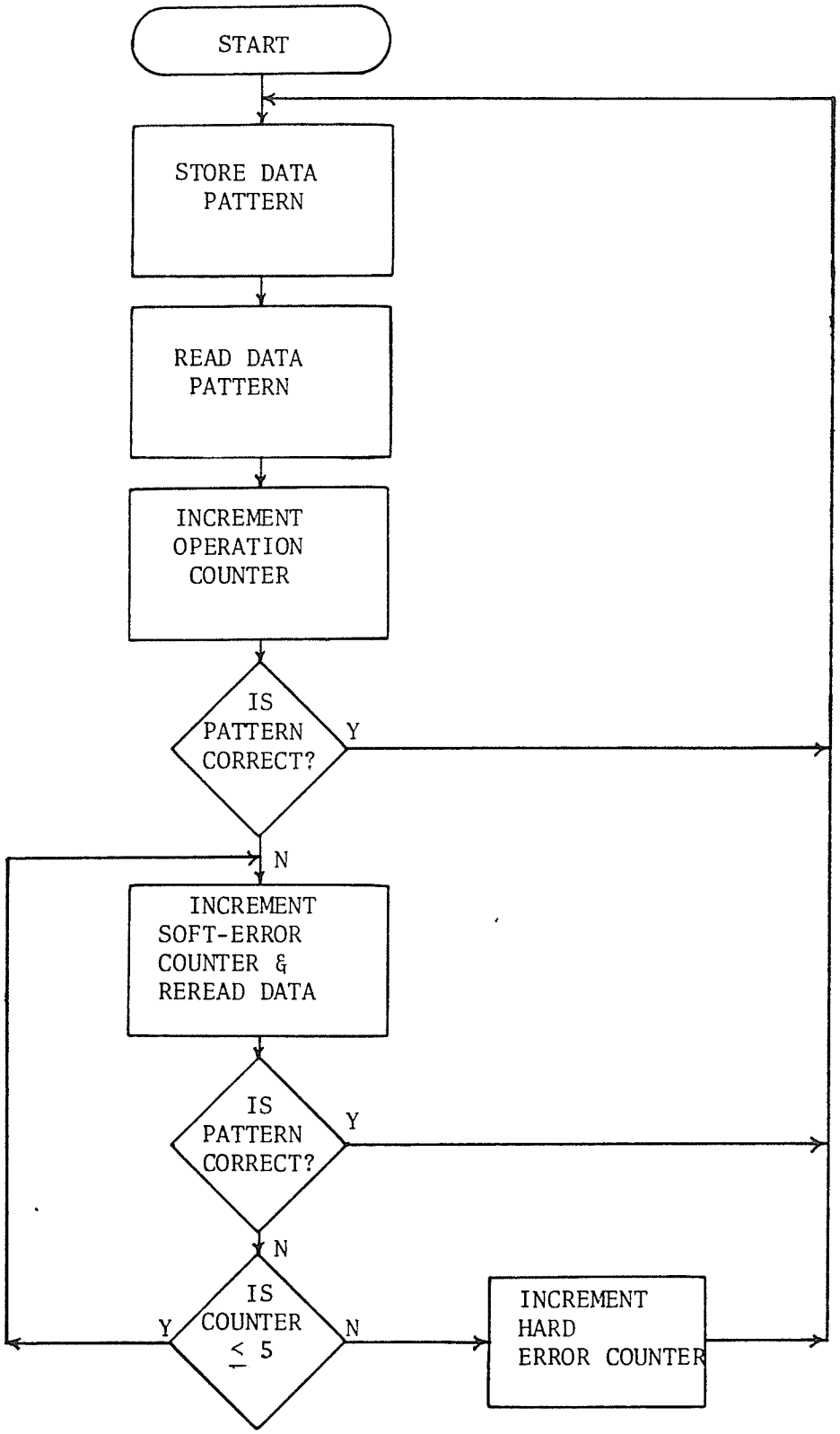


Figure 5.1: Test Software Flowchart

if the previous test was successful, success is determined by a low error rate. One inherent assumption in performing each test is that, if one test is successful and failure occurs during the next test, the failure is in the portion of the device being tested. For example, if an error occurs in test 2, it is assumed that the failure is in the major loop and not in the generator or detector portion.

5.2.3 Test 3:Major Loop Nearest Neighbour Testing

The object of this test is to determine the failure rate in the major loop due to bubble-to-bubble interactions. The bit pattern used in performing this test was BAA82557 Hex.

To fully test nearest neighbour interactions in the major loop, the multi-page mode of operation was used. The multi-page mode maximizes the number of data bits in the major loop, hence providing the largest inter-bubble interaction.

5.2.4 Test 4:Minor Loop Failure Mode Testing

The test pattern and the objectives of test 4 are the same as for the major loop failure mode testing. However, care must be exercised in choosing the data pattern since it must be stored serially in each minor loop. The pattern used is FF08800F77 Hex. During major loop testing, this bit pattern could be broken up into each minor loop. This break up is possible during the read/write process because the bit pattern is assembled and rotated serially in the major loop. To store the data pattern in the minor loops, the failure mode bit pattern consists of 40 pages of ones or zeros, where the contents of each page are determined

by each bit in the test pattern. The test procedure is to write this data pattern and perform continuous reads. This provides a means of rotating the pattern around the minor loop and verifying data integrity.

5.2.5 Test 5:Minor Loop Nearest Neighbour Testing

This test also utilizes the same bit pattern used for major loop nearest neighbour testing. Again the data pattern must be stored such that the serial pattern is stored in each minor loop. The method for storing the data pattern used in test 4 is again employed here. However, in this test, 640_{10} pages of "1s" and "0s" are stored corresponding to a repetitive combination of the bit pattern BAA82557 Hex. This provides the maximum inter-bubble interaction thereby increasing the likelihood of failure.

5.3 GENERAL COMMENTS

In performing the above five tests a distinction was made between a soft error and a hard error. The consistent verification scheme was observed during each test. This involved comparing the read data with the expected data, and if an error occured, the error would be flagged as a soft error. On an error condition the read process was repeated until no error occured or until five soft-errors were encountered, whichever occured first. The occurance of five soft-errors constituted a hard-error, since it was assumed to be incorrectable by performing a reread. If a hard error condition was detected the entire test procedure was repeated. Each test procedure also maintains a software counter for the number of read/write operations performed which in turn gives a direct indication of the number of bits that were read or written per test.

A provision was also incorporated in the test software to provide a means of checking any particular minor loop if continuous errors were observed in that minor loop. This test procedure is invoked by entering a "6" after the prompt is displayed. After entering a "6" the monitor prompts for a 18 byte data pattern which is to be rotated through the entire memory. For example, by entering the following bit pattern 000...010...000 (total length equals 144 bits), the minor loop corresponding to the active high bit (logical one) will be continually exercised.

5.4 OPERATIONAL TESTING

The object of operational testing was to observe the effects of device operation on the control signals. In particular, the test process involves examining changes in the coil driving currents, generate currents, transfer in and transfer out currents, replicate/annihilate currents, and detector output signals. The effects of temperature on the generate current were also observed. The results of this test are presented in the following chapter. Operational testing provided a qualitative evaluation of the bubble memory driving circuits.

A total of six tests were performed to evaluate the reliability and useability of a bubble memory system. The following tests were performed:

a) The effects of the generate element on data bits and the effects of bit patterns on the output signal,

b) Major and minor loop failure mode testing,

c) The effects of inter-bubble interactions in the major and minor loops, and

d) A qualitative evaluation of the bubble memory driving circuits.

In performing bit pattern sensitivity testing, any failures are flagged by means of a printout indicating the actual data pattern, the observed data pattern, and a record of the number of read/ write operations performed until failure occurred. The record provides a means for evaluating the effective error rate, and hence the reliability of the bubble memory system.

Chapter VI

TEST RESULTS

Bubble memory system testing was performed to determine the following: (i) the effects of bit patterns on data integrity, (ii) to observe the effects of prolonged operation on control signals, and (iii) the effects of prolonged operation and bit patterns on the detector output signals. Test methodology was discussed in the previous chapter with the results of the tests presented in this chapter.

6.1 BIT PATTERN RESULTS

Five tests were performed to evaluate the reliability of the magnetic bubble memory system. To evaluate the reliability, a minimum of 10^9 bits per test were read. Table 6.1 summarizes the results of the various tests.

A total of zero soft and hard errors were detected during the course of system testing, which involved cycling over 2.4×10^{10} bits through the bubble memory. In performing the five tests, the next test was always started immediately following the completion of the current test. This ensured that the entire system was always operating in a consistent test environment.

In examining the above results, it can be concluded that under normal operating conditions the bubble memory system is extremely reliable. This reliability can be measured in terms of the observed error rate of

TABLE 6.1

Bit Pattern Sensitivity - Test Results

Test	Total No. of bits/test	No. of soft errors	No. of hard errors
Generator and detector testing	2.0200922×10^9	0	0
Major loop failure mode testing	1.6522512×10^9	0	0
Major loop nearest neighbour testing	9.1993268×10^9	0	0
Minor loop failure mode testing	2.6784197×10^9	0	0
Minor loop nearest neighbour	8.7700794×10^9	0	0

less than 4.11×10^{-11} . This represents a probability of a bit in error of less than 4.11×10^{-11} when the module is exercised using worst case patterns, without any error correction. This error rate compares quite favorably with the error rate of other devices as shown in Fig. 6.1 [65].

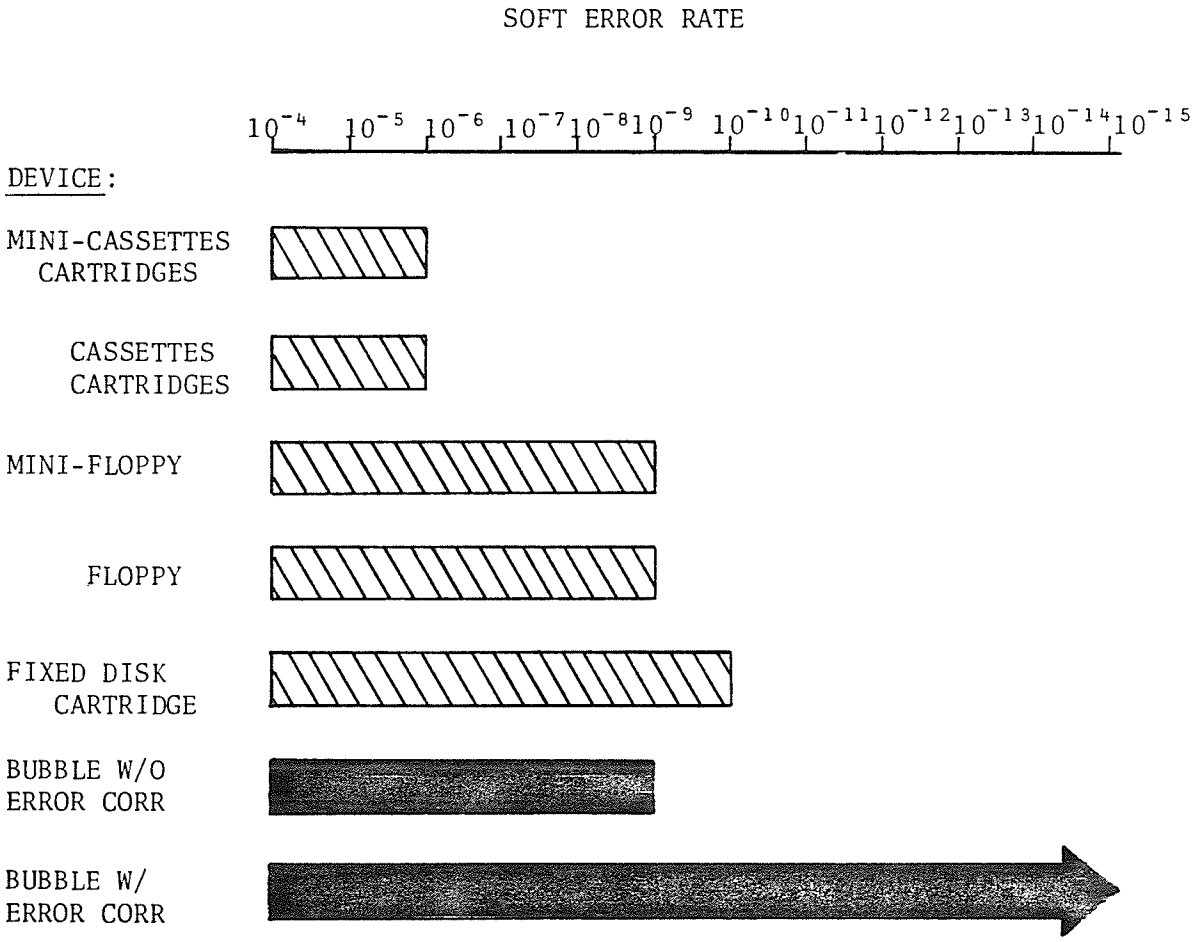


Figure 6.1: Device Error Rate Comparison

6.2 EXPERIMENTAL EQUIPMENT AND PROCEDURES

In performing tests to observe the effects of prolonged operation on control signals and bit patterns on the detector output signals, the following test equipment was used:

1- Tektronix oscilloscope model 465 (100 MHz bandwidth)
1- Tektronix storage oscilloscope model 466 (100 MHz bandwidth)
1- Fluke 85002- model 2100A digital thermometer with a Cu/Con thermocouple.
1- Tektronix active differential probe and amplifier model P6046
1- Tektronix AC current probe model P6019
1- Tektronix AC current probe model P6016

The above equipment was calibrated according to manufacturer specifications. All probes were rechecked prior to use by measuring known currents and temperatures. The differential probe was checked by measuring ground potential with a physical separation of the probe tips kept as close to zero as possible. The observed voltage was within specifications supplied by the manufacturer.

While performing the various tests, certain precautions were taken to maintain consistency of results. One precaution was to run the test equipment until thermal equilibrium was attained, thereby minimizing the effects of the dc thermal drift. Once thermal equilibrium was achieved, the calibration was rechecked and adjusted if required. Another method to ensure consistent results was to rigidly fix the test probes such that no positional effects were introduced into the measurements. Positional effects are more pronounced in the differential probe due to interference induced in the probe leads from external sources. Finally, the thermocouple was affixed to the bubble module to ensure that the thermal time constant did not effect the accuracy of temperature measurements.

6.3 EXPERIMENTAL RESULTS

Two tests were performed to determine the effects of prolonged system operation. One test was for a period of 21 hours and the second test on a different bubble module was for a period of 96 hours. A total of 1.686×10^9 bits were cycled in a period of 96 hours. The second test was conducted because the replicate/annihilate gate of the first module was accidently destroyed rendering the device inoperable. The results of the two tests are illustrated in Figs. 6.2-6.5. Test one was for a period of 21 hours and the results are presented for completeness. From the figures it can be noted that the results of test one disagree with the results of test two. This disagreement can be attributed to the number of data points collected for test one. Measurements were performed every half hour for the first three hours, after which time it was observed that the readings were consistent with each other. Since the readings were consistent the collection rate was reduced to every two hours. Because of the limited number of measurements for test one, a valid analysis cannot be performed on the data. The results of test two are all in agreement with manufacturer's specifications which indicate no degradation in driving capability of the bubble control signals. Figure 6.6 illustrates the shape and amplitude of the various control signals.

A study of the effects of device operation on the bubble output signals was also performed. Photographs of bubble output signals during the course of measurements indicate that device operation had no effect on the bubble output signals. This is illustrated in Fig. 6.7 and 6.8 which represent the bubble output signals observed at the beginning and end of the prolonged test period. Both figures depict a bit pattern correspond-

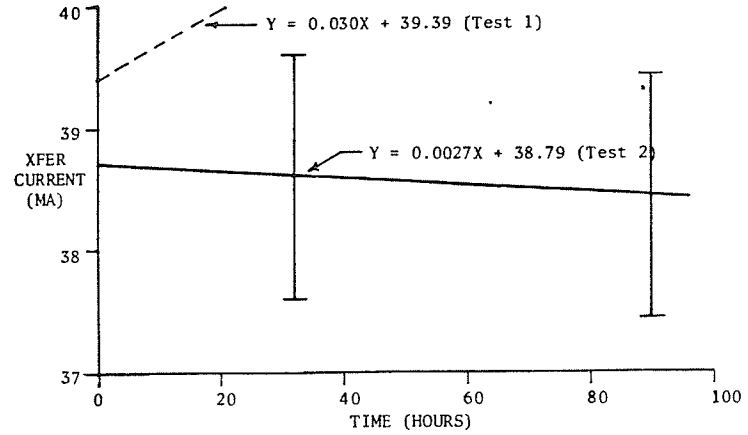


Figure 6.2: Transfer Current vs Time

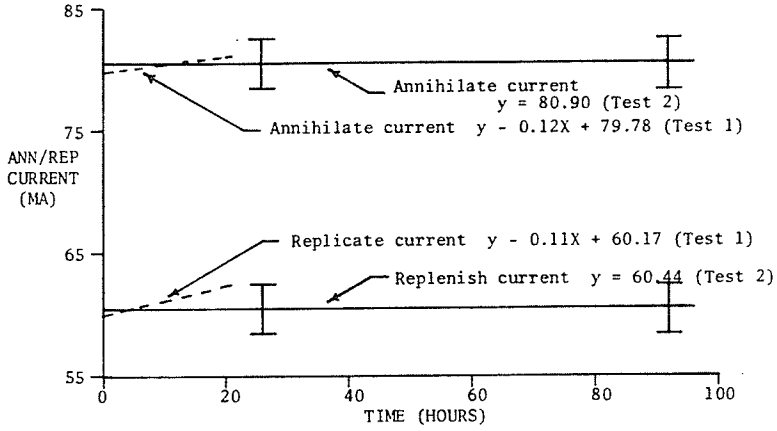


Figure 6.3: Annihilate/Replicate Current vs Time

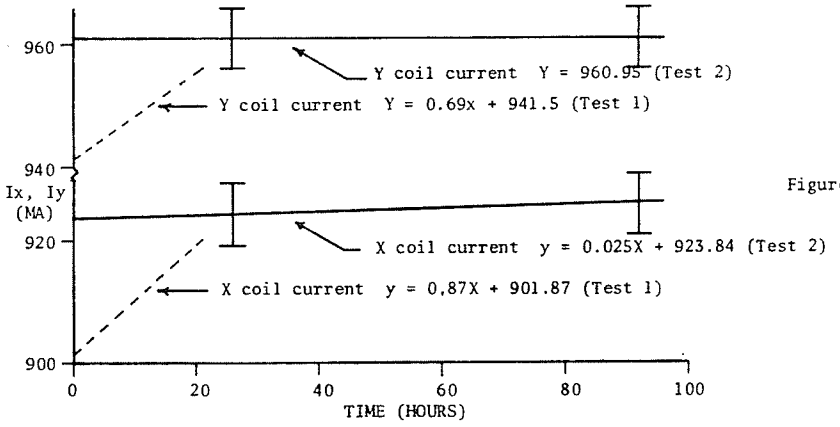


Figure 6.4: Coil Drive Current vs Time

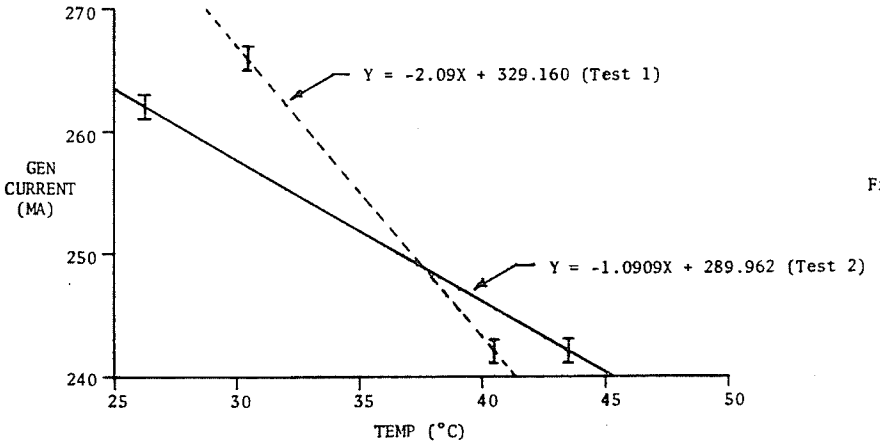
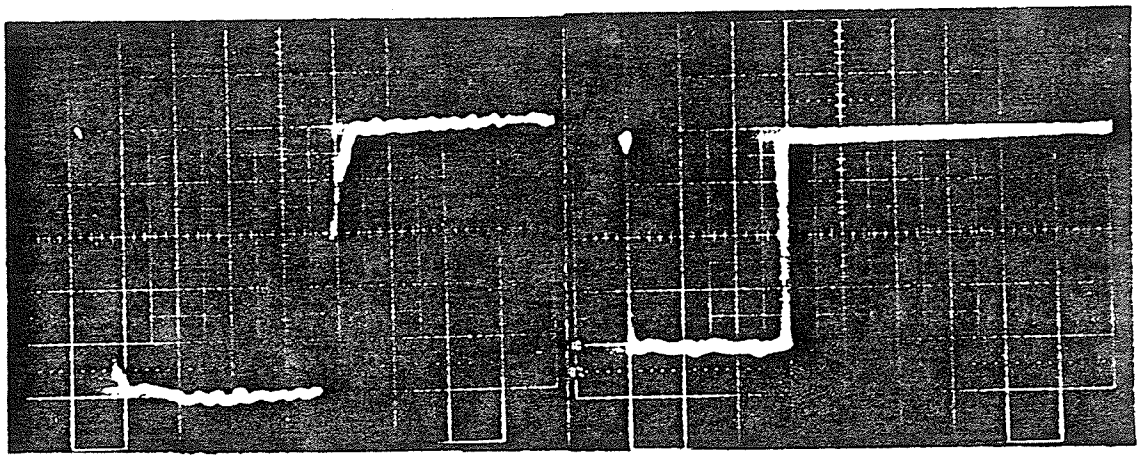
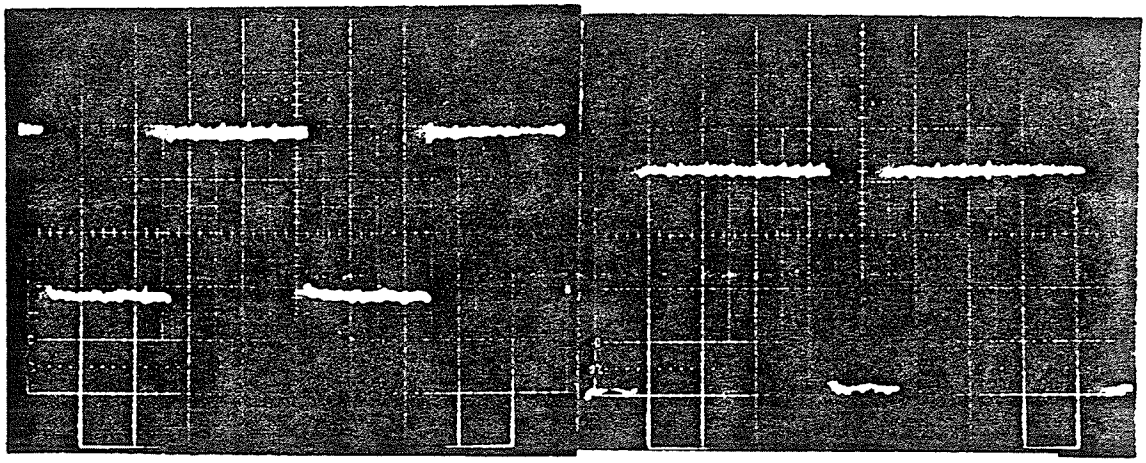


Figure 6.5: Generate Current vs Temperature



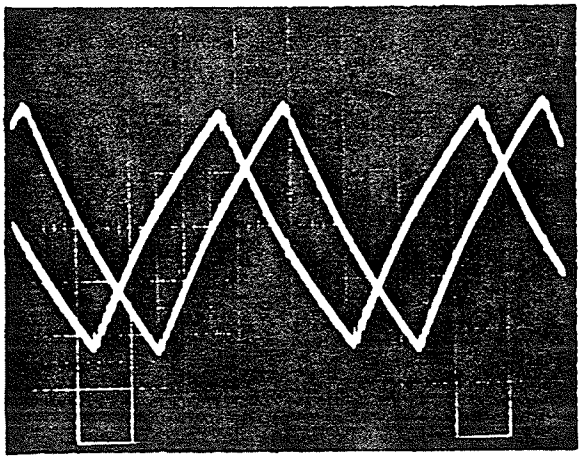
Generate Current
50 mA/DIV Vert.
0.1 μs/DIV Horz.

Transfer Current
10 mA/DIV Vert.
2 μs/DIV Horz.



Replicate Current
20 mA/DIV Vert.
2 μs/DIV Horz.

Annihilate Current
20 mA/DIV Vert.
20 μs/DIV Horz.



Coil Current (Ix lags Iy)
0.2 A/DIV Vert.
2 μs/DIV Horz.

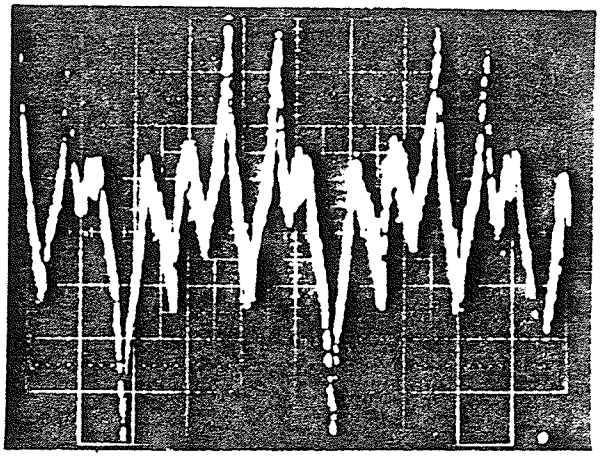
Figure 6.6: Photographs of Control Signals

ing to "11", where the actual bit is represented by the second positive peak. For comparison purposes, a bit pattern corressponding to "00" is illustrated in Fig. 6.9.

6.4 GENERAL OBSERVATIONS

Figures 6.10 and 6.11 illustrate two interesting points. Both figures show the effects of bubble-bubble interaction, as described below. Figure 6.10 shows that the output signal of isolated bubbles is smaller in amplitude than the output signal of a bubble in the vicinity of other bubbles. Figure 6.11 illustrates the effects of neighbouring bubbles on a centrally-located bubble. From Fig. 6.11 the output signal of the centrally-located bubble is marginally larger than the output signal of adjacent bubbles. This can be attributed to the flux contribution from adjacent bubbles.

The TIB-0203 uses dual magneto-resistive detectors in a differential configuration to sense magnetic domains. Domains are propagated on one detector track and a dummy detector is used to increase the signal to noise ratio of the output signal by subtracting common mode signals generated by the rotating in-plane field. One general observation that was made is that the effects of the dummy detector currents are less pronounced on the output signal than that of the actual detector. The current in the dummy detector could be varied as much as 1.5mA before errors were observed in the output signal, whereas the current in the actual detector was varied by as little as 0.5mA before bit errors were detected.



1mv/DIV Vert.
5μs/DIV Horz.

Figure 6.7: Bubble Output Signal at Start of Test

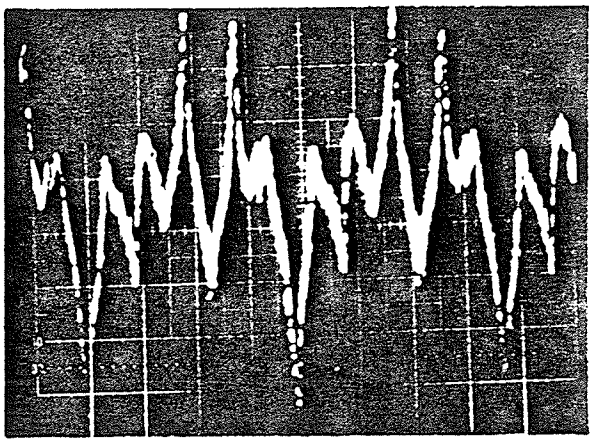


Figure 6.8: Bubble Output Signal at End of Test

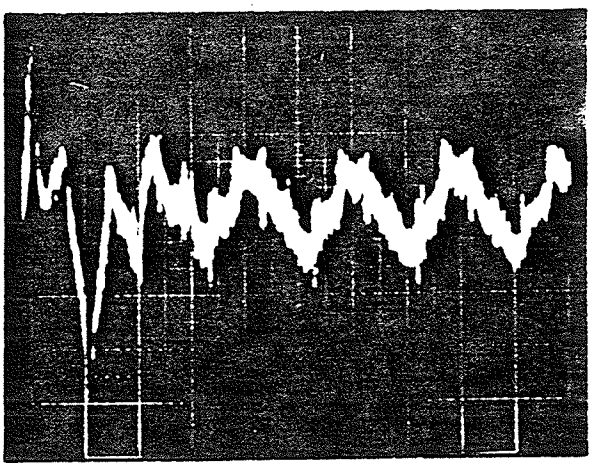


Figure 6.9: Output Signal Representing "00"

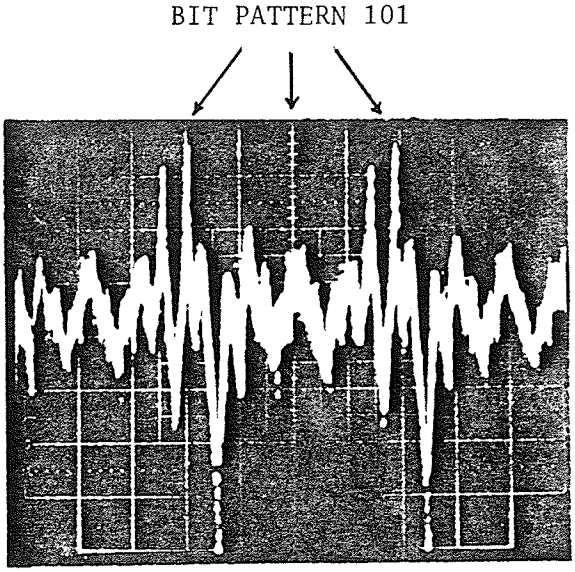


Figure 6.10: Effects of Bubble-Bubble Interaction on Lone Bubble

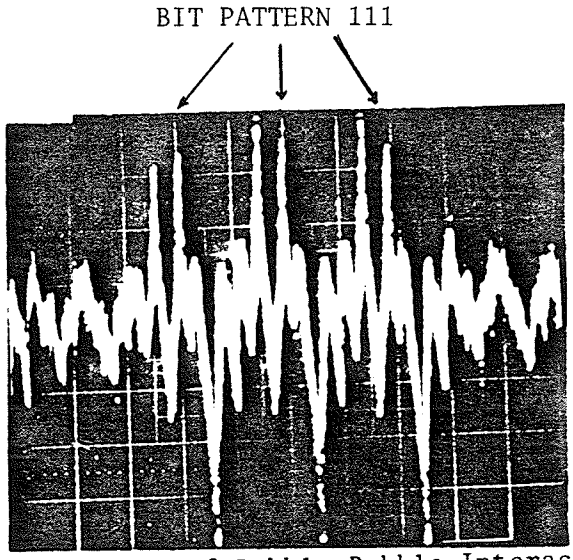


Figure 6.11: Effects of Bubble-Bubble Interaction on Surrounded Bubble

Various tests were performed to evaluate the reliability of the bubble memory system. Bit pattern sensitivity tests were conducted to determine system error rates, i.e., the probability of a bit in error. In performing the tests, no soft or hard errors were observed. In any single test, the worst case error rate was 0.6×10^{-11} and an overall error rate of 4.11×10^{-11} was observed. Both these error rates are less than the error rate of 10^{-9} specified by the manufacturer for device operation without any error correction. Tests were also conducted to observe the effects of prolonged operation on control signals and the effects of bit patterns on the detector output signals. The measurements were conducted over a period of 96 hours. Collected data indicates that there are no observable effects due to prolonged device operation. From the results it can be concluded that the implemented bubble memory system and bubble module are both highly reliable. It was also observed that bubble-bubble interaction has some effects on the strength of the output signal.

Chapter VII

CONCLUSIONS

Magnetic bubbles are small cylindrical magnetic domains in single crystal ferrimagnetic thin films. The domains are stabilized by an external bias field applied perpendicular to the film surface. The stability of the domain is obtained through a balance of the following three forces:

a) The force due to the external bias field,

b) The force due to the magnetostatic energy, and

c) The force due to the domain wall energy.

Bubbles can be used to represent information through binary coding, where the absence or presence of a domain represents a logical "0" or logical "1" respectively. Various memory devices have been manufactured and postulated. Present commercial devices employ permalloy propagation structures to produce field gradients required to manipulate the cylindrical domains. The next generation devices will utilize ion-implanted contiguous disk propagating structures. Future generation devices could use bubble-bubble interaction as a means of propagation as in the bubble lattice configuration or depart from the standard techniques and employ current carrying sheets to produce the field gradients required for bubble propagation. Both techniques promise increased storage densities by at least an order of magnitude from the present limit of 1M bits offered by permalloy devices. Experimental contiguous disk devices with storage

capacities of 11.5Mb have been developed. Aside from an increase in storage densities, current sheet memories offer the possibility of operating devices at speeds of up to 20 MHz, which by far exceeds the present limit of 400 kHz. Because bubble devices are in the initial phases of development little work has been done to study the reliability of the device in actual operating systems.

To study reliability of magnetic bubble memories, a bubble memory system was implemented. The system employed the second generation TIB-0203 magnetic bubble memories in a microprocessor based system. A MC6802 was used to control the memory devices with the aid of a bubble memory controller. The bubble memory system was comprised of the following components: (i) a central processor, (ii) a bubble memory controller, (iii) one bubble module with provision for eight modules, and (iv) and an asynchronous interface. Intelligent control was obtained by interacting with the system through a terminal and a 4K byte system monitor. Other features of the system include a 10K byte buffer, and provision for direct memory access (a feature not directly implementable in a 6802 based system). These features would facilitate the implementation of buffered synchronous communication with burst data rates exceeding the 50 kHz data rate of the bubble memory.

Reliability tests were conducted on the above system. Two types of tests were performed, one to determine the sensitivity of the device to various bit patterns, and the second to observe the effects of prolonged operation on:

a) Device control signals, and

b) Detector output signals.

Five tests were performed to evaluate sensitivity to bit patterns. The tests exercised the different control functions and the entire propagating and storage area. Tests were designed to test the following three failure modes.

a) Failure due to collapse,

b) Failure due to strip-out, and

c) Failure to cross a gap region.

Aside from failure mode testing, tests were performed to determine the effects of bubble-bubble interaction. From the various tests that were performed there were no observed soft or hard errors. With a total number of bits cycled through the memory exceeding 2.4×10^{10} bits, the probability of a bit in error is less than 4.11×10^{-11}. This error rate is less than the error rate of 10^{-9} specified by the manufacturer.

The test results indicate that the magnetic bubble memory system that was implemented is highly reliable, when operating under normal conditions, even without error correction. The results are indicative of the reliability of bubble memories which can be used for high density nonvolatile storage. Bubble memories offer solid state storage for applications requiring low power consumption, portability, or with size and weight constraints, and/or applications in which electromechanical storage is adversely affected by the physical environment. The advantages and characteristics of bubble memories outlined here have been the primary reasons for the existence of this technology. Applications of MBM range from numerical controllers, portable computers, military applications to voice announcement systems, illustrating the versatility of this device.

Chapter VIII

RECOMMENDATIONS

During the design phase of the system, considerations were made to allow for further system modifications. The considerations include a small buffer RAM and a DMA facility for the MC6802. It is recommended that the following enhancements to the system be made to increase the transfer rate from the bubble module to an external host system:

a) In place of the internal 20 byte FIFO in the bubble memory contoller, provide eight external 8 bit parallel in/parallel out shift registers. These registers can be memory mapped to eight consecutive locations and can be used to simulate the internal FIFO in the multipage mode of operation. Since all control signals (such as the defective loop map and clock signals synchronized with the rotating field), are externally available, the shift registers can be used as a means of operating bubble modules in parallel. To initiate action from the controller, dummy data can be written and read from the controller. This data will not be stored in the bubble module since the internal FIFO would not be connected to any external circuitry. This implementation would increase the data rate from the bubble module from 50 kHz to 400 kHz.

b) Incorporate the synchronous communication channel and the direct memory access capability. The advantages and use of such an implementation have been discussed in chapter four.

For further research it is recommended that a study of system operation under different environmental conditions be performed. This includes studying the effects of prolonged operation under different thermal conditions and studying the effects of electromagnetic interference.

It is also recommended that the study of different memory organizations and propagation structures be carried out. A study of these areas would inevitably lead to future bubble memories with higher storage densities and faster access times. One method foreseeable for improving access times would be to implement an architecture that does not rely on manipulating the entire storage array. The current-sheet memory technology is most promising in this regard, since it does not require a coil assembly to provide the rotating in-plane field.

REFERENCES

1. D. Toombs, "An update: CCD and bubble memories ." IEEE Spectrum, April 1978, pp 22-30.

2. D. Toombs, "CCD and bubble memories: System implications. " IEEE Spectrum May 1978, pp 36-39.

3. A. J. Perneski, "Propagation of cylindrical magnetic domains in orthoferrites." IEEE Transactions on Magnetics vol. MAG-5, No. 3, Sept. 1969, pp 554-557.

4. A. H. Bobeck, et al, " Uniaxial magnetic garnets for domain wall bubble devices." Applied Physics Letters vol 17, No. 3, Aug. 1,1970

5. J. A. Pistorius, J.M. Robertson, and W.T. Stacy, "The perfection of garnet bubble materials. " Philips Technical Review vol 35, No. 1, 1975, pp 1-10.

6. P. Chaudhari, J. J. Cuomo, and R. J. Gambino, "Amorphous metallic films for bubble domain applications." IBM Journal of Research and Development Jan. 1973, pp 66-68.

7. M. H. Kryder, Lung-Jo Tao, and C. H. Wilts, "Dynamics of amorphous film bubble devices." IEEE Transactions on Magnetics vol MAG-13, No. 5, Sept. 1977 , pp 1626-1631.

8. H. Chang, Magnetic Bubble Technology:Integrated Circuit Magnetics for Digital Storage and Processing IEEE Press, 1975.

9. A. H. Bobeck and H. E. D. Scovil, "Magnetic bubbles," Scientific American vol 224, No. 6, June 1971, pp 78-90.

10. T. C. Chen and H. Chang, "Magnetic bubble memory and logic," Advances in Computers vol 17, 1978, pp 223-282.

11. C. Kooy and U. Enz , "Experimental and theoretical study of the domain configuration in thin layers of $BaFe_{12}O_{19}$.", Philips Research Reports 15, 1960, pp 7-29.

12. A. H. Bobeck, "properties and device applications of magnetic domains in orthoferrites. ", Bell System Technical Journal vol 46, No. 8, October 1967, pp 1901-1925.

13. A. A. Thiele, "The theory of cylindrical magnetic domains.", The Bell System Technical Journal vol 48, December 1969, pp 3287-3335.

14. A. A. Thiele, "Device implications of the theory of cylindrical magnetic domains.", Bell System Technical Journal vol 50, March 1971, pp 727-775.

15. W. F. Druyvesteyn, F.A. Kuijpers, A.G.H. Verhulst and C.H.M. Witmer, "Single-mask bubble memory with rotating field control.", Philips Technical Review vol 36, No. 6, 1976, pp 146-159.

16. W. Kinsner and E. Della Torre, "Magnetic field distributions due to permalloy bubble propagation circuits.", Journal of Applied Physics vol. 46, No. 2, February 1975, pp 933-935.

17. T. H. O'Dell, Magnetic Bubbles North-Holland Publishing Co., Amsterdam, 1975.

18. A. H. Bobeck and E. Della Torre, Magnetic Bubbles North-Holland Publishing Co., Amsterdam, 1975.

19. E. Della Torre and W. Kinsner, "The parallel bar bubble propagating circuit.", IEEE Transactions on Magnetics vol. MAG-9, No. 3, Sept. 1973, pp 298-303.

20. E. Della Torre and W. Kinsner, "Develepments in magnetic bubble memories.", Canadian Electrical Engineering Journal vol. 3, No. 3, 1977, pp 3-9.

21. J. A. Copeland, J. P. Elward, W. A. Johnson, and J. G. Ruch, "Single conductor magnetic-bubble propagation circuits.", Journal of Applied Physics vol. 42, No. 4, 15 March 1971, pp 1266-1267.

22. E. H. L. J. Dekker, K. L. L. Van Mierloo, and R. de Werdt, "Combination of field and current access magnetic bubble circuits.", IEEE Transactions on Magnetics vol. MAG-13, No. 5, Sept. 1977, pp 1261-1263.

23. Y.S. Lin, G.S. Almasi, and G.E. Keefe, "Contiguous-disk bubble domian devices.", IEEE Transactions on Magnetics vol. MAG-13. No. 6, Nov. 1977, pp 1744-1764.

24. A. H. Bobeck, S. L. Blank, A. D Butherus, F.J. Ciak, and W. Struass, "Current access magnetic bubble circuits.", The Bell System Technical Journal vol. 58, No. 6, July-August 1979, pp 1453-1540.

25. K.Y. Ahn, et al, "Fabrication of contiguous-disk magnetic bubble devices." IEEE Transactions on Magnetics vol. MAG-16, No. 1, Jan. 1980, pp 93-98.

26. G. S. Almasi, et al, " Bubble domain propagation mechanisms in ion-implanted structures.", AIP Conf. Proc. vol 24, 1974, pp 630-632.

27. A. Hubert, "Charged walls in the magnetic films.", IEEE Transactions on Magnetics vol. MAG-15, No. 5, Sept. 1979, pp 1251-1260.

28. M. J. Freiser, "On the zigzag form of charged domain walls.", IBM Journal of Research and Development vol. 23, No. 3, May 1979, pp 330-338.

29. T. J. Nelson, et al, "Design of bubble device elements employing ion- implanted propagation patterns.", The Bell System Technical Journal, vol. 59, No. 2, Feb. 1980, pp 229-257.

30. T. J. Nelson, et al, "Ion-implanted bubble circuit function design.", IEEE Transactions on Magnetics, vol. MAG-17, No. 1, Jan 1981, pp 1134- 1141.

31. W. Kinsner, E. Della Torre, and R. Hutton, "Bubble cutting circuits.", IEEE Transactions on Magnetics vol. MAG-10, No.4, December 1974, pp 1071-1079.

32. W. Kinsner and E. Della Torre, "Modelling and optimization of magnetic bubble generators. ", Automatic Control Theory and Applications vol. 5, No. 1, January 1977, pp 1-10.

33. H. Dotsch, "The microwave generation and manipulation of magnetic domains.", Philips Technical Review vol. 37, No. 23, 1977, pp 38-41.

34. W. Strauss, "Detection of cylindrical magnetic domains.", Journal of Applied Physics vol 42, No. 4, 15 March 1971, pp 1251-1257.

35. W. Ishak, W. Kinsner, and E. Della Torre, "Magnetostrictive-piezoelectric bubble detectors.", AIP Proceedings 20th Conference Magentism and Magnetic Materials, San Fransisco, Dec. 1974.

36. R. S. Tebble, Magnetic Domains Methuen and Co. Ltd., London 1969.

37. P.J. Grundy, "Magnetic bubbles and their observation in the electron microscope.", Contemporary Physics vol. 18, No. 1, 1977, pp 47-72.

38. S. Yoshizawa, et al, "Small bubble domain detection by Hall effects of InSb films.", IEEE Transactions on Magnetics Sept. 1972, pp 454-457.

39. W. Kinsner and E. Della Torre, " Acoustic bubble detector. ", presented at the INTERMAG Conf., #25-12, London, Apr. 1975.

40. Fujitsu Bubble Memory Cassette, product information, Fujitsu America Inc.,

41. G. Neero, "Serial or major/minor loop? Choose the bubble memory to match the application.", Electronic Design 24, 22 November 1979, pp 172-175.

42. G. Cox and W. Manezuk, "All about the TIB0303 bubble memory chip-loops timing requirement, support.", Electronic Design 10, 10 May 1979, pp 56-60.

43. W.C. Mavity, " The RBM 256- a block replicate bubble memory of a different color.", Electronic Design 10, 10 May 1979, pp 66-70.

44. B. Bryson, D. Clover and D. Lee, " Megabit bubble memory chip gets support from LSI family.", Electronics April 26, 1979, pp 105-111.

45. W. Kinsner ,et al.,United States Patent No. 4,059,829, issued Nov. 22, 1977, "Multi State Magnetic Bubble Domain Cell for Random Access Memories."

46. J.E. Juliussen, "Magnetic bubble memory interfacing." 11th Compcon. Cat. 75CH0988-6C, 1975, pp 87-90.

47. P. K. George and G. Reyling Jr., " Bubble memories come to the boil." Electronics August 2, 1979, pp 99-108.

48. S. Konishi, Ta-Lin Hsu, and B. R. Brown, " Discrimination of s=1 and s=0 bubbles by dynamic bubble collapse.", IEEE Transactions on Magnetics vol. MAG-15, No.1, Jan 1975, pp 885-890.

49. B. A. Calhoun, J. S. Eggenberger, L. L. Rosier and L. F. Shaw, "Column access of a bubble lattice:Column translation and lattice translation.", IBM Journal of Research and Development July 1976, pp 368-375.

50. C.P. Ho and Hsu Chang, " Field access bubble-lattice propagation devices." IEEE Transactions on Magnetics vol. MAG-13, No. 6, Nov. 1977, pp 1744-1764.

51. J. G. Posa, "Bell Laboratories show off chip with a capacity of 11.5 megabits.", Electronics August 2, 1979, pp 44-46.

52. H. Chang, Magnetic Bubble Memory Technology Marcel Dekker Inc., New York, 1978.

53. A.H. Bobeck, P.I. Bonyhard, and J.E. Geusic, "Magnetic bubbles- An emerging new memory technology.", Proceedings of the IEEE vol. 63, No. 8, August 1975, pp 1176-1195.

54. B. S. Scott," The bubble memory data terminal and its role in distributed processing.", IEEE International Sysmposium on Circuits and Systems 1980 Conf. Proc., vol.3, cat. no.C41564-4/80/0000-0738, pp 738-740.

55. W. Kinsner and B. Joll, "A Stand Alone Magnetic Bubble Memory",Presented at the International Microelectronics Conference, Tokyo, May 28-30, 1980

56. Texas Instruments, Magnetic Bubble Memory: System Application Manual, 1979.

57. Texas Instruments, User's Manual:TM990/210 Series Bubble Memory System., June 1980.

58. Motorola, The Complete Microcomputer Data Library, 1978.

59. Intel, Intel Component Data Catalog. 1979.

60. Texas Instruments, Magnetic Bubble Memory and Associated Circuits, 1978.

61. F. B. Hagedorn, et al, "Magnetic bubble device testing.", IEEE Transactions on Magnetics vol. MAG-13, No. 5, Sept. 1977, pp 1364-1369.

62. P. H. Burlison, "Bubble memories are high-technology devices that demand high-level testing.", Electronic Design 10, May 10, 1979, pp 74-78.

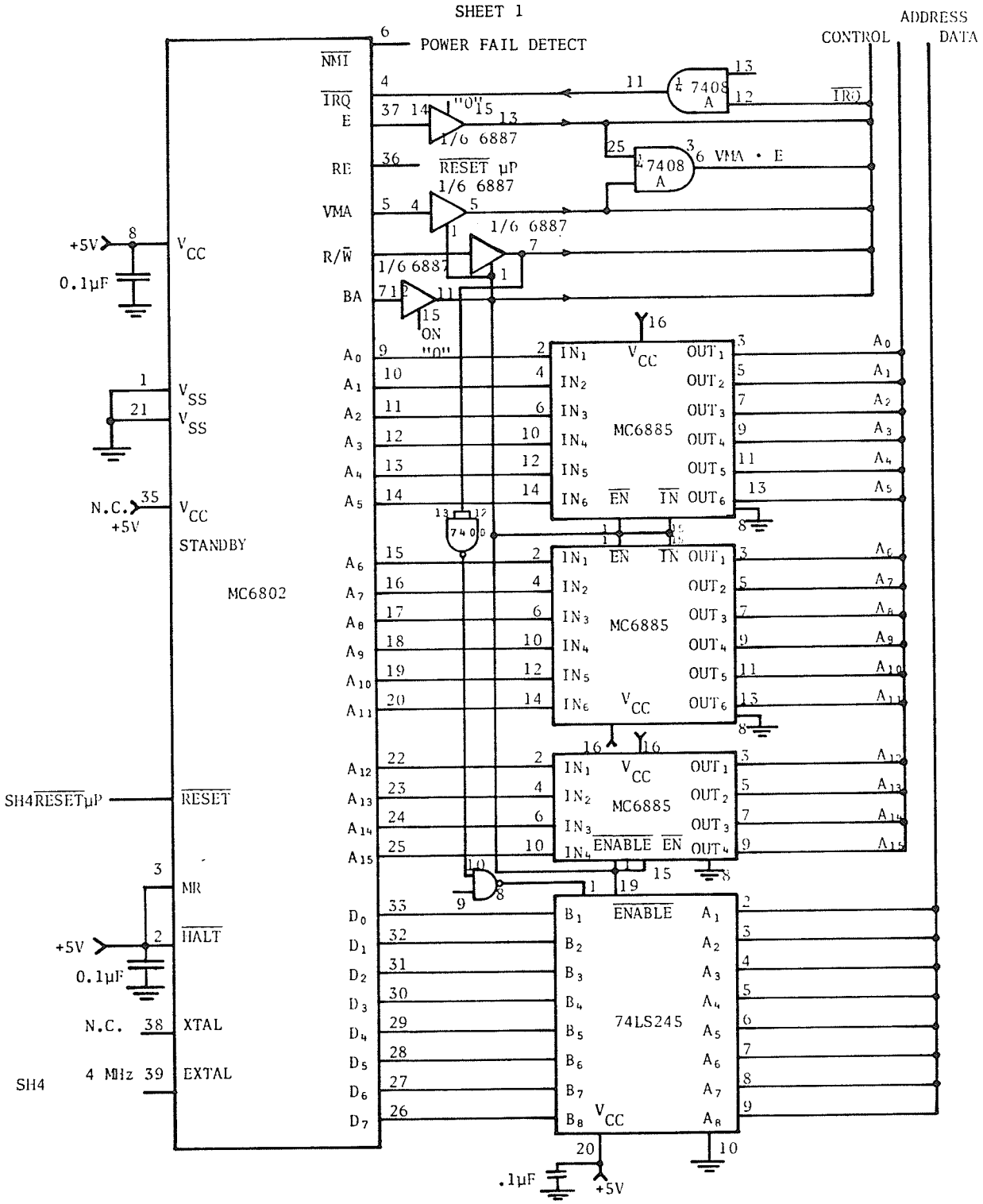
63. S. Bisset, S. Bristow, and Tom T. Chen, "Bubble memories demand unique test methods.", Electronics May 10, 1979, pp 117-122.

64. D. C. Cheng and J. E. Davis, "Characterization and testing of magnetic bubble memory.", 1979 IEEE Test Conference Proc. Session 3, Cat. Ch1509-9/79/0000-0053, pp 53-64.

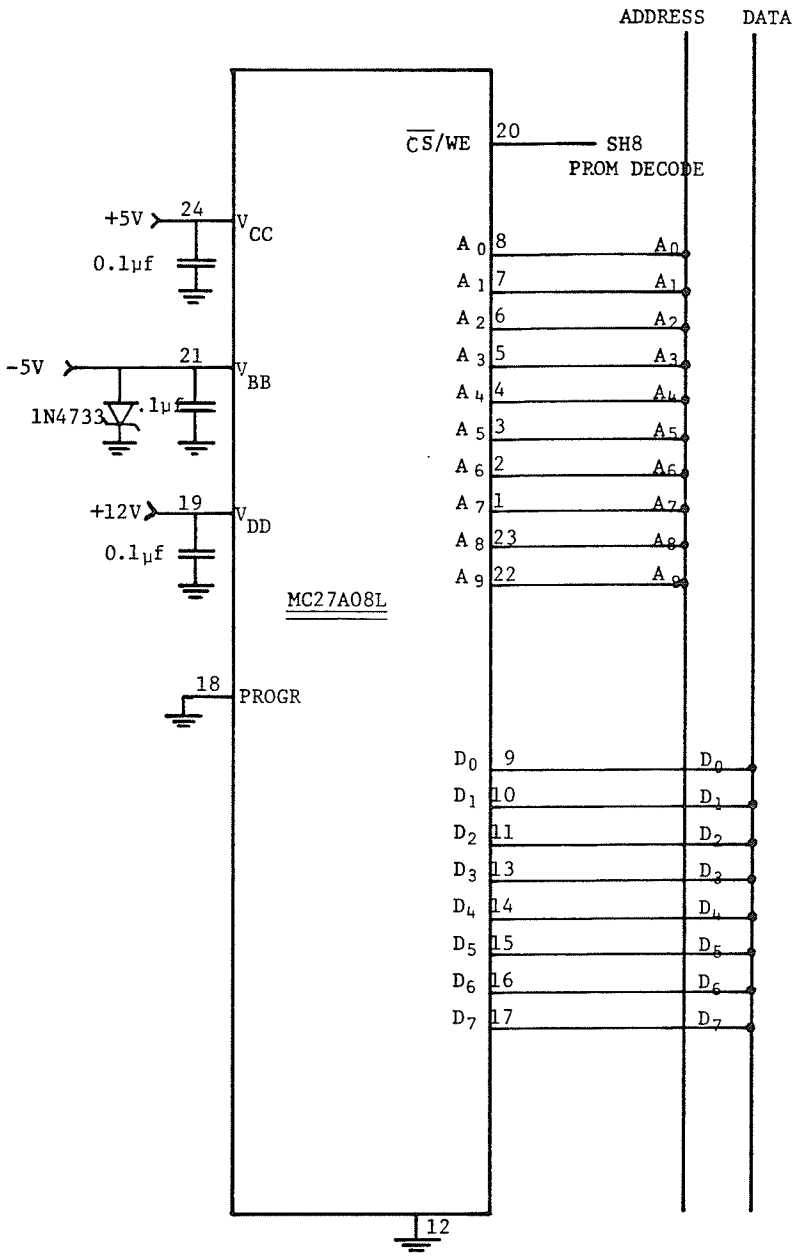
65. G. Cox, UCLA Extension Course, Bubble Domain Technology, April 14-18, 1980

Appendix A

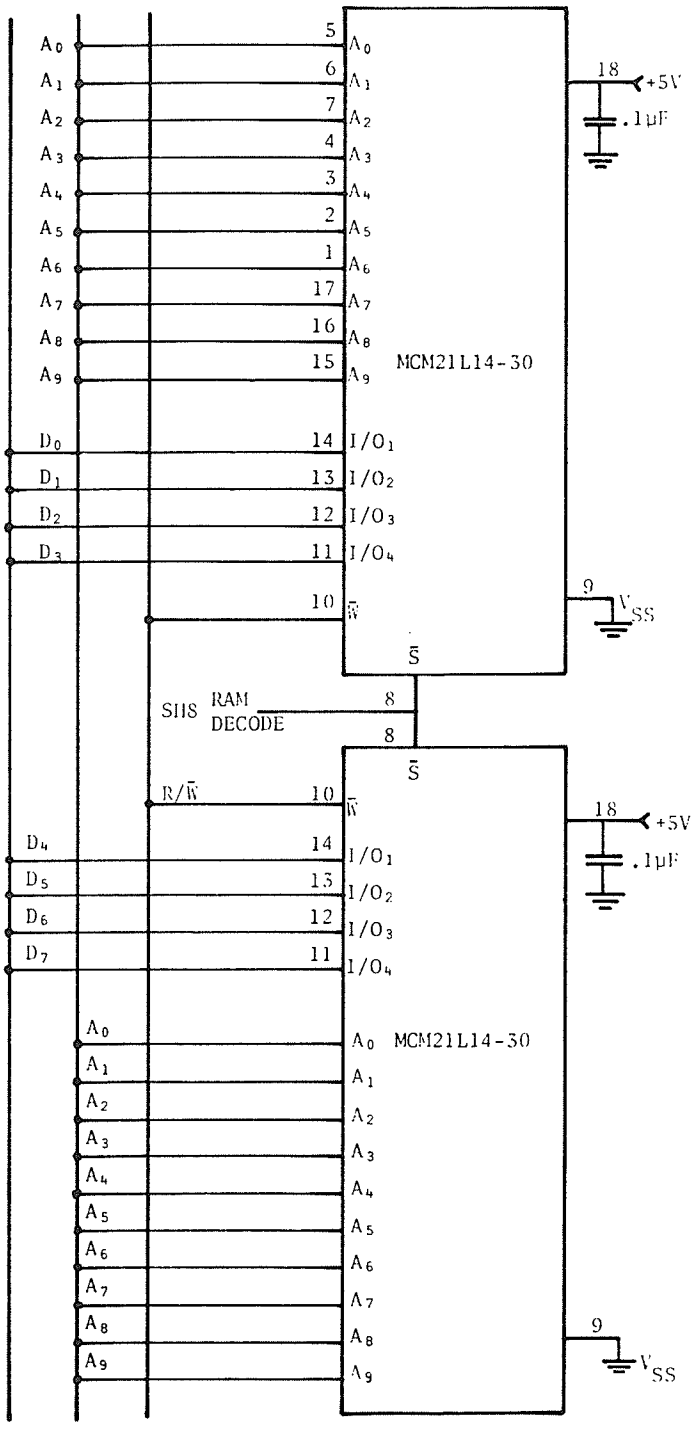
HARDWARE SCHEMATICS

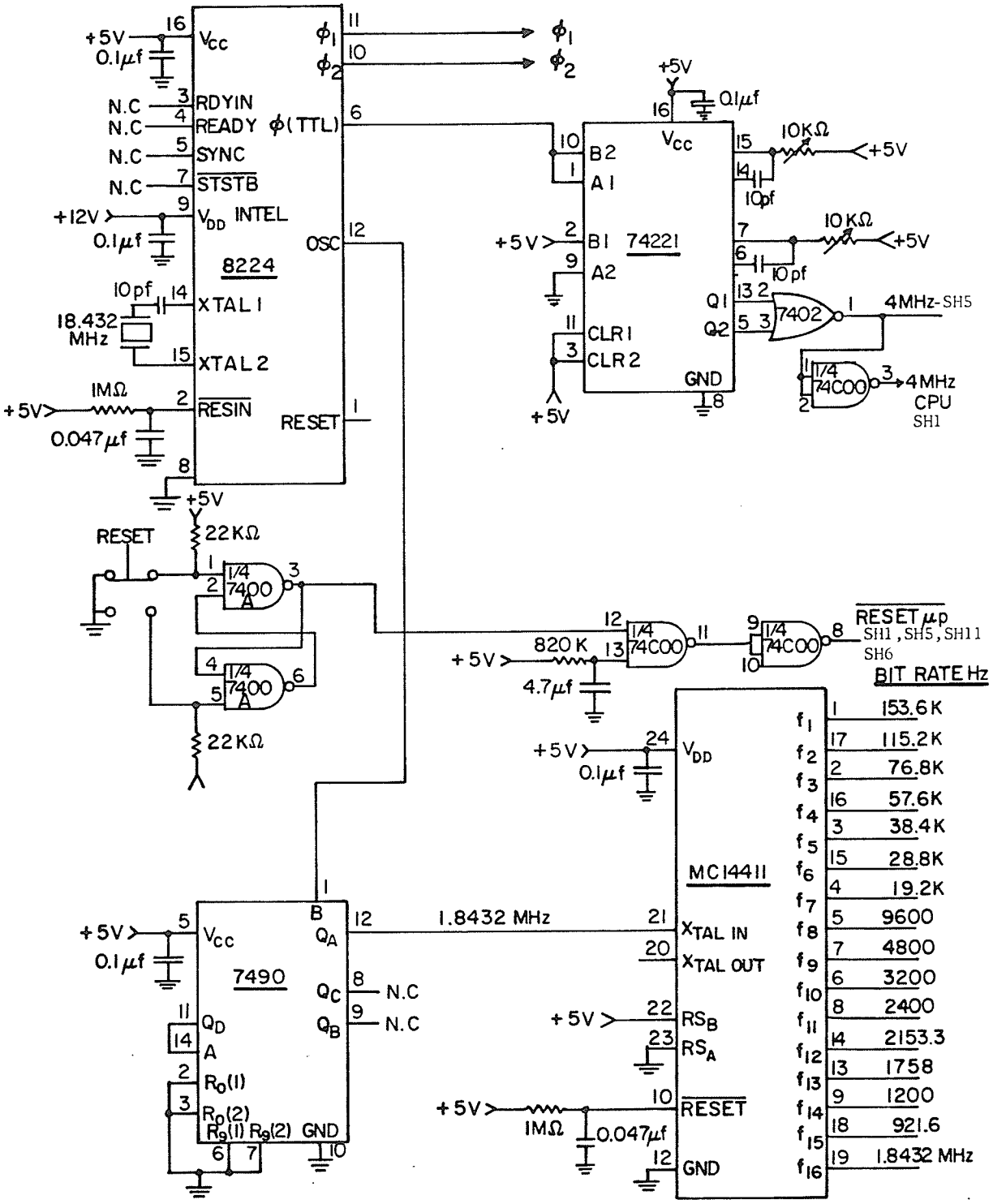

SHEET 1
ADDRESS
CONTROL
DATA
POWER FAIL DETECT
NMI
IRQ
E
RE
VMA
R/W
BA
RESET µP
1/6 6887
7408 A
VMA • E
+5V
0.1µF
VCC
VSS
N.C.
STANDBY
MC6802
MC6885
74LS245
ENABLE
SH4RESETµP
RESET
MR
HALT
XTAL
EXTAL
4 MHz
SH4
.1µF

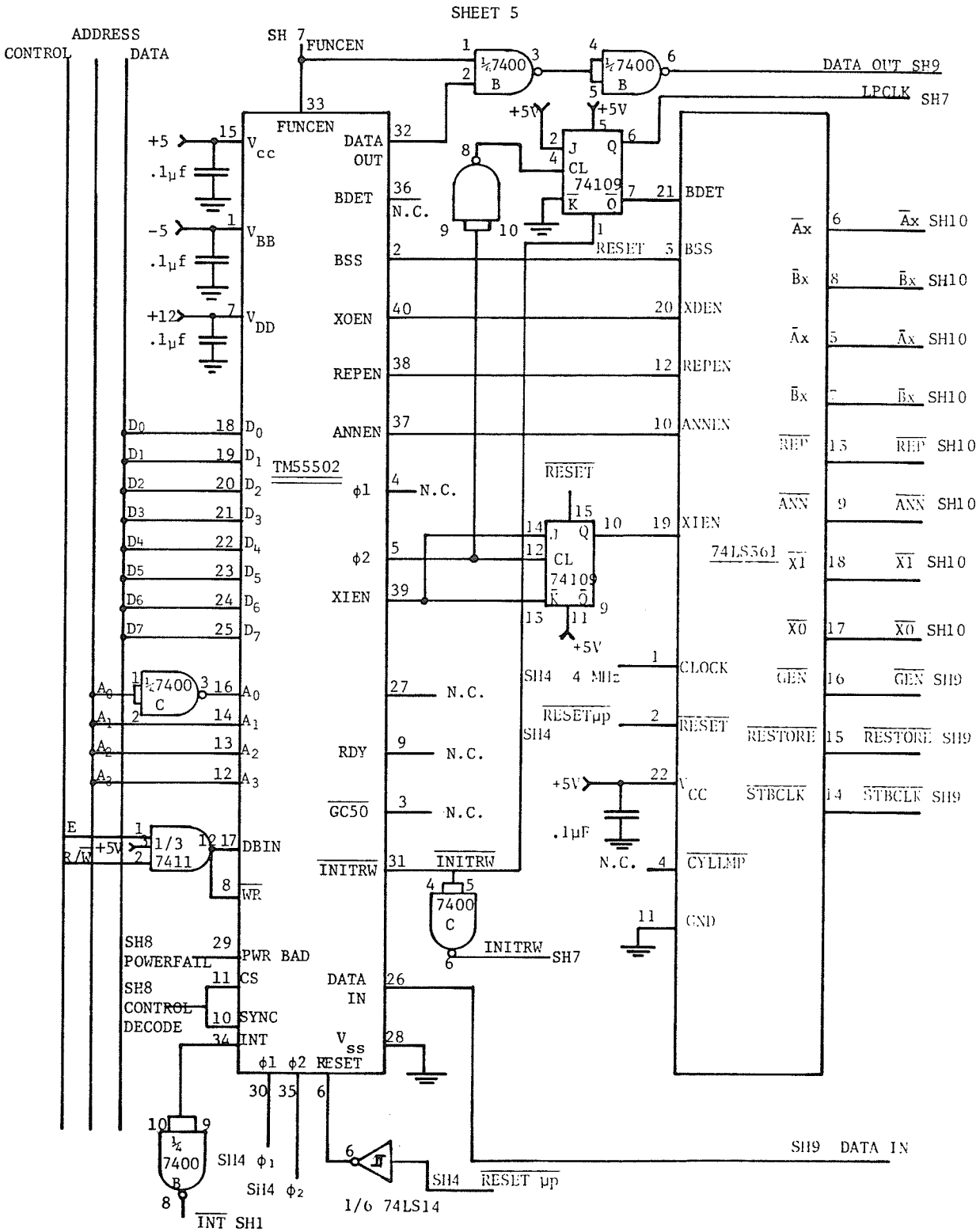
SHEET 2


ADDRESS
DATA
CS/WE
20
SH8
PROM DECODE
+5V
24
VCC
0.1µf
-5V
21
VBB
1N4733
.1µf
+12V
19
VDD
0.1µf
MC27A08L
18
PROGR
12
A0 8
A1 7
A2 6
A3 5
A4 4
A5 3
A6 2
A7 1
A8 23
A9 22
D0 9
D1 10
D2 11
D3 13
D4 14
D5 15
D6 16
D7 17

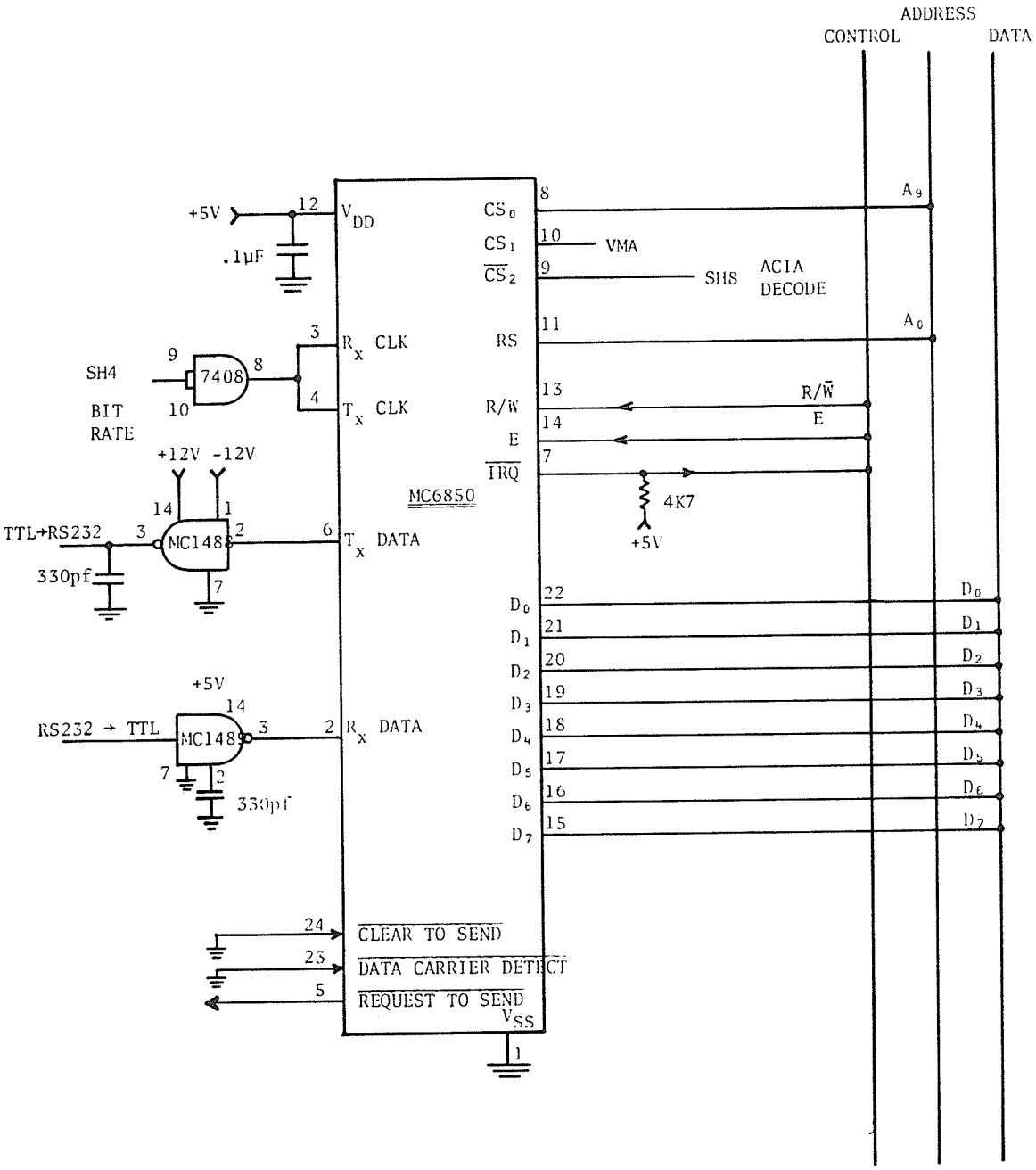
SHEET 3

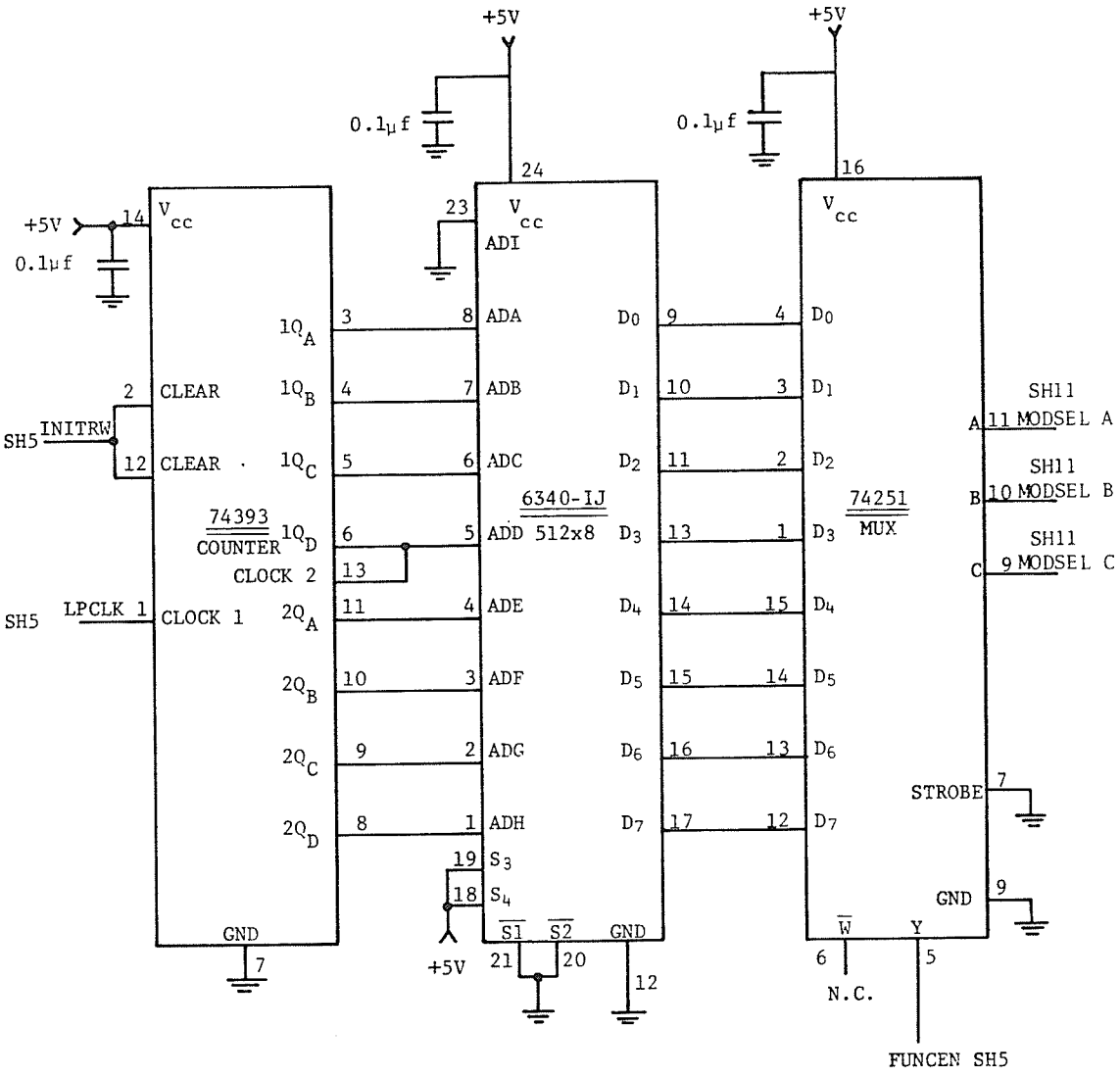

A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
5
6
7
4
3
2
1
17
16
15
MCM21L14-30
18
+5V
.1μF
D0
D1
D2
D3
14
13
12
11
I/O1
I/O2
I/O3
I/O4
10
R/W
9
VSS
S
SHS
RAM
DECODE
8
D4
D5
D6
D7
MCM21L14-30


8224
INTEL
Vcc
VDD
RDYIN
READY
SYNC
STSTB
XTAL 1
XTAL 2
RESIN
RESET
OSC
φ(TTL)
18.432 MHz
1MΩ
0.047μf
0.1μf
10pf
+5V
+12V
N.C
74221
B2
A1
B1
A2
CLR1
CLR2
Q1
Q2
GND
10KΩ
7402
4MHz-SH5
74C00
4MHz CPU SH1
22KΩ
7400
820K
4.7μf
RESET μp
SH1, SH5, SH11
SH6
BIT RATE Hz
MC14411
XTAL IN
XTAL OUT
RSB
RSA
7490
QA
QB
QC
QD
R0(1)
R0(2)
R9(1)
R9(2)
1.8432 MHz
N.C
153.6K
115.2K
76.8K
57.6K
38.4K
28.8K
19.2K
9600
4800
3200
2400
2153.3
1758
1200
921.6
1.8432 MHz

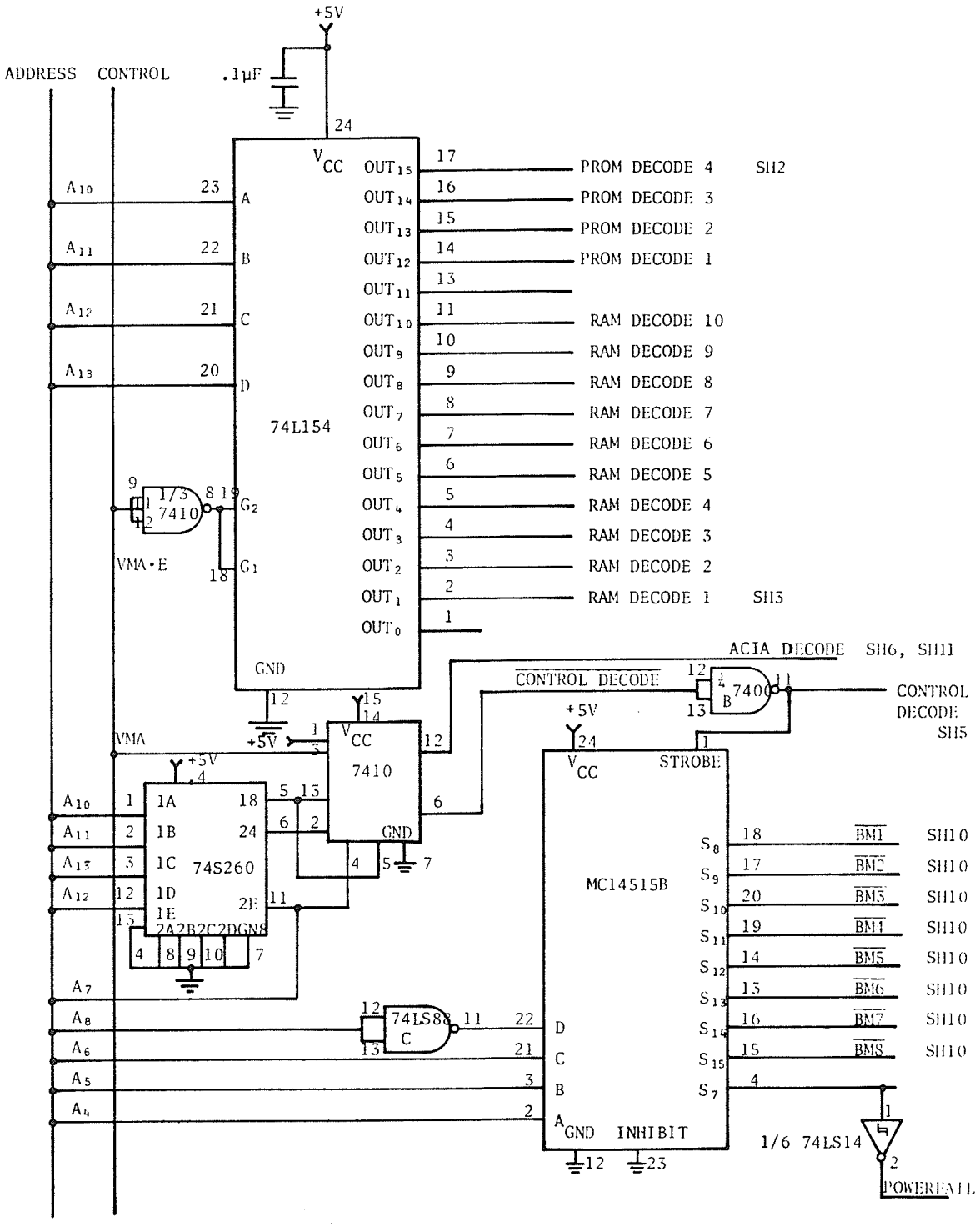

SHEET 5
ADDRESS
CONTROL
DATA
SH 7
FUNCEN
DATA OUT SH9
LPCLK SH7
TM55502
74LS561
+5
-5
+12
.1μf
N.C.
RESET
INITRW
CLOCK
4 MHz
SH4
RESETμp
CYLLMP
STBCLK
RESTORE
DBIN
WR
PWR BAD
POWERFAIL
SH8
CONTROL
DECODE
SYNC
INT
DATA IN
SH9 DATA IN
SH4 RESET μp
SH4 φ1
SH4 φ2
INT SH1
1/6 74LS14
INITRW SH7
GEN SH9
XIEN
XDEN
REPEN
ANNEN
BDET
BSS
CND

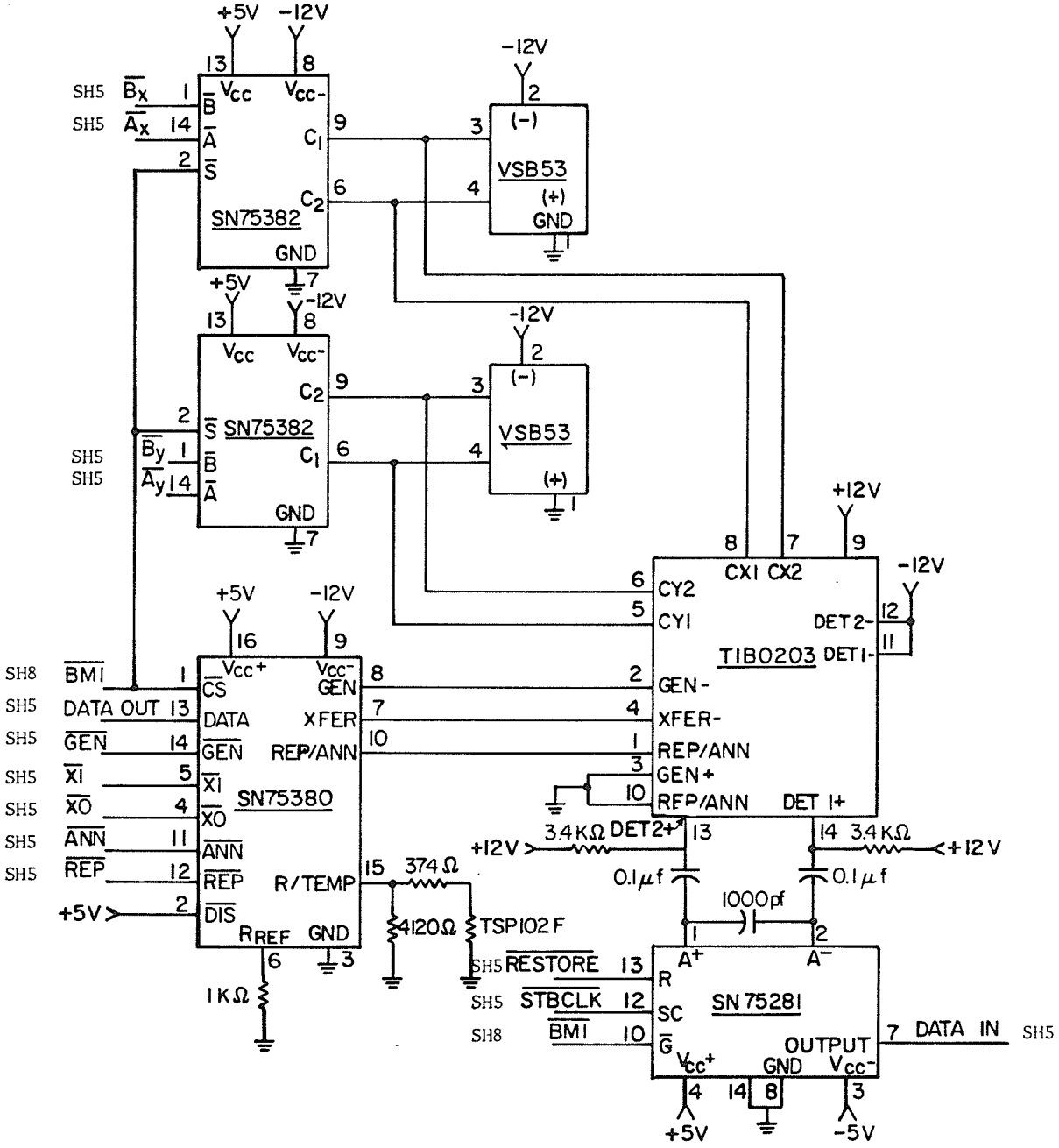
SHEET 6


ADDRESS
CONTROL
DATA
+5V
12
V_{DD}
.1μF
CS_0
8
A_9
CS_1
10
VMA
$\overline{CS}_2$
9
SH8
AC1A
DECODE
3
R_X CLK
RS
11
A_0
SH4
9
7408
8
BIT
RATE
10
4
T_X CLK
R/W
13
R/$\bar{W}$
E
14
E
7
$\overline{IRQ}$
+12V -12V
MC6850
4K7
+5V
14
1
TTL→RS232
3
MC1488
2
6
T_X DATA
330pf
7
D_0
22
D_0
D_1
21
D_1
D_2
20
D_2
D_3
19
D_3
D_4
18
D_4
D_5
17
D_5
D_6
16
D_6
D_7
15
D_7
+5V
14
RS232 → TTL
MC1489
3
2
R_X DATA
7
2
330pf
24
CLEAR TO SEND
25
DATA CARRIER DETECT
5
REQUEST TO SEND
V_{SS}
1

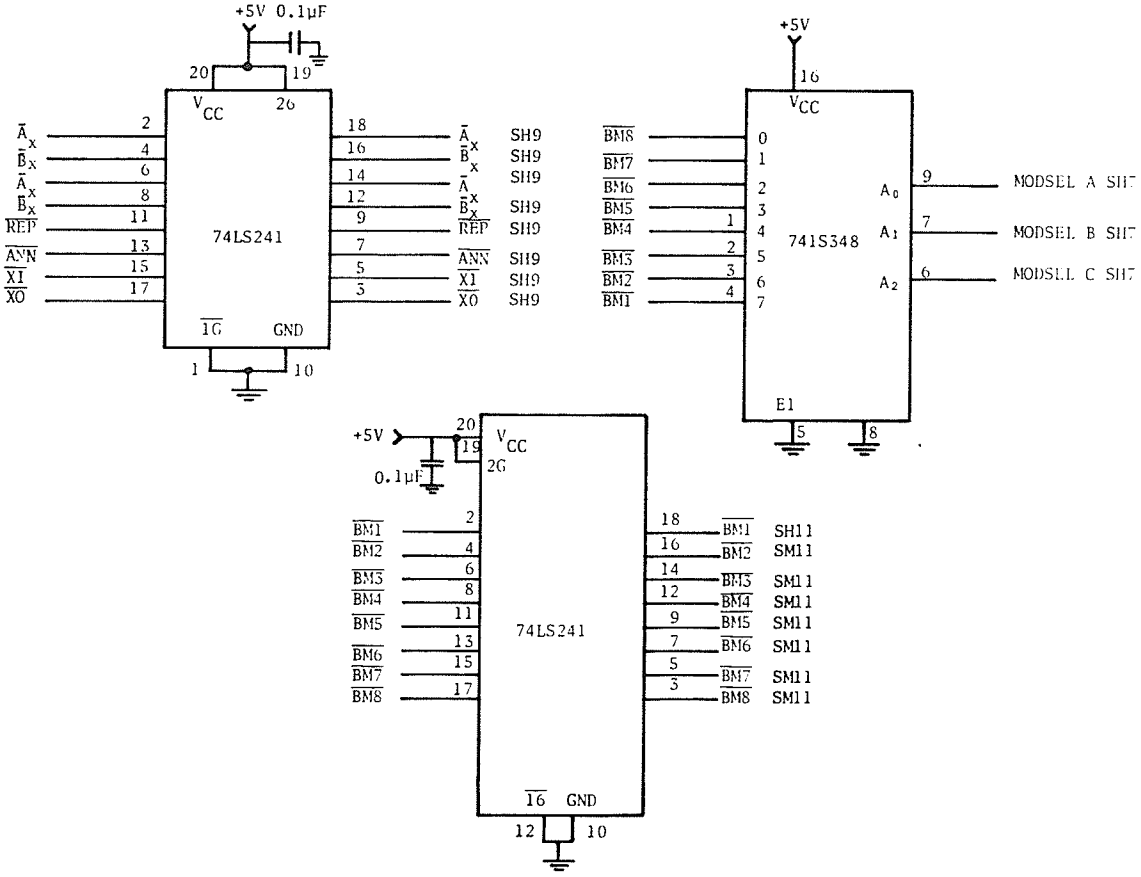

+5V
0.1µf
+5V
0.1µf
24
16
+5V
14
0.1µf
23
ADI
74393
COUNTER
CLOCK 2
CLOCK 1
CLEAR
CLEAR
SH5
INITRW
SH5
LPCLK 1
1QA
1QB
1QC
1QD
2QA
2QB
2QC
2QD
GND
7
6340-IJ
ADD 512x8
ADA
ADB
ADC
ADD
ADE
ADF
ADG
ADH
S3
S4
S1
S2
GND
+5V
21
20
12
74251
MUX
D0
D1
D2
D3
D4
D5
D6
D7
W
Y
6
N.C.
5
FUNCEN SH5
STROBE
7
GND
9
SH11
A 11 MODSEL A
SH11
B 10 MODSEL B
SH11
C 9 MODSEL C

SHEET 8

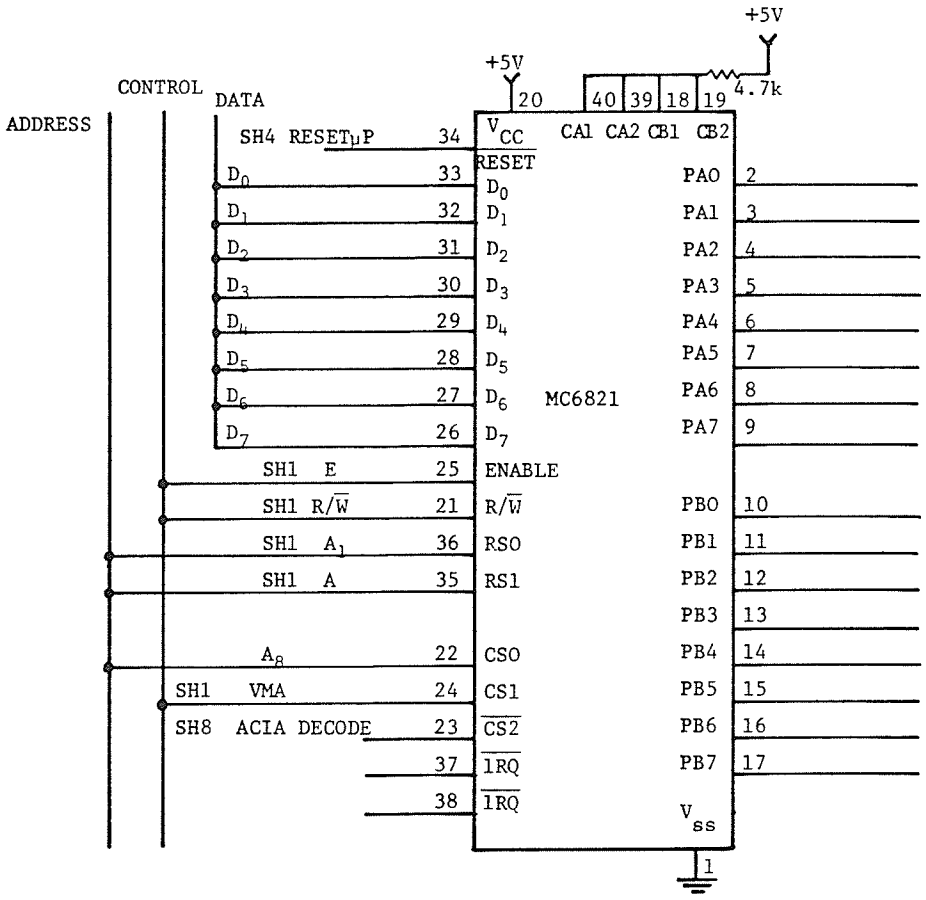

+5V
ADDRESS
CONTROL
.1µF
24
V_{CC}
74L154
A_{10}
A_{11}
A_{12}
A_{13}
23
22
21
20
A
B
C
D
OUT_{15}
OUT_{14}
OUT_{13}
OUT_{12}
OUT_{11}
OUT_{10}
OUT_9
OUT_8
OUT_7
OUT_6
OUT_5
OUT_4
OUT_3
OUT_2
OUT_1
OUT_0
17
16
15
14
13
11
10
9
8
7
6
5
4
3
2
1
PROM DECODE 4 SH2
PROM DECODE 3
PROM DECODE 2
PROM DECODE 1
RAM DECODE 10
RAM DECODE 9
RAM DECODE 8
RAM DECODE 7
RAM DECODE 6
RAM DECODE 5
RAM DECODE 4
RAM DECODE 3
RAM DECODE 2
RAM DECODE 1 SH3
GND
12
9
1/3 7410
8 19
G_2
G_1
18
VMA•E
ACIA DECODE SH6, SH11
CONTROL DECODE
12
1/4 7408
B
13
11
CONTROL DECODE SH5
+5V
15
14
V_{CC}
7410
GND
12
6
VMA
+5V
1
3
+5V
4
A_{10}
A_{11}
A_{13}
A_{12}
1
2
3
12
13
1A
1B
1C
1D
1E
74S260
2A 2B 2C 2D GND
18
24
11
5
13
6
2
4
5
7
8
9
10
A_7
A_8
A_6
A_5
A_4
12
13
74LS88
C
11
22
21
3
2
+5V
24
V_{CC}
STROBE
1
MC14515B
D
C
B
A
GND
INHIBIT
12
23
S_8
S_9
S_{10}
S_{11}
S_{12}
S_{13}
S_{14}
S_{15}
S_7
18
17
20
19
14
13
16
15
4
$\overline{BM1}$
$\overline{BM2}$
$\overline{BM3}$
$\overline{BM4}$
$\overline{BM5}$
$\overline{BM6}$
$\overline{BM7}$
$\overline{BM8}$
SH10
1/6 74LS14
1
2
POWERFAIL

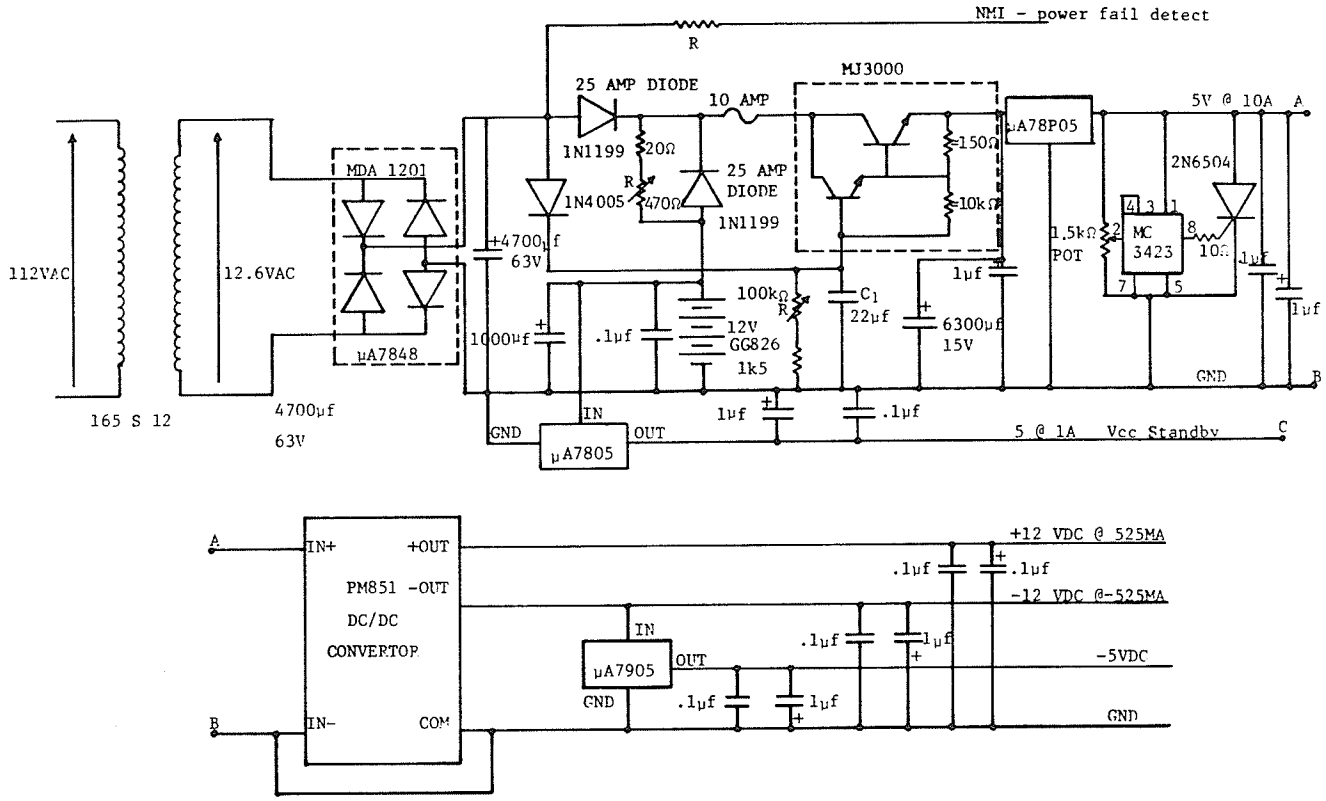

+5V
-12V
SN75382
VSB53
GND
+12V
CX1 CX2
CY2
CY1
DET2-
DET1-
TIB0203
GEN-
XFER-
REP/ANN
GEN+
REP/ANN
DET 1+
DET2+
3.4KΩ
0.1μf
1000pf
SN75380
R/TEMP
374Ω
4120Ω
TSP102F
RREF
1KΩ
SN75281
A+
A-
R
SC
G
OUTPUT
DATA IN
RESTORE
STBCLK
BM1
DATA OUT
GEN
XI
XO
ANN
REP
DIS
CS
DATA
XFER
REP/ANN
Vcc+
Vcc-
GND

SHEET 10


+5V 0.1µF
20
19
V_{CC}
26
74LS241
$\bar{A}_x$ 2
$\bar{B}_x$ 4
$\bar{A}_x$ 6
$\bar{B}_x$ 8
REP 11
ANN 13
XI 15
XO 17
1G
GND
1
10
18 $\bar{A}_x$ SH9
16 $\bar{B}_x$ SH9
14 $\bar{A}_x$ SH9
12 $\bar{B}_x$ SH9
9 REP SH9
7 ANN SH9
5 XI SH9
3 XO SH9
+5V
16
V_{CC}
741S348
BM8 0
BM7 1
BM6 2
BM5 3
BM4 4
BM3 5
BM2 6
BM1 7
E1
5
8
A_0 9 MODSEL A SH7
A_1 7 MODSEL B SH7
A_2 6 MODSEL C SH7
+5V
0.1µF
20
19
V_{CC}
2G
74LS241
BM1 2
BM2 4
BM3 6
BM4 8
BM5 11
BM6 13
BM7 15
BM8 17
16 GND
12
10
18 BM1 SM11
16 BM2 SM11
14 BM3 SM11
12 BM4 SM11
9 BM5 SM11
7 BM6 SM11
5 BM7 SM11
3 BM8 SM11

SHEET 11


+5V
+5V
4.7k
CONTROL
DATA
ADDRESS
SH4 RESET μP
34
20
40
39
18
19
V_{CC}
CA1
CA2
CB1
CB2
RESET
D_0
33
D_0
PA0
2
D_1
32
D_1
PA1
3
D_2
31
D_2
PA2
4
D_3
30
D_3
PA3
5
D_4
29
D_4
PA4
6
D_5
28
D_5
PA5
7
D_6
27
D_6
MC6821
PA6
8
D_7
26
D_7
PA7
9
SH1 E
25
ENABLE
SH1 R/$\overline{W}$
21
R/$\overline{W}$
PB0
10
SH1 A_1
36
RS0
PB1
11
SH1 A
35
RS1
PB2
12
PB3
13
A_8
22
CS0
PB4
14
SH1 VMA
24
CS1
PB5
15
SH8 ACIA DECODE
23
$\overline{CS2}$
PB6
16
37
$\overline{IRQ}$
PB7
17
38
$\overline{IRQ}$
V_{SS}
1

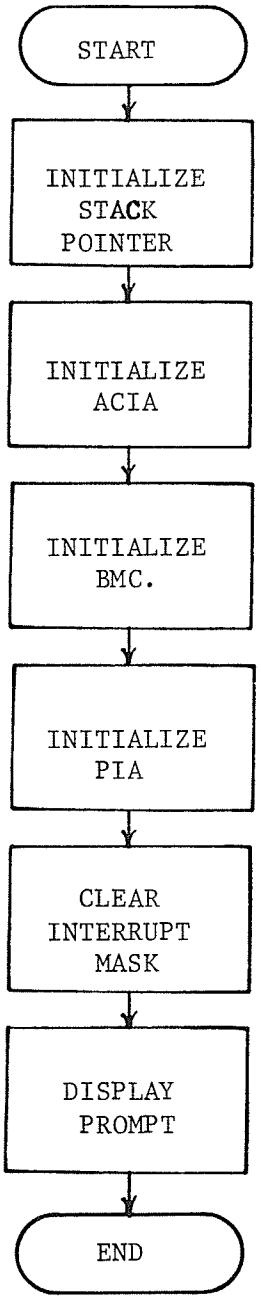


POWER SUPPLY DESIGN

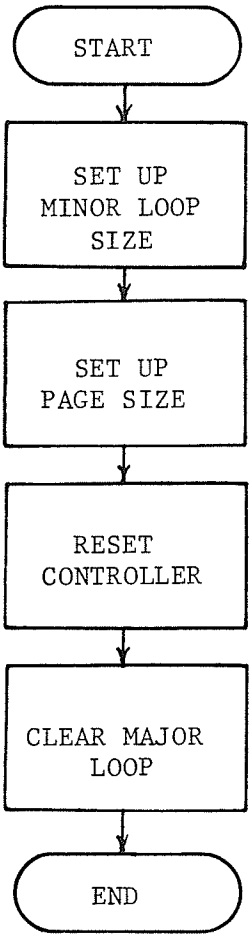
Appendix B

SYSTEM MONITOR

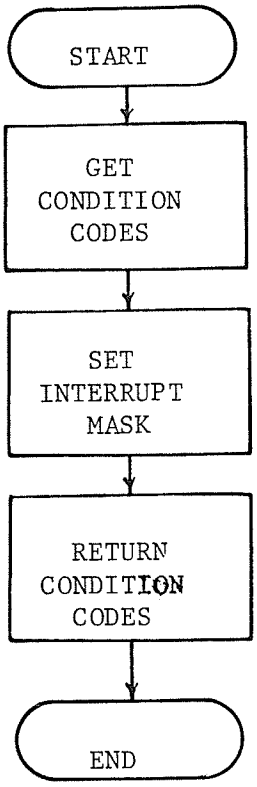
POWER UP INITIALIZATION



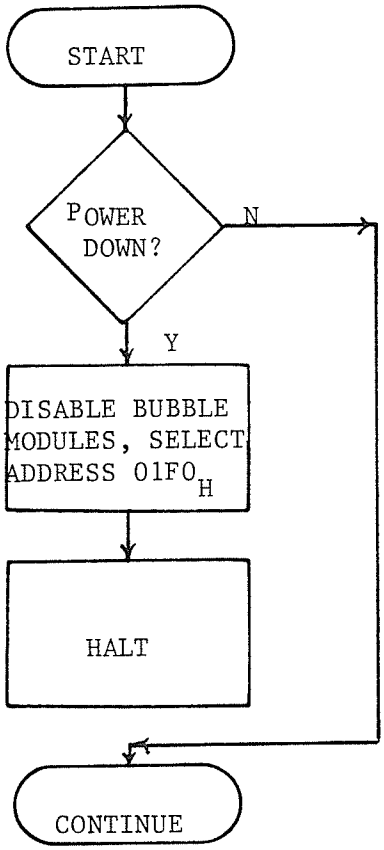
INITIALIZE BMC.

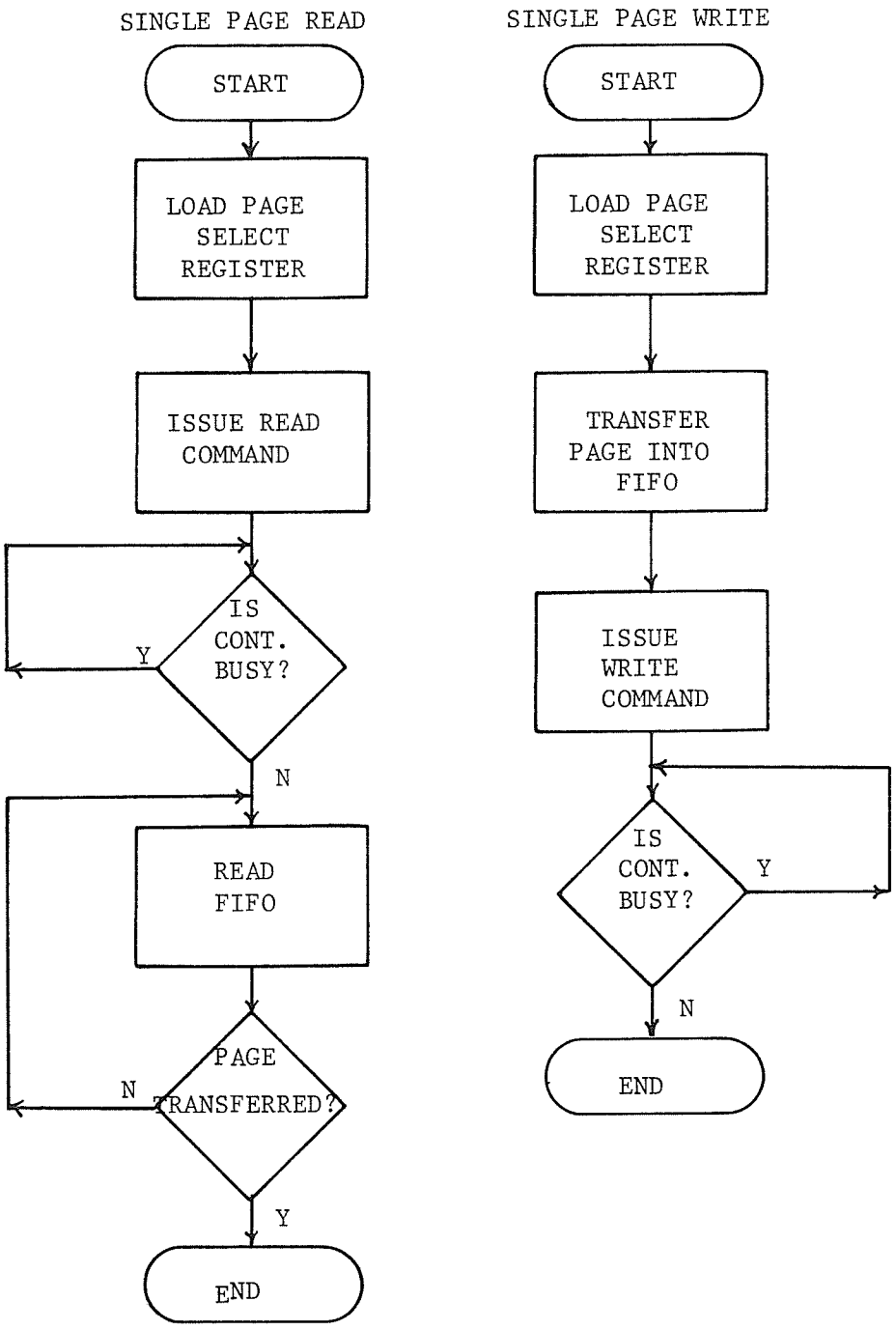


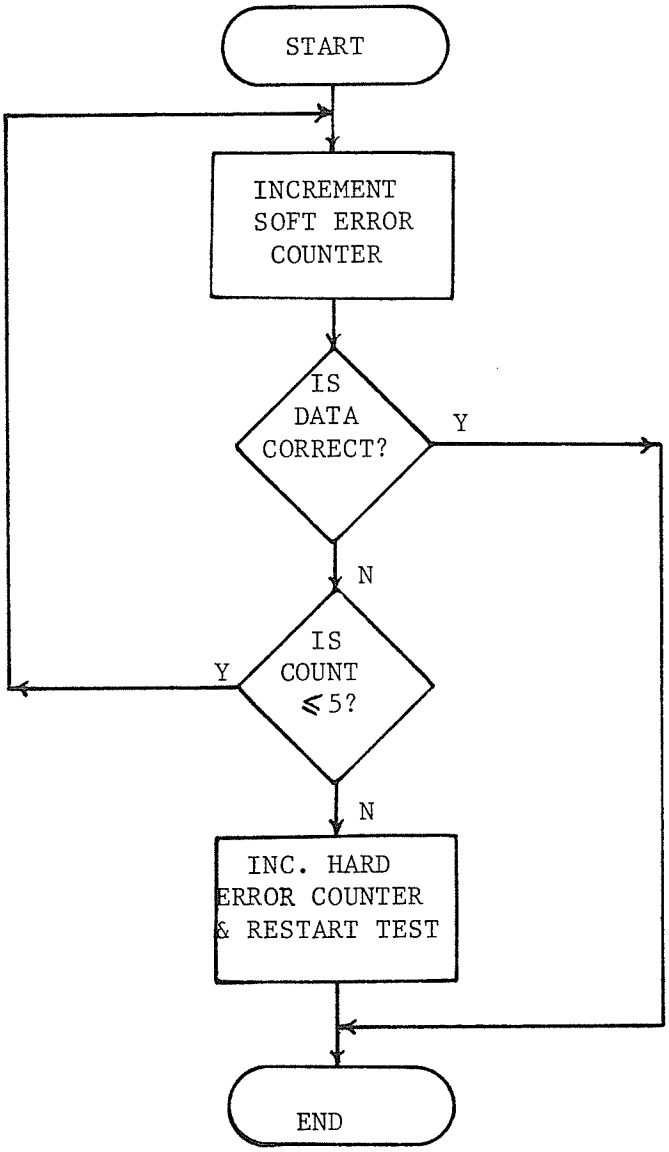
INTERRUPT ROUTINE



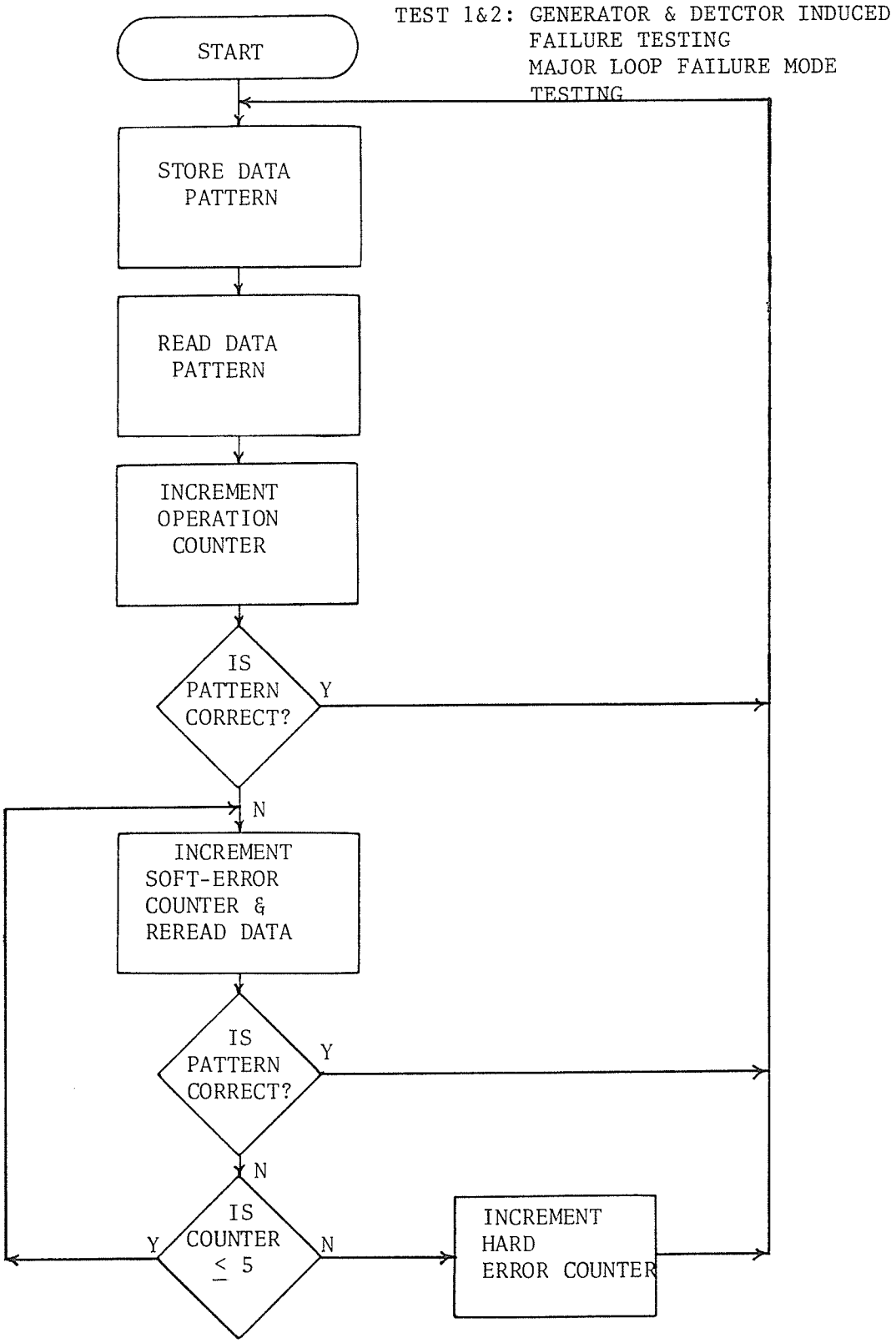
POWER DOWN ROUTINE

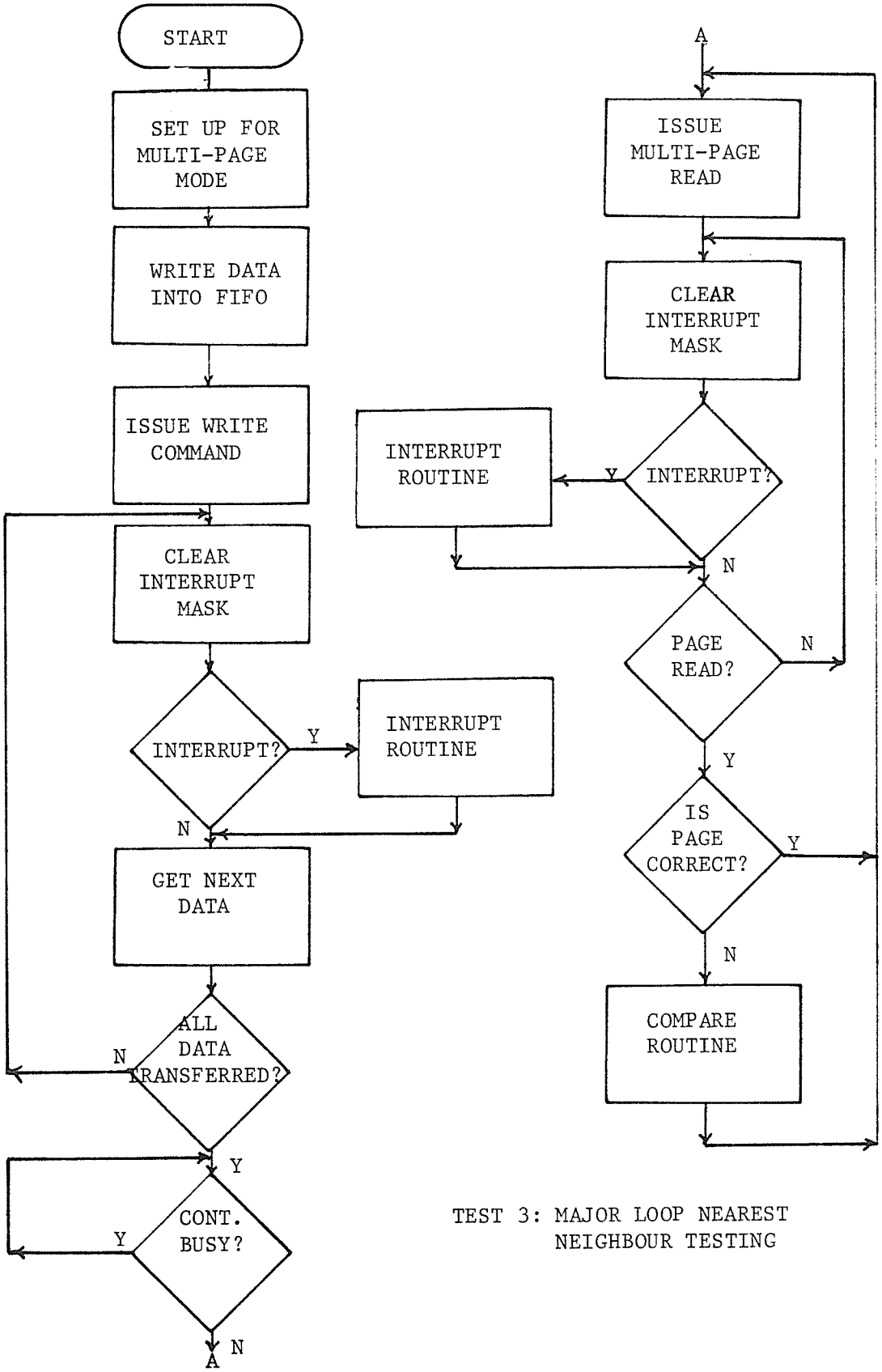



SINGLE PAGE READ
START
LOAD PAGE SELECT REGISTER
ISSUE READ COMMAND
IS CONT. BUSY?
Y
N
READ FIFO
PAGE TRANSFERRED?
N
Y
END
SINGLE PAGE WRITE
START
LOAD PAGE SELECT REGISTER
TRANSFER PAGE INTO FIFO
ISSUE WRITE COMMAND
IS CONT. BUSY?
Y
N
END



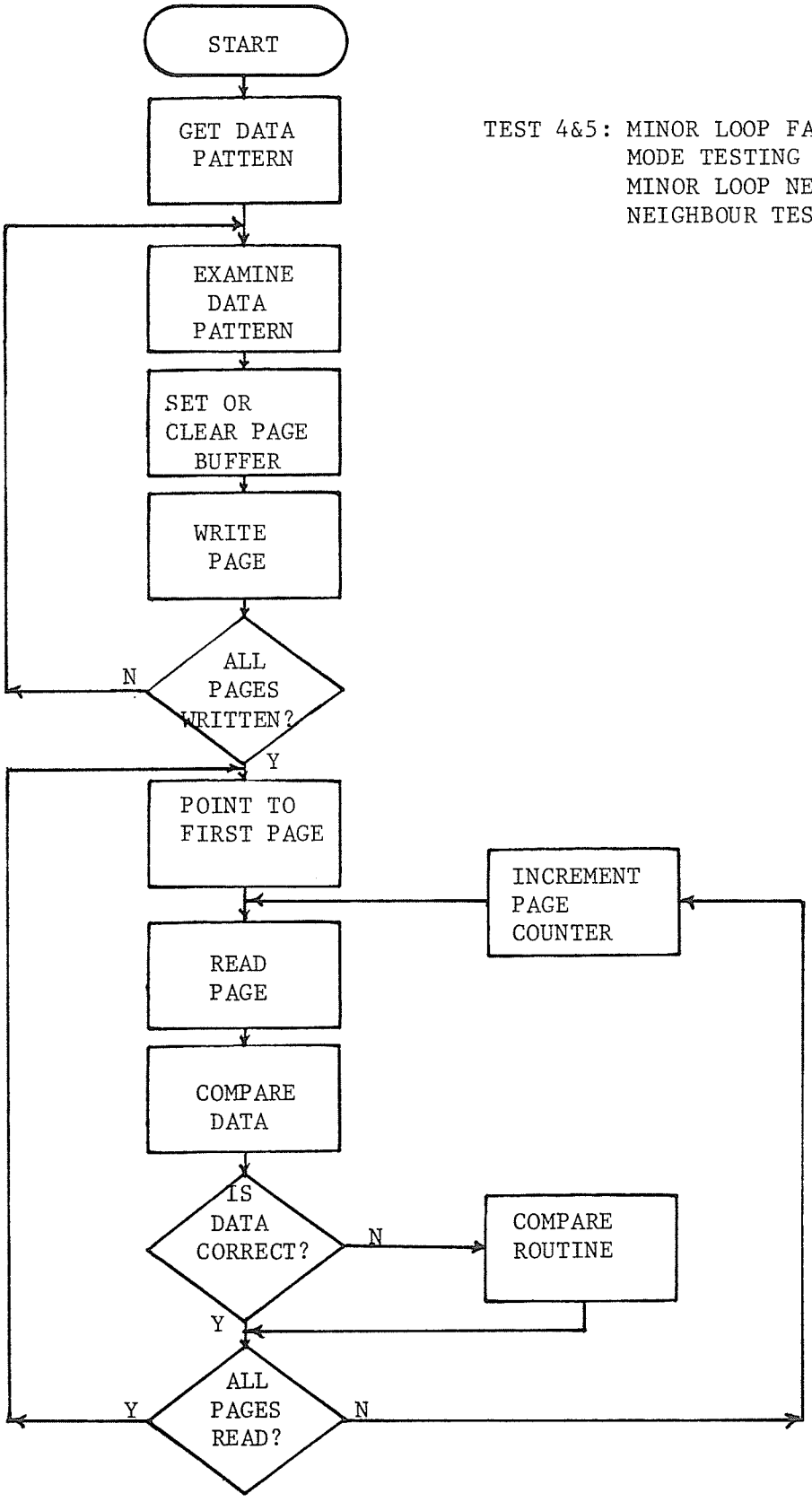
COMPARE ROUTINE


TEST 1&2: GENERATOR & DETCTOR INDUCED FAILURE TESTING
MAJOR LOOP FAILURE MODE TESTING
START
STORE DATA PATTERN
READ DATA PATTERN
INCREMENT OPERATION COUNTER
IS PATTERN CORRECT?
Y
N
INCREMENT SOFT-ERROR COUNTER & REREAD DATA
IS PATTERN CORRECT?
Y
N
IS COUNTER ≤ 5
Y
N
INCREMENT HARD ERROR COUNTER



TEST 3: MAJOR LOOP NEAREST NEIGHBOUR TESTING

TEST 4&5: MINOR LOOP FAILURE MODE TESTING
MINOR LOOP NEAREST NEIGHBOUR TESTING



```
                        BUBBLE MEMORY SYSTEM MONITOR
     0100   PIADATAA EQU 0100H      ;PIA DATA REG. A
     0101   PIACONTA EQU 0101H      ;PIA CONT REG. A
     0102   PIADATAB EQU 0102H      ;PIA DATA REG. B
     0103   PIACONTB EQU 0103H      ;PIA CONT REG. B
     2000   DELAY    EQU 2000H      ;TIME DELAY FOR POWER FAIL
     01F0   DISABLE  EQU 01F0H      ;ADDRESS CALLED TO DISABLE BUBBLE MODULES
     0082   RE       EQU 82H        ;READ FIFO ADDR.
     0082   RDCMD    EQU 82H        ;SINGLE PAGE READ COMMAND
     0080   PGSEL    EQU 80H        ;PAGE SEL REG. ADDR
     0083   CONT     EQU 83H        ;BUBBLE CONT REG. ADDR.
     0084   STAS     EQU 84H        ;STATUS REG. ADDR.
     0085   WR       EQU 85H        ;WRITE FIFO ADDR.
     0084   WRCMD    EQU 84H        ;SINGLE PAGE WRITE CMD.
     0086   PGCNT    EQU 86H        ;PAGE COUNT REG. ADDR.
     0012   MPRDCMD  EQU 12H        ;MULTI PAGE READ COMMAND
     0010   MULTI    EQU 10H        ;COMMAND TO INIT. MULTIPAGE MODE
     0014   MPWRCMD  EQU 14H        ;MULTI PAGE WRITE COMMAND
     0080   RESETINT EQU 80H        ;COMMAND TO RESET INTERRUPT
     0200   ACIACT   EQU 0200H      ;ACIA CONT REG. ADDR.
     0201   ACIATR   EQU 0201H      ;ACIA I/O REG. ADDR.
     0088   MINLP    EQU 88H        ;MINOR LOOP SIZE REG. ADDR.
     008D   PGSIZE   EQU 8DH        ;PAGE SIZE REG. ADDR.
     0020   TPSTK    EQU 20H        ;TOP OF STACK
     0021   BUF      EQU TPSTK+1    ;PAGE BUFFER BEG. ADDR.
     0033   ENDBUF   EQU BUF+12H    ;END OF PAGE BUFFER
     0281   ENDMEM   EQU 0281H      ;END OF BUBBLE MEMORY
     0034   BUF1     EQU ENDBUF+1   ;MULTI-PAGE BUFFER
     0058   ENDBUF1  EQU BUF1+36    ;END OF MULTI-PAGE BUFFER

     0059           ORG 59H

0059 0001   LOC      BLOCK 1        ;ASCII OF HEX DATA
005A 0002   TEMP1    BLOCK 2        ;STORE ADDR. OF DUMP START
005C 0002   TEMP2    BLOCK 2        ;INDEX REG TEMP STORE 2
005E 0002   TEMP3    BLOCK 2        ;DUMP END ADDR. STORE
0060 0002   TEMP4    BLOCK 2        ;PAGE START ADDR. TEMP STORE
0062 0002   TEMP     BLOCK 2        ;PAGE SEL ADDR. STORE
0064 0001   LOC1     BLOCK 1        ;TEMP REG.
0065 0002   TEMP5    BLOCK 2        ;STACK POINTER TEMP STORE
0067 0002   TEMP6    BLOCK 2        ;NO. OF SOFT ERRORS
0069 0002   TEMP7    BLOCK 2        ;MS. NO. OF R/W OPERATIONS
006B 0002   TEMP8    BLOCK 2        ;LS. NO. OF R/W OPERATIONS
006D 0002   TEMP9    BLOCK 2        ;USED TO TRANSFER INDEX REGISTER
```

```
006F 0002   TEMP10   BLOCK 2        ;STORE STARTING PAGE ADDRESS
0071 0002   TEMP11   BLOCK 2        ;THIRD TEMP. STORAGE FOR INDEX REGISTER
0073 0001   LOC2     BLOCK 1        ;TEMP. REG USED FOR HARD ERROR COUNTER
0074 0001   LOC3     BLOCK 1        ;USED AS TEMPORARY COUNTER
     F000   BOS      ORG  0F000H


F000 8E0020 BEGIN  LDS    #TPSTK    ;INIT. STACK POINTER
F003 BDF0EA        JSR    ACIAIT    ;INIT. ACIA
F006 BDF0F5        JSR    BUBINT    ;INIT. BUBBLE MEMORY CONT.
F009 CEFE04        LDX    #0FE04H   ;INIT. PIA PORT A FOR 1 INPUT AND REST AS OUTPUTS
F00C FF0100        STX    PIADATAA  ;STORE AT PIA CONT. REG. A
F00F CEFF04        LDX    #0FF04H   ;INIT. PIA PORT B FOR OUTPUTS
F012 FF0102        STX    PIADATAB  ;STORE AT PIA CONT. REG. B
F015 0E            CLI              ;CLEAR INTERRUPT MASK
F016 CEFB29        LDX    #PRMPT1   ;DISPLAY INIT. PROMPT
F019 BDF1BE        JSR    MESSA
F01C CEFB3B START  LDX    #PROMPT   ;DISPLAY SECOND PROMPT
F01F BDF1BE        JSR    MESSA
F022 8642          LDA A  #"B"      ;SET UP FOR HEX INPUT ONLY
F024 9759          STA A  @LOC
F026 CE0000        LDX    #0
F029 FF0073        STX    LOC2      ;CLEAR TEMPORARY REGISTERS
F02C FF0067        STX    TEMP6
F02F FF0069        STX    TEMP7
F032 FF006B        STX    TEMP8
F035 BDF101        JSR    BUB       ;RESET CONTROLLER
F038 BDF11C REPT   JSR    INPUT
F03B 8145          CMP A  #"E"      ;CHECK FOR E
F03D 2758          BEQ    COM1
F03F 8143          CMP A  #"C"      ;CHECK FOR C
F041 2757          BEQ    COM2
F043 8144          CMP A  #"D"      ;CHECK FOR D
F045 2756          BEQ    COM3
F047 8147          CMP A  #"G"      ;CHECK FOR G
F049 2755          BEQ    COM4
F04B 8152          CMP A  #"R"      ;CHECK FOR R
F04D 2754          BEQ    COM5
F04F 8148          CMP A  #"H"      ;CHECK FOR H
F051 2753          BEQ    COM6
F053 8146          CMP A  #"F"      ;CHECK FOR F
F055 2752          BEQ    COM7
F057 814C          CMP A  #"L"      ;CHECK FOR L
F059 2751          BEQ    COM8
F05B 8154          CMP A  #"T"      ;CHECK FOR T
F05D 2750          BEQ    COM9
F05F 8149          CMP A  #"I"      ;CHECK FOR I
F061 274F          BEQ    COM10
F063 8153          CMP A  #"S"      ;CHECK FOR S
F065 274E          BEQ    COM11
F067 8156          CMP A  #"V"      ;CHECK FOR V
F069 275F          BEQ    COM12
F06B 8131          CMP A  #"1"      ;CHECK FOR 1
```

```
F06D 2749          BEQ    COM13
F06F 8132          CMP A  #"2"      ;CHECK FOR 2
F071 2748          BEQ    COM14
F073 8133          CMP A  #"3"      ;CHECK FOR 3
F075 2747          BEQ    COM15
F077 8134          CMP A  #"4"      ;CHECK FOR 4
F079 2746          BEQ    COM16
F07B 8135          CMP A  #"5"      ;CHECK FOR 5
F07D 2745          BEQ    COM17
F07F 8136          CMP A  #"6"      ;CHECK FOR 6
F081 2744          BEQ    COM18
F083 8D02          BSR    REJECT    ;COMMAND REJECT
F085 20B1          BRA    REPT      ;CYCLE THRU COMMAND LIST

           ;THIS ROUTINE IS CALLED IF AN INVALID COM.
           ;IS ENCOUNTERED

F087 8608   REJECT LDA A  #08H      ;CLEAR CHAR. TRANSMITTED
F089 BDF145        JSR    TRANS      ;IF CHAR NOT VALID
F08C 8620          LDA A  #20H      ;TRANSMITT SPACE
F08E BDF145        JSR    TRANS
F091 8608          LDA A  #08H      ;TRANSMITT BACK-SPACE
F093 BDF145        JSR    TRANS
F096 39            RTS

           ;COMMAND SELECTION

F097 7EF293 COM1   JMP    EXAM
F09A 7EF3A7 COM2   JMP    CNTNUE
F09D 7EF20A COM3   JMP    DUMP
F0A0 7EF283 COM4   JMP    GO
F0A3 7EF466 COM5   JMP    READ
F0A6 7EF10D COM6   JMP    HELP
F0A9 7EF560 COM7   JMP    FILL
F0AC 7EF58F COM8   JMP    LIST1
F0AF 7EF5B6 COM9   JMP    TRSFER
F0B2 7EF6CB COM10  JMP    INTLZE
F0B5 7EF6D7 COM11  JMP    START1
F0B8 7EF72F COM13  JMP    TEST1     ;GENERATOR AND DETECTOR TESTING
F0BB 7EF73F COM14  JMP    TEST2     ;MAJOR LOOP FAILURE MODE TESTING
F0BE 7EF801 COM15  JMP    TEST3     ;MAJOR LOOP NEAREST NEIGHBOR TESTING
F0C1 7EF914 COM16  JMP    TEST4     ;MINOR LOOP FAILURE MODE TESTING
F0C4 7EF924 COM17  JMP    TEST4A    ;MINOR LOOP NEAREST NEIGHBOR TESTING
F0C7 7EFA23 COM18  JMP    TEST5     ;TEST EACH MINOR LOOP WITH TEST PATTERN
F0CA BDF43F COM12  JSR    VERFY
F0CD 7EF01C        JMP    START

           ;DETERMINE IF ASCII OR HEX IS TO BE USED

F0D0 CEFB40 AB     LDX    #MESAGE    ;OUTPUT PROMPT
F0D3 BDF1BE        JSR    MESSA
F0D6 BDF11C E1     JSR    INPUT
F0D9 8141          CMP A  #"A"      ;CHECK FOR A
```

```
F0DB 270A          BEQ    E9        ;SET UP FOR ASCII INPUTS
F0DD 8148          CMP A  #"H"      ;CHECK FOR H
F0DF 2704          BEQ    E2        ;SET UP FOR HEX INPUTS
F0E1 8DA4          BSR    REJECT    ;REJECT COMMAND
F0E3 20F1          BRA    E1        ;REPEAT
F0E5 8642   E2     LDA A  #"B"      ;SET HEX INPUT FLAG
F0E7 9759   E9     STA A  @LOC      ;STORE FLAG
F0E9 39            RTS

            ;INITIALIZE ACIA

F0EA 8603   ACIAIT LDA A  #03H      ;ACIA MASTER RESET
F0EC B70200        STA A  ACIACT
F0EF 8649          LDA A  #49H      ;INIT. ACIA FOR RTS=HIGH
F0F1 B70200        STA A  ACIACT    ;7BITS+E PARITY+1 STP BIT,/16 MODE
F0F4 39            RTS

            ;INITIALIZE BUBBLE MEMORY CONT.

F0F5 8602   BUBINT LDA A  #02H      ;INIT. MINOR LOOP SIZE
F0F7 9788          STA A  MINLP     ;TO 0281 HEX -MS. BYTE
F0F9 8681          LDA A  #81H
F0FB 9789          STA A  MINLP+1    ;LS. BYTE
F0FD 8612          LDA A  #12H      ;INIT. PAGE SIZE TO 18 BYTES
F0FF 978D          STA A  PGSIZE
F101 8640   BUB    LDA A  #40H      ;MASTER RESET
F103 9783          STA A  CONT
F105 8601          LDA A  #01H      ;ISSUE INIT. COMMAND
F107 9783          STA A  CONT
F109 BDF30F        JSR    STAT
F10C 39            RTS

            ;HELP COMMAND

F10D CEFCBB HELP   LDX    #M26      ;OUTPUT PROMPT
F110 BDF1C4        JSR    MESS
F113 CEFD5E F19    LDX    #M28      ;OUTPUT HELP TABLE
F116 BDF1BE        JSR    MESSA
F119 7EF01C        JMP    START     ;RESTART

            ;TERMINAL INPUT RETURNED IN ACC A

F11C B60200 INPUT  LDA A  ACIACT    ;LOAD CONT REG.
F11F 8530          BIT A  #30H      ;CHECK FOR ERROR
F121 2702          BEQ    NOERR
F123 8D16          BSR    ERROR     ;BRANCH ON ERROR CONDITION
F125 47     NOERR  ASR A             ;CHECK FOR FULL BUFFER IF NOERR
F126 24F4          BCC    INPUT     ;RE-CHECK IF NO INPUT
F128 B60201        LDA A  ACIATR    ;INPUT CHAR.
F12B 811B          CMP A  #1BH      ;CHECK FOR RESTART,ESC
F12D 2603          BNE    A1
F12F 7EF01C        JMP    START     ;RESTART ON ESC CHAR.
F132 810A   A1     CMP A  #0AH      ;CHECK FOR LINE FEED
```

```
F134 2601          BNE    A2
F136 4F            CLR A            ;STRIP OFF LINE FEEDS
F137 BDF145 A2     JSR    TRANS     ;ECHO CHAR.
F13A 39            RTS
F13B CEFB11 ERROR  LDX    #M1       ;OUTPUT ERROR MESSAGE
F13E BDF1C4        JSR    MESS
F141 B60200        LDA A  ACIACT    ;CLEAR ERROR FLAG
F144 39            RTS

           ;OUTPUT CHAR VIA ACC A

F145 37    TRANS   PSH B            ;SAVE REGISTERS
F146 36            PSH A
F147 8D1F          BSR    TRANS1    ;CHECK FOR EMPTY TRANS BUFFER
F149 B70201        STA A  ACIATR    ;OUTPUT CHAR
F14C 01            NOP              ;THE NOP IS USED AS A SPACE FILLER
F14D 01            NOP
F14E 810D          CMP A  #0DH      ;CHECK FOR CR
F150 2613          BNE    A3        ;AVOID THE FOLLOWING IF POSSIBLE
F152 C607          LDA B  #07H      ;LOAD NULL COUNT
F154 8600          LDA A  #00H      ;INSERT NULLS AFTER CR
F156 8D10  A4      BSR    TRANS1    ;CHECK IF CLEAR TO TRANSMITT
F158 B70201        STA A  ACIATR    ;TRANSMITT NULLS
F15B 5A            DEC B            ;DECREMENT NULL COUNT
F15C 26F8          BNE    A4        ;CHECK IF ALL TRANSFERRED
F15E 860A          LDA A  #0AH      ;OUTPUT LINE FEED
F160 8D06          BSR    TRANS1    ;CHECK IF CLEAR TO TRANSMITT
F162 B70201        STA A  ACIATR    ;TRANS. LINE-FEED
F165 32    A3      PUL A            ;RESTORE REGISTERS
F166 33            PUL B
F167 39            RTS

           ;THIS CHECKS IF IT IS CLEAR TO TRANSMITT
F
F168 37    TRANS1  PSH B            ;SAVE REGISTER
F169 F60200 TRANS2 LDA B  ACIACT    ;CHECK IF TRANSMIT
F16C C502          BIT B  #02H      ;BUFFER IS EMPTY
F16E 27F9          BEQ    TRANS2    ;REPEAT IF NOT CLEAR
F170 33            PUL B            ;RESTORE REGISTER
F171 39            RTS

           ;CONVERTS ASCII TO HEX
F
F172 8540  CNVERH  BIT A  #40H      ;CHECK IF A-F
F174 2702          BEQ    DOWN
F176 8B09          ADD A  #09H      ;CONVERT TO ASCII
F178 840F  DOWN    AND A  #0FH
F17A 39            RTS

           ;CONVERTS HEX TO ASCII

F17B 8B90  CNVERA  ADD A  #90H      ;CONVERT TO HEX
F17D 19            DAA
```

```
F17E 8940          ADC A  #40H
F180 19            DAA
F181 39            RTS

            ;BYTE RETURNED IN ACC A - USED TO FORM BYTE OF DATA

F182 BDF11C PACK   JSR    INPUT     ;CONVERT 2 ACSII CHARS
F185 37     PACK1  PSH B             ;INTO A BYTE
F186 8D1A   PACK4  BSR    CHECK     ;CHECK FOR VALID BIN.
F188 2513          BCS    PACK3     ;REREAD IF NOT VALID BINARY
F18A 8DE6          BSR    CNVERH    ;CONVERT ASCII TO HEX
F18C 16            TAB
F18D BDF11C PACK2  JSR    INPUT     ;INPUT SECOND ASCII CHAR.
F190 8D10          BSR    CHECK     ;CHECK IF VALID BIN.
F192 25F9          BCS    PACK2     ;REREAD IF NOT VALID BINARY
F194 8DDC          BSR    CNVERH    ;CONVERT ASCII TO HEX
F196 58            ASL B            ;PACK INTO BYTE
F197 58            ASL B
F198 58            ASL B
F199 58            ASL B
F19A 1B            ABA              ;BYTE OF DATA IN ACC - A
F19B 33            PUL B
F19C 39            RTS
            ;THIS SECTION REREADS INPUT
F19D BDF11C PACK3  JSR    INPUT
F1A0 20E4          BRA    PACK4
            ;THIS SECTION CHECKS FOR VALID INPUTS
F1A2 36     CHECK  PSH A             ;CHECK FOR VALID BIN.
F1A3 8146          CMP A  #"F"       ;CHECK IF VALUE BETWEEN A-F
F1A5 2211          BHI    D1
F1A7 812F          CMP A  #2FH
F1A9 230D          BLS    D1
F1AB 8140          CMP A  #40H       ;CHECK IF VALUE BETWEEN 0-9
F1AD 2709          BEQ    D1
F1AF 840F          AND A  #0FH
F1B1 8109          CMP A  #09H
F1B3 2203          BHI    D1
F1B5 32            PUL A
F1B6 0C            CLC
F1B7 39            RTS
F1B8 BDF087 D1     JSR    REJECT    ;REJECT IF INVALID HEX.
F1BB 32            PUL A
F1BC 0D            SEC
F1BD 39            RTS

            ;OUTPUT A MESSAGE

F1BE 36     MESSA  PSH A             ;SAVE REGISTER
F1BF 860D          LDA A  #0DH
F1C1 8D82          BSR    TRANS     ;OUTPUT CR
F1C3 32            PUL A             ;MAINTAIN STACK POINTER
F1C4 36     MESS   PSH A
F1C5 A600   ABOVE  LDA A  0,X        ;POINT TO MESSAGE
```

```
F1C7 8140          CMP A  #"@"      ;CHECK FOR END OF MESSAGE
F1C9 2706          BEQ    UNDER     ;EXIT IF MESSAGE TRANSMITTED
F1CB BDF145        JSR    TRANS     ;TRANS CHAR.
F1CE 08            INX              ;POINT TO NEXT CHAR.
F1CF 20F4          BRA    ABOVE     ;REPEAT
F1D1 8620   UNDER  LDA A  #20H
F1D3 BDF145        JSR    TRANS     ;OUTPUT A SPACE
F1D6 32            PUL A            ;RESTORE REGISTER
F1D7 39            RTS

            ;OUTPUT ONE BYTE OF DATA BY COVERTING
            ;ONE BYTE INTO TWO ASCII CHARS.

F1D8 37     BREAK  PSH B
F1D9 16            TAB
F1DA 84F0          AND A  #0F0H     ;MASK OF MS. NIBBLE
F1DC C40F          AND B  #0FH      ;MASK OF LS. NIBBLE
F1DE 44            LSR A            ;SHIFT BYTE INTO POSITION
F1DF 44            LSR A
F1E0 44            LSR A
F1E1 44            LSR A
F1E2 8D97          BSR    CNVERA    ;CONVERT MS. NIBBLE TO ASCII
F1E4 BDF145        JSR    TRANS     ;OUTPUT MS. NIBBLE
F1E7 17            TBA
F1E8 8D91          BSR    CNVERA    ;CONVERT LS. NIBBLE TO ASCII
F1EA BDF145        JSR    TRANS     ;OUTPUT LS. NIBBLE
F1ED 33            PUL B
F1EE 39            RTS

            ;GET 2 BYTE INPUT FROM TERMINAL

F1EF 8D91   ADDR   BSR    PACK      ;PACK TWO CHAR. INTO ONE BYTE
F1F1 975A          STA A  @TEMP1    ;SAVE BYTE
F1F3 8D8D          BSR    PACK      ;PACK TWO CHAR. INTO ONE BYTE
F1F5 975B          STA A  @TEMP1+1  ;SAVE BYTE
F1F7 DE5A          LDX    @TEMP1    ;LOAD TWO BYTE VALUE
F1F9 39            RTS

            ;OUTPUT TWO BYTES

F1FA 860D   BREAK1 LDA A  #0DH      ;OUTPUT CR
F1FC BDF145        JSR    TRANS
F1FF DF5C   BREAK2 STX    @TEMP2    ;SAVE DATA TO BE TRANSMITTED
F201 965C          LDA A  @TEMP2    ;LOAD WITH MS. BYTE
F203 8DD3          BSR    BREAK     ;TRANS. MS. BYTE
F205 965D          LDA A  @TEMP2+1  ;LOAD WITH LS. BYTE
F207 8DCF          BSR    BREAK     ;TRANS. LS. BYTE
F209 39            RTS

            ;DUMP CONTENTS OF RAM

F20A CEFB25 DUMP   LDX    #M3       ;OUTPUT REST OF PROMPT
F20D BDF1C4        JSR    MESS
```

```
F210 8620          LDA A  #" "
F212 BDF145        JSR    TRANS     ;OUTPUT SPACE
F215 BDF145        JSR    TRANS     ;OUTPUT ANOTHER SPACE
F218 BDF182        JSR    PACK      ;READ MS. BYTE OF START ADDRESS
F21B 975A          STA A  @TEMP1    ;GET MS. BYTE OF START ADDR.
F21D BDF182        JSR    PACK      ;READ LS. BYTE OF START ADDR.
F220 975B          STA A  @TEMP1+1   ;GET LS. BYTE OF START ADDR.
F222 8620          LDA A  #20H      ;OUTPUT SPACE
F224 BDF145        JSR    TRANS
F227 BDF145        JSR    TRANS     ;OUTPUT ANOTHER SPACE
F22A BDF182        JSR    PACK      ;READ MS. BYET OF END ADDR.
F22D 975E          STA A  @TEMP3    ;GET MS. BYTE OF END ADDR.
F22F BDF182        JSR    PACK      ;READ LS. BYTE OF END ADDR.
F232 975F          STA A  @TEMP3+1  ;GET LS. BYTE OF END ADDR.
F234 DE5A          LDX    @TEMP1    ;LOAD REG. WITH START ADDR.
F236 C610    A10   LDA B  #16       ;SET UP BYTE COUNTER
F238 BDF1FA DUMP1  JSR    BREAK1    ;OUTPUT ADDRESS
F23B 09            DEX
F23C 08      A9    INX
F23D 8620          LDA A  #20H      ;OUTPUT SPACE
F23F BDF145        JSR    TRANS
F242 A600          LDA A  0,X       ;GET DATA
F244 BDF1D8        JSR    BREAK     ;TRANSMITT DATA
F247 5A            DEC B            ;DECREMENT DATA COUNTER
F248 26F2          BNE    A9        ;16 BYTES OUTPUTED?
F24A DF5C          STX    @TEMP2    ;SAVE ADDR. POINTER
F24C C605          LDA B  #05H      ;LOAD WITH NO. OF SPACES
F24E 8620    A11   LDA A  #" "      ;OUTPUT 5 SPACES
F250 BDF145        JSR    TRANS
F253 5A            DEC B            ;DEC. SPACE COUNTER
F254 26F8          BNE    A11       ;CONTINUE IF NOT FINISHED
F256 DE5A          LDX    @TEMP1    ;GET START ADDR. AGAIN
F258 A600    A12   LDA A  0,X       ;GET DATA
F25A 2B08          BMI    A20
F25C 811F          CMP A  #1FH      ;CHECK TO SEE IF VALID ASCII
F25E 2304          BLS    A20
F260 817E          CMP A  #7EH
F262 2302          BLS    A13
F264 862E    A20   LDA A  #"."
F266 BDF145 A13    JSR    TRANS     ;OUTPUT ASCII IF VALID
F269 9C5E          CPX    @TEMP3    ;OR DOT IF INVALID ASCII
F26B 2713          BEQ    A8        ;EXIT IF END REACHED
F26D 08            INX              ;INCREMENT ADDR. COUNTER
F26E 9C5E          CPX    @TEMP3    ;HAS END BEEN REACHED?
F270 270E          BEQ    A8        ;EXIT IF END REACHED
F272 9C5C          CPX    @TEMP2    ;HAS ALL ASCII BEEN TRANSMITTED?
F274 26E2          BNE    A12       ;CONTINUE IF NOT
F276 DE5C          LDX    @TEMP2    ;RESTORE ADDRESS COUNTER
F278 08            INX              ;FOR NEXT LINE OF OUTPUT
F279 DF5A          STX    @TEMP1    ;SAVE ADDR. COUNTER
F27B BDF3EE        JSR    IO        ;GET KEYBOARD ENTRY IF ANY
F27E 20B6          BRA    A10       ;OUTPUT STARTING OF NEXT LINE
F280 7EF01C A8     JMP    START     ;RESTART
```

```
            ;EXECUTE PROGRAM

F283 CEFCBF GO      LDX     #M27        ;OUTPUT PROMPT
F286 BDF1C4         JSR     MESS
F289 BDF182         JSR     PACK        ;GET MS. BYTE OF ADDRESS
F28C 16             TAB
F28D BDF182         JSR     PACK        ;GET LS. BYTE OF ADDRESS
F290 36             PSH A
F291 37             PSH B
F292 39             RTS                 ;START EXECUTION

            ;EXAMINE MEMORY LOCATION

F293 CEFB21 EXAM    LDX     #M2         ;OUTPUT PROMPT
F296 BDF1C4         JSR     MESS
F299 8620           LDA A   #20H
F29B BDF145         JSR     TRANS       ;OUTPUT SPACE
F29E BDF182         JSR     PACK        ;GET MS. BYTE OF ADDRESS
F2A1 16             TAB
F2A2 BDF182         JSR     PACK        ;GET LS. BYTE OF ADDRESS
F2A5 36             PSH A               ;PUT ADDRESS ON STACK
F2A6 37             PSH B
F2A7 30             TSX
F2A8 EE00           LDX     0,X         ;LOAD REG. WITH EXAM ADDRESS
F2AA 31             INS                 ;RESTORE STACK POINTER
F2AB 31             INS
F2AC 860D           LDA A   #0DH
F2AE BDF145         JSR     TRANS       ;OUTPUT CARRIAGE RETURN
F2B1 BDF1FF A7      JSR     BREAK2      ;OUTPUT ADDRESS
F2B4 8620           LDA A   #20H        ;OUTPUT SPACE
F2B6 BDF145         JSR     TRANS
F2B9 A600           LDA A   0,X         ;LOAD WITH DATA
F2BB BDF1D8         JSR     BREAK       ;OUTPUT DATA
F2BE 8620           LDA A   #" "
F2C0 BDF145         JSR     TRANS       ;OUTPUT SPACE
F2C3 BDF11C A6      JSR     INPUT       ;INPUT CHAR.
F2C6 8143           CMP A   #"C"
F2C8 271A           BEQ     CHANGE      ;CHANGE COMMAND?
F2CA 810D           CMP A   #0DH
F2CC 2713           BEQ     NEXT        ;NEXT COMMAND?
F2CE 8142           CMP A   #"B"        ;B- FOR BACK CMD
F2D0 2707           BEQ     BACK        ;BACK COMMAND?
F2D2 8120           CMP A   #" "
F2D4 26ED           BNE     A6          ;RETURN TO MAIN PROG
F2D6 7EF01C         JMP     START

            ;BACK COMMAND

F2D9 09     BACK    DEX                 ;OUTPUT DATA CORRESPONDING TO
F2DA 860D           LDA A   #0DH        ;PREVIOUS ADDRESS
F2DC BDF145         JSR     TRANS
F2DF 20D0           BRA     A7
```

```
            ;NEXT COMMAND

F2E1 08     NEXT    INX             ;OUTPUT DATA CORRESPONDING TO
F2E2 20CD           BRA     A7      ;NEXT ADDRESS

  ;CHANGE COMMAND

F2E4 BDF182 CHANGE  JSR     PACK    ;REPLACE OLD DATA WITH
F2E7 A700           STA A   0,X     ;NEW DATA
F2E9 20D8           BRA     A6


                    MACRO   NUMBER
                    STX     @TEMP   ;STORE BEGIN. ADDR.
                    LDA A   @TEMP   ;LOAD PAGE SEL REG.
                    STA A   PGSEL   ;WITH PAGE NUMBER
                    LDA A   @TEMP+1 ;TO BE ACCESSED
                    STA A   PGSEL+1
                    ENDM


           ;SINGLE PAGE READ
           ;INDEX REG. CONTAINS STARTING PAGE NO.

F2EB 36     SREAD   PSH A           ;SAVE REGISTERS
F2EC 37             PSH B
                    NUMBER
F2ED DF62           STX     @TEMP   ;STORE BEGIN. ADDR.
F2EF 9662           LDA A   @TEMP   ;LOAD PAGE SEL REG.
F2F1 9780           STA A   PGSEL   ;WITH PAGE NUMBER
F2F3 9663           LDA A   @TEMP+1 ;TO BE ACCESSED
F2F5 9781           STA A   PGSEL+1
F2F7 8682           LDA A   #RDCMD  ;ISSUE READ COMMAND
F2F9 9783           STA A   CONT
F2FB 8D12           BSR     STAT    ;CHECK IF CONT. BUSY
F2FD C612           LDA B   #12H    ;SET UP BYTE COUNTER
F2FF CE0021         LDX     #BUF
F302 9682   A14     LDA A   RE      ;READ FIFO
F304 A700           STA A   0,X     ;STORE DATA INTO BUFFER
F306 08             INX
F307 5A             DEC B
F308 26F8           BNE     A14     ;PAGE TRANSFERED?
F30A DE62           LDX     @TEMP   ;RESTORE REGISTERS
F30C 33             PUL B
F30D 32             PUL A
F30E 39             RTS

           ;CHECK FOR CONT. BUSY

F30F 9684   STAT    LDA A   STAS    ;LOAD STATUS REG.
F311 8520           BIT A   #20H    ;CHECK IF CONT. NOT BUSY
F313 27FA           BEQ     STAT
```

```
F315 9684   A16    LDA A  STAS
F317 8520          BIT A  #20H      ;CHECK IF CONT. BUSY
F319 26FA          BNE    A16
F31B 39            RTS

           ;SINGLE PAGE WRITE
           ;INDEX REG. CONTAINS STARTING PAGE NO.

F31C 36    SWRITE  PSH A
                   NUMBER
F31D DF62          STX    @TEMP     ;STORE BEGIN. ADDR.
F31F 9662          LDA A  @TEMP     ;LOAD PAGE SEL REG.
F321 9780          STA A  PGSEL     ;WITH PAGE NUMBER
F323 9663          LDA A  @TEMP+1   ;TO BE ACCESSED
F325 9781          STA A  PGSEL+1
F327 37            PSH B
F328 C612          LDA B  #12H      ;SET UP BYTE COUNTER
F32A CE0021        LDX    #BUF      ;POINT TO BEG. OF PAGE BUF.
F32D A600   A18    LDA A  0,X       ;GET DATA FROM BUFFER
F32F 9785          STA A  WR        ;SAVE DATA INTO FIFO
F331 08            INX
F332 5A            DEC B            ;DECREMENT BYTE COUNTER
F333 26F8          BNE    A18       ;18 BYTES TRANSFERED?
F335 8684          LDA A  #WRCMD    ;ISSUE WRITE COMMAND
F337 9783          STA A  CONT
F339 8DD4          BSR    STAT      ;CHECK CONTROLLER STATUS
F33B DE62          LDX    @TEMP     ;RESTORE REGISTERS
F33D 33            PUL B
F33E 32            PUL A
F33F 39            RTS

           ;GET START AND END ADDRESSES FOR BUBBLE MEMORY
           ;TRANSACTIONS.

F340 01    INPUT2  NOP
F341 BDF0D0 B1     JSR    AB        ;CHECK IF ASCII OR BIN.
F344 CEFC63 B29    LDX    #M20      ;OUTPUT MESSAGE
F347 BDF1BE        JSR    MESSA
F34A BDF1EF        JSR    ADDR      ;GET START ADDR.
F34D DF60          STX    @TEMP4    ;STORE START ADDR.
F34F 965A          LDA A  @TEMP1    ;CHECK IF ILLEGAL ADDR.,THAT IS,
F351 8102          CMP A  #02H      ;IS THE ADDRESS BETWEEN 0000
F353 2206          BHI    B30       ;AND 0280 HEX.
F355 965B          LDA A  @TEMP1+1
F357 8180          CMP A  #80H
F359 2308          BLS    B28       ;END OF CHECK
F35B CEFBBA B30    LDX    #M10
F35E BDF1BE        JSR    MESSA     ;OUTPUT MESSAGE - ILLEGAL ADDRESS
F361 20E1          BRA    B29
F363 CEFC6F B28    LDX    #M21      ;OUTPUT NEXT MESSAGE
F366 BDF1BE        JSR    MESSA
F369 BDF1EF        JSR    ADDR      ;GET END ADDR.
F36C 965A          LDA A  @TEMP1    ;CHECK IF ILLEGAL ADDR.
```

```
F36E 8101          CMP A  #01H      ;THIS PORTION AGAIN CHECKS
F370 2312          BLS    B31       ;TO SEE IF THE END ADDRESS
F372 8102          CMP A  #02H      ;IS VALID
F374 2206          BHI    B32
F376 965B          LDA A  @TEMP1+1
F378 8180          CMP A  #80H
F37A 2308          BLS    B31
F37C CEFBBA B32    LDX    #M10
F37F BDF1BE        JSR    MESSA     ;OUTPUT MESSAGE - ILLEGAL ADDRESS
F382 20DF          BRA    B28
            ;THIS ROUTINE INPUTS DATA FROM THE TERMINAL
            ;IN EITHER ASCII OR HEX DEPENDING ON
            ;INITIAL SET UP CONDITONS.
            ;THEN THE ROUTINE FILLS A BUFFER WITH THAT DATA.

F385 CEFB9D INPUT3 LDX    #M7
F388 BDF1BE        JSR    MESSA     ;OUTPUT MESSAGE - DATA PATTERN
F38B D659          LDA B  @LOC
F38D C142          CMP B  #"B"      ;CHECK IF HEX OR BINARY NEEDED
F38F 2605          BNE    B3
F396 BDF11C B3     JSR    INPUT     ;GET ASCII DATA
F399 9764   OVER   STA A  @LOC1     ;STORE IN @TEMPREG
F39B CE0021 OVER1  LDX    #BUF      ;FILL PAGE BUFFER WITH DATA
F39E A700   B4     STA A  0,X
F3A0 08            INX
F3A1 8C0033        CPX    #ENDBUF
F3A4 26F8          BNE    B4        ;IS BUFFER FULL?
F3A6 39            RTS

            ;CONTINUOUS READ/WRITES TO BUBBLE MODULE
F3A7  CEFB6D CNTNUE LDX    #M4
F3AA BDF1C4        JSR    MESS      ;OUTPUT REST OF PROMPT
F3AD 8D91          BSR    INPUT2    ;GET START AND END ADDRESS
F3AF 8DD4          BSR    INPUT3    ;FILL BUFFER WITH DATA
F3B1 DE5A          LDX    @TEMP1    ;GET START ADDRESS
F3B3 08            INX
F3B4 DF5A          STX    @TEMP1    ;STORE NEXT ADDRESS
F3B6 CE0000        LDX    #0        ;INIT. ERROR CNTR TO ZERO
F3B9 DF67          STX    @TEMP6
F3BB DF69          STX    @TEMP7    ;INIT. R/W OPER. CNTR
F3BF BDF436        JSR    B10B      ;OUTPUT MESSAGE - **WORKING**
F3C2 DE6B   B7     LDX    @TEMP8    ;INCREMENT OPERATION COUNTER
F3C4 08            INX              ;LEAST SIGNIFICANT TWO BYTES
F3C5 DF6B          STX    @TEMP8    ;SAVE COUNTER
F3C7 2609          BNE    DOWN1
F3C9 DE69          LDX    @TEMP7    ;INCREMENT OPERATION COUNTER
F3CB 08            INX              ;MOST SIGNIFICANT TWO BYTES
```

```
F3CC DF69          STX    @TEMP7   ;SAVE COUNTER
F3CE 9664          LDA A  @LOC1    ;RETRIEVE DATA PATTERN
F3D0 8DC9          BSR    OVER1    ;FILL BUFFER WITH DATA PATTERN
F3D2 DE60   DOWN1  LDX    @TEMP4   ;WRITE  N  PAGES OF DATA
F3D4 BDF31C B6     JSR    SWRITE   ;CHECKING TO INSURE END
F3D7 8D15          BSR    IO       ;HAS NOT BEEN REACHED
F3D9 08            INX
F3DA 9C5A          CPX    @TEMP1   ;IS END REACHED?
F3DC 26F6          BNE    B6       ;IF NOT,WRITE OUT PATTERN
F3DE DE60   B5     LDX    @TEMP4   ;READ  N  PAGES OF DATA
F3E0 BDF2EB B8     JSR    SREAD    ;CHECKING TO INSURE END
F3E3 8D09          BSR    IO       ;HAS NOT BEEN REACHED
F3E5 8D2B          BSR    VER      ;PRINT OUT NUMBER OF CYCLES RUN
F3E7 08            INX
F3E8 9C5A          CPX    @TEMP1   ;IS END REACHED?
F3EA 26F4          BNE    B8       ;IF NOT CONTINUE READING
F3EC 20D4          BRA    B7       ;REPEAT PROCEDURE

           ;CHECK FOR RESTART OR VERIFICATION KEY CLOSURE

F3EE 36     IO     PSH A           ;SAVE REGISTERS
F3EF 37            PSH B
F3F0 DF65          STX    @TEMP5
F3F2 B60200        LDA A  ACIACT   ;CHECK FOR DATA IN ACIA
F3F5 47            ASR A
F3F6 2410          BCC    V1       ;EXIT IF NO DATA
F3F8 B60201        LDA A  ACIATR   ;LOAD CHARACTER FROM ACIA
F3FB 811B          CMP A  #1BH     ;CHECK FOR ESC CHAR.
F3FD 270E          BEQ    B9       ;BRANCH IF AN ESC CHAR
F3FF 8156          CMP A  #"V"     ;CHECK FOR A V ENTRY
F401 2605          BNE    V1       ;BRANCH IF NOT FOUND
F403 8D3A          BSR    VERFY    ;PRINT OUT ERROR COUNT
F405 BDF436        JSR    B10B     ;PRINT **WORKING**
F408 DE65   V1     LDX    @TEMP5   ;RESTORE REGISTERS
F40A 33            PUL B
F40B 32            PUL A
F40C 39            RTS
F
F40D 31     B9     INS             ;RESTORE STACK POINTER
F40E 31            INS
F40F 7EF01C        JMP    START    ;RESTART MONITOR

           ;VERIFY BUFFER WITH CHARACTER

F412 DF5C   VER    STX    @TEMP2   ;STORE CURRENT PAGE POSN.
F414 CE0021        LDX    #BUF     ;POINT TO BEG. OF PAGE BUFFER
F417 9664          LDA A  @LOC1    ;LOAD ACTUAL DATA
F419 A100   B11    CMP A  0,X      ;COMPARE CHAR. WITH BUF.
F41B 2609          BNE    B10      ;BRANCH IF ERROR
F41D 08            INX             ;POIUNT TO NEXT CHAR. IN BUFFER
F41E 8C0033        CPX    #ENDBUF   ;HAS END BEEN REACHED?
F421 26F6          BNE    B11      ;CONTINUE CHECKING IF NOT COMPLETE
F423 DE5C          LDX    @TEMP2   ;RESTORE PAGE POSN.
```

```
F425 39             RTS

             ;OUTPUT FOLLOWING MESSAGE - **WORKING**

F426 CEFBAB B10     LDX    #M8       ;POINT TO MESSAGE
F429 BDF1BE         JSR    MESSA     ;OUTPUT MESSAGE  - READ ERROR
F42C DE5C           LDX    @TEMP2    ;GET CURRENT PAGE POSN.
F42E BDF525         JSR    OUTPUT    ;OUTPUT PAGE OF DATA
F431 DE67    B10A   LDX    @TEMP6    ;INC. ERROR CNTR
F433 08             INX
F434 DF67           STX    @TEMP6    ;SAVE ERROR COUNTER
F436 CEFCDB B10B    LDX    #M29      ;OUTPUT MESSAGE - **WORKING**
F439 BDF1BE         JSR    MESSA
F43C DE5C           LDX    @TEMP2    ;RESTORE PAGE POSN INDICATOR
F43E 39             RTS

             ;OUTPUT NUMBER OF CYCLES RUN & NO. OF ERRORS EMCOUNTERED

F43F CEFD42 VERFY   LDX    #M39      ;MESSAGE - NO.  OF HARD ERRRORS
F442 BDF1BE         JSR    MESSA     ;OUTPUT MESSAGE
F445 9673           LDA A  @LOC2     ;LOAD ERROR COUNT
F447 BDF1D8         JSR    BREAK     ;PRINT ERROR COUNT
F44A CEFCC7         LDX    #M30      ;MESAGE - NO. OF SOFT ERRORS
F44D BDF1BE         JSR    MESSA     ;OUTPUT MESSAGE
F450 DE67           LDX    @TEMP6    ;LOAD ERROR COUNT
F452 BDF1FF         JSR    BREAK2    ;PRINT ERROR COUNT
F455 CEFCE9         LDX    #M31      ;MESSAGE - NO. OF R/W OPERATIONS
F458 BDF1BE         JSR    MESSA     ;OUTPUT MESSAGE
F45B DE69           LDX    @TEMP7    ;LOAD COUNTER
F45D BDF1FF         JSR    BREAK2    ;OUTPUT COUNTER
F460 DE6B           LDX    @TEMP8    ;LOAD COUNTER  LS. BYTES
F462 BDF1FF         JSR    BREAK2    ;OUTPUT COUNT
F465 39             RTS

             ;READ PAGE FROM BUBBLE MODULE

F466 CEFBB6 READ    LDX    #M9       ;OUTPUT REST OF PROMPT
F469 BDF1C4         JSR    MESS
F46C BDF0D0 B26     JSR    AB        ;CHECK IF ASCII OF BIN.
F46F CEFC63 B13     LDX    #M20      ;MESSAGE - START PAGE:
F472 BDF1BE         JSR    MESSA
F475 BDF1EF         JSR    ADDR      ;GET START ADDRESS
F478 965A           LDA A  @TEMP1    ;CHECK IF ILLEGAL ADDR.
F47A 8101           CMP A  #01H
F47C 2312           BLS    AROUND
F47E 8102           CMP A  #02H
F480 2206           BHI    B33
F482 965B           LDA A  @TEMP1+1
F484 8180           CMP A  #80H
F486 2308           BLS    AROUND    ;END OF CHECK
F488 CEFBBA B33     LDX    #M10      ;MESSAGE - ILLEGAL PAGE
F48B BDF1BE         JSR    MESSA     ;OUTPUT MESSAGE
F48E 20DF           BRA    B13       ;GET NEW PAGE ADDRESS
```

```
F490 BDF2EB AROUND JSR    SREAD     ;READ PAGE
F493 DF5C          STX    @TEMP2    ;SAVE PAGE CURRENT PAGE POSN.
F495 BDF525        JSR    OUTPUT    ;OUTPUT PAGE ADDRESS
F498 CEFBCA B18    LDX    #M11      ;CHANGE OUTPUT PROMPT - ==>
F49B BDF1BE        JSR    MESSA     ;OUTPUT PROMPT
F49E BDF11C B15    JSR    INPUT     ;GET TERMINAL ENTRY
F4A1 814E          CMP A  #"N"      ;CHECK IF NEXT PG. REQ.
F4A3 2711          BEQ    NEXT1     ;BRANCH IF NEXT APGE REQUIRED
F4A5 8142          CMP A  #"B"      ;CHECK IF PREV. PG. REQ.
F4A7 271C          BEQ    BACK1     ;BRACH IF PREV. PAGE REQUIRED
F4A9 8143          CMP A  #"C"      ;CHECK IF CHANGE REQUESTED
F4AB 2604          BNE    N5        ;GO TO COMMAND REJECT
F4AD 8D24          BSR    CHNGE1    ;GO TO CHANGE ROUTINE
F4AF 20DF          BRA    AROUND    ;REPEAT PROCEDURE
F4B1 BDF087 N5     JSR    REJECT    ;COMMAND REJECT
F4B4 20E8          BRA    B15       ;GET NEW INPUT & REPEAT
F4B6 DE5C   NEXT1  LDX    @TEMP2    ;OUTPUT NEXT PAGE
F4B8 08            INX               ;UNLESS PAGE 0281H IS
F4B9 8C0281        CPX    #ENDMEM   ;REACHED,THEN OUPUT PAGE 0
F4BC 26D2          BNE    AROUND    ;OUTPUT DESIRED PAGE
F4BE CE0000        LDX    #0        ;POINT TO PAGE ZERO
F4C1 DF5C          STX    @TEMP2    ;SAVE PAGE POSN. ADDR.
F4C3 20CB          BRA    AROUND    ;OUTPUT PAGE
F4C5 DE5C   BACK1  LDX    @TEMP2    ;OUTPUT PREV. PAGE
F4C7 2703          BEQ    B25
F4C9 09            DEX               ;IF PAGE 0 IS REACHED
F4CA 20C4          BRA    AROUND    ;OUTPUT PAGE 0280H
F4CC CE0280 B25    LDX    #ENDMEM-1 ;GET LAST PAGE ADDR.
F4CF DF5C          STX    @TEMP2    ;SAVE PAGE ADDR.
F4D1 20BD          BRA    AROUND    ;OUTPUT PAGE & REPEAT

           ;CHANGE CONTENTS OF PAGE

F4D3 8D6E   CHNGE1 BSR    SAME1     ;OUTPUT CURRENT PAGE OF DATA &
F4D5 271E          BEQ    B21       ;CHECK FO ASCII OR HEX OUTPUT
           ;ASCII SECTION
F4D7 BDF11C B23    JSR    INPUT     ;GET TERMINAL INPUT
F4DA 815E          CMP A  #5EH      ;CHECK FOR UP ARROW
F4DC 270A          BEQ    B22       ;IF FOUND, MAINTAIN DATA
F4DE A700          STA A  0,X       ;OTHERWISE STORE DATA INTO BUFFER
F4E0 08            INX               ;POINT TO NEXT CHAR. IN BUFFER
F4E1 8C0033        CPX    #ENDBUF    ;IS END REACHED?
F4E4 26F1          BNE    B23       ;CONTINUE OUTPUT
F4E6 2028          BRA    B27       ;WRITE CHANGED BUFFER
F4E8 8608   B22    LDA A  #08H      ;LOAD WITH BACK-SPACE CHAR
F4EA BDF145        JSR    TRANS     ;OUTPUT BS CHAR.
F4ED A600          LDA A  0,X       ;LOAD PRESENT DATA
F4EF 8D2C          BSR    SAME3     ;OUTPUT DATA
F4F1 26E4          BNE    B23       ;CHECK IF NEW INPUT
F4F3 201B          BRA    B27       ;WRITE CHANGED BUFFER
           ;HEX SECTION
F4F5 BDF11C B21    JSR    INPUT     ;GET TERMINAL INPUT
F4F8 815E          CMP A  #5EH      ;CHECK FOR UP ARROW
```

```
F4FA 270B          BEQ    B24       ;MAINTAIN DATA
F4FC BDF185        JSR    PACK1     ;GET BYTE OF DATA
F4FF A700          STA A  0,X       ;STORE DATA INTO BUF
F501 8D18          BSR    SAME4     ;OUTPUT SPACE & CHECK FOR BUF. END
F503 26F0          BNE    B21       ;REPEAT IF NO THE END OF BUFFER
F505 2009          BRA    B27       ;WRITE CHANGED BUFFER
F507 8608   B24    LDA A  #08H      ;LOAD WITH BACK-SPACE CHAR.
F509 BDF145        JSR    TRANS     ;TRANS BS CHAR.
F50C 8D08          BSR    SAME      ;OUTPUT BUFFER DATA
F50E 26E5          BNE    B21       ;GET NEW INPUT
F510 DE5C   B27    LDX    @TEMP2    ;SAVE PAGE POSN INDICATOR
F512 BDF31C        JSR    SWRITE    ;WRITE OUT CHANGED PAGE
F515 39            RTS

           ;OUTPUT PAGE

F516 A600   SAME   LDA A  0,X       ;READ DATA FROM BUF.
F518 BDF1D8 SAME2  JSR    BREAK     ;OUTPUT CHAR IF BIN.
F51B 8620   SAME4  LDA A  #" "      ;OUTPUT SPACE
F51D BDF145 SAME3  JSR    TRANS
F520 08            INX
F521 8C0033        CPX    #ENDBUF   ;IS END OF BUFFER REACHED
F524 39            RTS

           ;OUTPUT PAGE BUFFER,X CONTAINS PAGE ADDR.

F525 DF5C   OUTPUT STX    @TEMP2    ;SAVE PAGE POSN. INDICATOR
F527 36            PSH A            ;SAVE REGISTERS
F528 37            PSH B
F529 8D18          BSR    SAME1     ;OUTPUT CURRENT PAGE POSN DATA & CHECK
F52B 270D          BEQ    B19       ;FOR ASCII OR HEX INPUT
           ;ASCII SECTION
F52D A600   B20    LDA A  0,X       ;GET DATA
F52F BDF145        JSR    TRANS     ;OUTPUT DATA
F532 08            INX              ;POINT TO NEXT CHAR.
F533 8C0033        CPX    #ENDBUF   ;IS END OF BUF. REACHED
F536 26F5          BNE    B20       ;CONTINUE IF NOT FINISHED
F538 2004          BRA    B35       ;EXIT
           ;HEX SECTION
F53A 8DDA   B19    BSR    SAME      ;OUTPUT HEX DATA
F53C 26FC          BNE    B19       ;CONTINUE UNTIL FINISHED
F53E DE5C   B35    LDX    @TEMP2    ;RESTORE PAGE POSN AND REGISTERS
F540 33            PUL B
F541 32            PUL A
F542 39            RTS

           ;OUTPUT CURRENT PAGE OF DATA & CHECK IF ASCII OR
           ;HEX FORMAT DESIRED

F543 CEFBCE SAME1  LDX    #M12      ;OUTPUT APPROPRIATE MESSAGE
F546 BDF1BE        JSR    MESSA
F549 965C          LDA A  @TEMP2    ;GET CURRENT PAGE POSN
F54B BDF1D8        JSR    BREAK     ;OUTPUT PAGE ADDRESS - MS BYTE
```

```
F54E 965D          LDA A  @TEMP2+1 ;GET CURRENT PAGE POSN.
F550 BDF1D8        JSR    BREAK    ;OUTPUT LS BYTE
F553 8620          LDA A  #" "     ;OUTPUT SPACE
F555 BDF145        JSR    TRANS
F558 CE0021        LDX    #BUF     ;LOAD X WITH BEG. OF BUFFER
F55B 9659          LDA A  @LOC     ;LOAD ASCII OR HEX INDICATOR
F55D 8142          CMP A  #"B"     ;CHECK IF ASCII OR HEX
F55F 39            RTS

            ;FILL BLOCK OF MEMORY WITH CERTAIN DATA PATTERN

F560 CEFBD4 FILL   LDX    #M13     ;OUTPUT PROMPT
F563 BDF1C4        JSR    MESS
F566 BDF340        JSR    INPUT2   ;GET START & END ADDRESSES
F569 BDF385        JSR    INPUT3   ;GET NECESSARY DATA
F56C DE5A          LDX    @TEMP1   ;GET CURRENT PAGE POSN.
F56E 08            INX             ;INCREMENT PAGE POSN
F56F DF5A          STX    @TEMP1   ;SAVE PAGE POSN
F571 CEFCDB        LDX    #M29     ;MESSAGE - **WORKING**
F574 BDF1BE        JSR    MESSA    ;OUTPUT MESSAGE
F577 DE60          LDX    @TEMP4   ;GET START PAGE ADDRESS
F579 BDF31C E6     JSR    SWRITE   ;WRITE OUT PAGE OF DATA
F57C 08            INX             ;INCREMENT PAGE POSN.
F57D 8C0281        CPX    #ENDMEM  ;CHECK IF END IS REACHED
F580 2704          BEQ    E5       ;EXIT
F582 9C5A          CPX    @TEMP1   ;CHECK IF BLOCK END IS REACHED
F584 26F3          BNE    E6       ;CONTINUE IF NOT FINISHED
F586 CEFBD8 E5     LDX    #M14     ;MESSAGE - FILLED
F589 BDF1BE        JSR    MESSA    ;OUTPUT MESSAGE
F58C 7EF01C        JMP    START    ;RESTART

            ;LIST BLOCK OF MEMORY TO TERMINAL

F58F CEFBE6 LIST1  LDX    #M15     ;OUTPUT PROMPT
F592 BDF1C4        JSR    MESS
F595 BDF340        JSR    INPUT2   ;GET I/O INFO
F598 DE5A          LDX    @TEMP1   ;GET PAGE CURRENT PAGE POSN.
F59A 08            INX             ;INCREMENT PAGE POSN
F59B DF5A          STX    @TEMP1   ;SAVE PAGE POSN
F59D DE60          LDX    @TEMP4   ;GET CURRENT PAGE POSN
F59F BDF2EB E8     JSR    SREAD    ;READ PAGE OF DATA
F5A2 DF5C          STX    @TEMP2   ;STORE PAGE POSN.
F5A4 BDF525        JSR    OUTPUT   ;OUTPUT PAGE OF DATA
F5A7 DE5C          LDX    @TEMP2   ;GET PAGE POSN
F5A9 08            INX             ;INCREMENT PAGE POSN
F5AA 8C0281        CPX    #ENDMEM  ;IS END REACHED?
F5AD 2704          BEQ    E7       ;EXIT IF YES
F5AF 9C5A          CPX    @TEMP1   ;CHECK IF BLOCK END REACHED
F5B1 26EC          BNE    E8       ;CONTINUE IF NO FINISHED
F5B3 7EF01C E7     JMP    START

            ;TRANSFER CONTENTS OF BUBBLE TO RAM
            ;OR RAM TO BUBBLE
```

```
F5B6 CEFBEA TRSFER LDX     #M16     ;OUTPUT REST OF PROMPT
F5B9 BDF1C4        JSR     MESS
F5BC CEFBF2 F1     LDX     #M17     ;OUTPUT APPROPRIATE MESSAGE
F5BF BDF1BE        JSR     MESSA
F5C2 BDF11C F5     JSR     INPUT    ;CHECK FOR R OR B
F5C5 8142          CMP A   #"B"     ;TRANSFER FROM RAM TO BUBBLE
F5C7 270C          BEQ     F3
F5C9 8152          CMP A   #"R"     ;TRANSFER FROM BUBBLE TO RAM
F5CB 2603          BNE     F24      ;REJECT COMMAND
F5CD 7EF63B        JMP    F4
F5D0 BDF087 F24    JSR     REJECT   ;COMMAND REJECT
F5D3 20ED          BRA     F5       ;GET NEW INPUT
           ;RAM TO BUBBLE TRANSFER
F5D5 CEFB81 F3     LDX     #M5      ;RAM TO BUBBLE TRANSFER
F5D8 BDF1BE        JSR     MESSA    ;OUTPUT MESSAGE
F5DB BDF1EF        JSR     ADDR     ;GET STARTING ADDR
F5DE DF60          STX     @TEMP4   ;STORE ADDRESS
F5E0 CEFB90        LDX     #M6      ;MESSAGE - END ADDRESS:
F5E3 BDF1BE        JSR     MESSA
F5E6 BDF1EF        JSR     ADDR     ;GET END ADDR.
F5E9 DF5E          STX     @TEMP3   ;SAVE END ADDRESS
F5EB CEFC3F        LDX     #M18     ;MESSAGE - TRANSFER PAGE
F5EE BDF1BE        JSR     MESSA
F5F1 BDF1EF        JSR     ADDR     ;GET TRANSFER PAGE ADDR.
F5F4 9F65          STS     @TEMP5   ;STORE STACK POINTER
F5F6 BDF436        JSR     B10B     ;MESSAGE - **WORKING**
F5F9 9E60   F7     LDS     @TEMP4   ;POINT TO FIRST DATA BYTE
F5FB 34            DES
F5FC CE0021        LDX     #BUF     ;POINT TO BEG. OF BUFFER
F5FF 32     F6     PUL A            ;GET DATA BYTE
F600 A700          STA A   0,X      ;FILL PAGE BUFFER
F602 DF60          STX     @TEMP4   ;SAVE STACK POINTER
F604 30            TSX              ;GET CURRENT DATA ADDR.
F605 09            DEX              ;DEC DATA POINTER
F606 9C5E          CPX     @TEMP3   ;IS END REACHED?
F608 2708          BEQ     F21      ;WRITE OUT BUFFER IF YES
F60A DE60          LDX     @TEMP4   ;RESTORE PAGE POSN. INDICATOR
F60C 08            INX              ;POINT TO NEXT PAGE
F60D 8C0033        CPX     #ENDBUF  ;BUFFER FULL?
F610 26ED          BNE     F6       ;FILL IF NO
F612 9F60   F21    STS     @TEMP4   ;SAVE DATA POINTER
F614 9E65          LDS     @TEMP5   ;RESTORE STACK POINTER
F616 DE5A          LDX     @TEMP1   ;POINT TO CURRENT PAGE POSN
F618 BDF31C        JSR     SWRITE   ;ISSUE SINGLE PAGE WRITE
F61B 08            INX
F61C DF5A          STX     @TEMP1   ;SET UP FOR NEXT PAGE
F61E 8C0281        CPX     #ENDMEM  ;CHECK IF END OF MEM.
F621 270F          BEQ     F9       ;EXIT IF FINISHED
F623 DE60          LDX     @TEMP4   ;GET CURRENT RAM ADDR.
F625 9C5E          CPX     @TEMP3   ;CHECK IF LAST RAM ADDR.
F627 26D0          BNE     F7       ;REPEAT IF NOT FINISHED
F629 CEFC51        LDX     #M19     ;MESS. - TRANSFER COMPLETE
```

```
F62C BDF1BE        JSR     MESSA   ;OUTPUT MESSAGE
F62F 7EF01C        JMP     START   ;RESTART
F632 CEFBD8 F9     LDX     #M14    ;MESSAGE - MEMORY FILLED
F635 BDF1BE        JSR     MESSA   ;OUTPUT MESSAGE
F638 7EF01C        JMP     START   ;RETURN TO MAIN PROG.
F63B CEFC63 F4     LDX     #M20    ;BUBBLE TO RAM TRANSFER
F63E BDF1BE        JSR     MESSA   ;OUTPUT MESSAGE
F641 BDF1EF        JSR     ADDR    ;GET STARTING PAGE
F644 965A          LDA A   @TEMP1  ;CHECK FOR VALID PAGE
F646 8101          CMP A   #01H
F648 2312          BLS     C2
F64A 8102          CMP A   #02     ;A VALID PAGE IS BETWEEN
F64C 2206          BHI     C1      ;0000 AND 0280 HEX
F64E 965B          LDA A   @TEMP1+1
F650 8180          CMP A   #80H
F652 2308          BLS     C2      ;END OF CHECK
F654 CEFBBA C1     LDX     #M10    ;MESSAGE - ILLEGAL ADDRESS
F657 BDF1BE        JSR     MESSA   ;OUTPUT MESSAGE
F65A 20DF          BRA     F4      ;GET NEW ADDRESS IF NOT VALID
F65C DF60   C2     STX     @TEMP4  ;SAVE PAGE POSN.
F65E CEFC6F C3     LDX     #M21    ;MESSAGE - END PAGE:
F661 BDF1BE        JSR     MESSA   ;OUTPUT MESSAGE
F664 BDF1EF        JSR     ADDR    ;GET ENDING PAGE
F667 965A          LDA A   @TEMP1  ;CHECK FOR VALID PAGE
F669 8101          CMP A   #01H
F66B 2312          BLS     C5
F66D 8102          CMP A   #02     ;A VALID PAGE IS BETWEEN
F66F 2206          BHI     C4      ;0000 AND 0280 HEX
F671 965B          LDA A   @TEMP1+1
F673 8180          CMP A   #80H
F675 2308          BLS     C5      ;END OF CHECK
F677 CEFBBA C4     LDX     #M10    ;MESSAGE - ILLEGAL ADDRESS
F67A BDF1BE        JSR     MESSA   ;OUTPUT MESSAGE
F67D 20DF          BRA     C3      ;GET NEW INPUT IF NOT VALID
F67F DF5E   C5     STX     @TEMP3  ;SAVE PAGE POSN.
F681 CEFC79 C7     LDX     #M22    ;MESS. - TRANSFER RAM ADDRESS
F684 BDF1BE        JSR     MESSA   ;OUTPUT MESSAGE
F687 BDF1EF        JSR     ADDR    ;GET TRANSFER RAM ADDRESS
F68A 965A          LDA A   @TEMP1  ;CHECK IF MEMORY IS NOT
F68C 8103          CMP A   #03     ;SYSTEM STACK AREA
F68E 2208          BHI     C6      ;EXIT IF NOT
F690 CEFBBA        LDX     #M10    ;MESSAGE - ILLEGAL ADDRESS
F693 BDF1BE        JSR     MESSA   ;OUTPUT MESSAGE
F696 20E9          BRA     C7      ;GET NEW ADDRESS
F698 CEFCDB C6     LDX     #M29    ;MESSAGE - **WORKING**
F69B BDF1BE        JSR     MESSA   ;OUTPUT MESSAGE
F69E DE60          LDX     @TEMP4  ;LOAD START PAGE
F6A0 BDF2EB F11    JSR     SREAD   ;GET PAGE OF DATA
F6A3 9F65          STS     @TEMP5  ;SAVE STACK POINTER
            ;THIS SECTION TRANSFERS THE CONTENTS OF
            ;THE PAGE BUFFER TO THE RAM
S6A5 9E5A          LDS     @TEMP1  ;LOAD WITH TRANSFER ADDRESS
F6A7 CE0021        LDX     #BUF    ;POINT TO BEG. OF BUFFER
```

```
F6AA A600   F10    LDA A   0,X      ;LOAD DATA
F6AC 36            PSH A           ;TRANSFER DATA
F6AD 31            INS             ;FIX UP TRANSFER ADDRESS
F6AE 31            INS
F6AF 08            INX             ;POINT OT NEXT DATA
F6B0 8C0033        CPX     #ENDBUF ;TRANSFER COMPLETE?
F6B3 26F5          BNE     F10     ;CONTINUE IF NOT
F6B5 9F5A          STS     @TEMP1  ;SAVE CURRENT TRANSFER ADDR.
F6B7 DE60          LDX     @TEMP4  ;LOAD CURRENT PAGE POSN
F6B9 08            INX             ;NEXT PAGE
F6BA DF60          STX     @TEMP4  ;SAVE PAGE POSN
F6BC 9E65          LDS     @TEMP5  ;RESTORE STACK POINTER
F6BE 9C5E          CPX     @TEMP3  ;FINAL PAGE?
F6C0 26DE          BNE     F11     ;CONTINUE IF NOT
F6C2 CEFC51        LDX     #M19    ;MESS. - TRANSFER COMPLETE
F6C5 BDF1BE        JSR     MESSA   ;OUTPUT MESSAGE
F6C8 7EF01C        JMP     START   ;RESTART

           ;ALLOW FOR CONTROLLER INIT.

F6CB CEFC8F INTLZE LDX     #M23    ;POINT TO PROMPT
F6CE BDF1C4        JSR     MESS    ;OUTPUT REST OF PROMPT
F6D1 BDF0F5 F17    JSR     BUBINT  ;INITIALIZE CONTROLLER
F6D4 7EF01C        JMP     START   ;RESTART

           ;DATA INPUT FROM COMPUTER (HALF DUPLEX)

F6D7 CEFC9A START1 LDX     #M24    ;MESS. - DATA INPUT MODE
F6DA BDF1C4        JSR     MESS    ;OUTPUT MESSAGE
F6DD CE0000        LDX     #0      ;INIT PAGE POSN COUNTER
F6E0 DF5A          STX     @TEMP1  ;SAVE PAGE POSN
F6E2 B60200 F12    LDA A   ACIACT  ;CHECK FOR START CHAR.
F6E5 8501          BIT A   #01H
F6E7 27F9          BEQ     F12     ;REPEAT IF NO DATA YET
F6E9 B60201        LDA A   ACIATR  ;LOAD DATA
F6EC 8157          CMP A   #"W"    ;IS IT A W
F6EE 26F2          BNE     F12     ;REPEAT IF NOT
F6F0 CE0021 F15    LDX     #BUF    ;FILL PAGE BUFFER WITH DATA
F6F3 B60200 F13    LDA A   ACIACT  ;CHECK FOR DATA
F6F6 8501          BIT A   #01H
F6F8 27F9          BEQ     F13     ;REPEAT IF NO DATA
F6FA B60201        LDA A   ACIATR  ;GET DATA
F6FD A700          STA A   0,X     ;STORE DATA IN PAGE BUFFER
F6FF 08            INX             ;POIN T TO NEXT POSN.
F700 812F          CMP A   #"/"    ;CHECK FOR END OF DATA
F702 2714          BEQ     F22     ;CONTINUE IF NOT THE END
F704 8C0033        CPX     #ENDBUF ;IS BUFFER FULL
F707 26EA          BNE     F13     ;CONTINUE IF NOT
F709 DE5A          LDX     @TEMP1  ;GET CURRENT PAGE POSN.
F70B BDF31C        JSR     SWRITE  ;WRITE CONTENTS OF BUFFER
F70E 08            INX             ;INC. PAGE POSN
F70F DF5A   F14    STX     @TEMP1  ;SAVE PAGE POSN.
F711 8C0281        CPX     #ENDMEM ;IS BUBBLE MEMORY FULL?
```

```
F714 2710          BEQ    F16             ;EXIT IF YES
F716 20D8          BRA    F15             ;ELSE CONTINUE
F718 DE5A   F22    LDX    @TEMP1          ;GET PAGE POSN
F71A BDF31C        JSR    SWRITE          ;WRITE OUT FINAL BUFFER
F71D CEFCAB        LDX    #M25            ;MESS - DATA TRANSFERRED
F720 BDF1BE        JSR    MESSA           ;OUTPUT MESSAGE
F723 7EF01C        JMP    START           ;RESTART
F726 CEFBD8 F16    LDX    #M14            ;MESSAGE - MEMORY FILLED
F729 BDF1BE        JSR    MESSA           ;OUTPUT MESSAGE
F72C 7EF01C        JMP    START           ;RESTART

            ;TEST 1        -GENERATOR AND DETECTOR TEST PATTERN
            ;              -PATTERN USED -F1150EEA00

F72F CEFD01 TEST1          LDX    #M32     ;MESSAGE - TEST1
F732 BDF1BE                JSR    MESSA    ;OUTPUT MESSAGE
F735 BDF436                JSR    B10B     ;OUTPUT - **WORKING**
F738 CEFB02 J3             LDX    #PAT1    ;POINT TO BEGINNING OF DATA PATTERN
F73B 8D12                  BSR    TEST     ;PERFORM TEST
F73D 20F9                  BRA    J3       ;REPEAT

            ;TEST 2        -MAJOR LOOP FAILURE MODE TESTING
            ;              -PATTERN USED - FF08800F77

F73F CEFD08 TEST2          LDX    #M33     ;MESSAGE TEST2
F742 BDF1BE                JSR    MESSA    ;OUTPUT MESSAGE
F745 BDF436                JSR    B10B     ;OUTPUT - **WORKING**
F748 CEFB07 J4             LDX    #PAT2    ;POINT TO BEGINNING OF DATA PATTERN
F74B 8D02                  BSR    TEST     ;PERFORM TEST
F74D 20F9                  BRA    J4       ;REPEAT

            ;THIS ROUTINE PERFORMS THE ACTUAL TESTING

F74F DF6D   TEST           STX    @TEMP9   ;SAVE INDEX REGISTER TO BE USED LATER
F751 8D4E                  BSR    TRANSFER ;FILL PAGE BUFFER WITH PATTERN
F753 CE0000                LDX    #0       ;POINT TO PAGE ZERO
F756 BDF31C                JSR    SWRITE   ;WRITE PAGE
F759 BDF2EB                JSR    SREAD    ;READ PAGE
F75C BDF7E4                JSR    COUNT    ;COUNT NO. OF OPERATIONS
F75F 8D1D                  BSR    COMPARE  ;COMPARE RESULTS
F761 2504                  BCS    BELOW    ;BRANCH ON ERROR CONDITION
F763 BDF3EE ABOVE4         JSR    IO       ;CHECK FOR ANY KEYBOARD IO
F766 39                    RTS
            ;THIS SECTION CALLED UPON ERROR CONDITION
F767 C606   BELOW          LDA B  #06      ;LOAD MAX. ERORR COUNT
F769 5A     U2             DEC B           ;DEC. COUNTER
F76A 270E                  BEQ    U3       ;JUMP TO HARD ERROR SECTION
F76C 8D71                  BSR    SOFTERR  ;UPDATE SOFT-ERROR COUNT
F76E CE0000                LDX    #0       ;LOAD WITH PAGE POSN.
F771 BDF2EB                JSR    SREAD    ;REREAD PAGE
F774 8D08                  BSR    COMPARE  ;COMPARE (CHECK FOR ERRORS)
F776 25F1                  BCS    U2       ;REPEAT IF ERRORS
F778 20E9                  BRA    ABOVE4   ;RETURN IF NO ERROR
```

```
F77A 8D42   U3              BSR   HARDERR  ;THIS SECTION EXECUTED IF FIVE
F77C 20E5                   BRA   ABOVE4   ;SOFT ERRORS ARE ENCOUNTERED
            ;THIS SECTION COMPARES READ DATA WITH ACTUAL DATA
F77E 37     COMPARE         PSH B          ;SAVE REGISTERS
F77F 36                     PSH A
F780 9F65                   STS   @TEMP5   ;SAVE STACK POINTER
F782 CE0021                 LDX   #BUF     ;POINT TO BEGINNING OF BUFFER
F785 9E6D   J6              LDS   @TEMP9   ;LOAD WITH DATA PATTERN ADDRESS
F787 34                     DES            ;POINT TO BEG. OF DATA PATTERN
F788 C605                   LDA B #05      ;COUNT BYTES IN PATTERN
F78A 32     J10             PUL A          ;GET BYTE OF PATTERN
F78B A100                   CMP A 0,X      ;COMPARE DATA
F78D 2706                   BEQ   J7       ;CONTINUE IF CORRECT
F78F 0D                     SEC            ;SET FLAG UPON ERROR CONDITION
F790 9E65   J8              LDS   @TEMP5   ;RESTORE STACK POINTER
F792 32                     PUL A          ;RESTORE REGISTERS
F793 33                     PUL B
F794 39                     RTS
            ;CONTINUE OPERATION IF CORRECT
F795 08     J7              INX            ;INC BUFFER POINTER
F796 8C0033                 CPX   #ENDBUF  ;CHECK IF ENTIRE BUFFER CHECKED
F799 0C                     CLC            ;CLEAR FLAG IF NO ERROR
F79A 27F4                   BEQ   J8       ;EXIT IF THE END
F79C 5A                     DEC B          ;CHECK IF LAST PATTERN
F79D 26EB                   BNE   J10      ;CONTINUE IF ALL CHECKS NOT DONE
F79F 20E4                   BRA   J6       ;REPEAT PROCEDURE

            ;FILL PAGE BUFFER WITH DATA PATTERN

F7A1 37     TRANSFER        PSH B          ;SAVE REGISTER
F7A2 9F65                   STS   @TEMP5   ;SAVE STACK POINTER
F7A4 CE0021                 LDX   #BUF     ;POINT TO BEGINNING OF BUFFER
F7A7 9E6D   J2              LDS   @TEMP9   ;GET POINTER OF DATA PATTERN
F7A9 34                     DES            ;POINT TO BEG. OF DATA PATTERN
F7AA C605                   LDA B #05      ;COUNT BYTES IN PATTERN
F7AC 32     J1              PUL A          ;GET BYTE OF PATTERN
F7AD A700                   STA A 0,X      ;STORE INTO BUFFER
F7AF 08                     INX            ;INC. BUFFER POINTER
F7B0 8C0033                 CPX   #ENDBUF  ;CHECK IF BUFFER FILLED
F7B3 2705                   BEQ   J5       ;EXIT IF FILED
F7B5 5A                     DEC B          ;CHECK IF END OF PATTERN
F7B6 26F4                   BNE   J1       ;CONTINUE IF NOT ALL
F7B8 20ED                   BRA   J2       ;REPEAT DATA PATTERN
F7BA 9E65   J5              LDS   @TEMP5   ;RESTORE STACK POINTER
F7BC 33                     PUL B          ;RESTORE REGISTER
F7BD 39                     RTS
            ;THIS SECTION CALLED UPON HARD ERROR CONDITION
F7BE 7C0073 HARDERR         INC   LOC2     ;INCREMENT HARD ERROR COUNTER
F7C1 CEFD0F U1              LDX   #M34     ;DISPLAY MESSAGE "DATA READ"
F7C4 BDF1BE                 JSR   MESSA    ;OUTPUT MESSAGE
F7C7 CE0000                 LDX   #0       ;SET UP PAGE POSN
F7CA BDF525                 JSR   OUTPUT   ;OUTPUT BUFFER CONTENTS
F7CD 8DD2                   BSR TRANSFER   ;REFILL BUFFER WITH DATA PATTERN
```

```
F7CF CEFD19                LDX   #M35      ;DISPALY MESSAGE "ACTUAL DATA PATTERN"
F7D2 BDF1BE                JSR   MESSA     ;OUTPUT MESSAGE
F7D5 BDF525                JSR   OUTPUT    ;DISPLAY READ DATA
F7D8 BDF43F                JSR   VERFY     ;OUTPUT RUNNING RESULTS
F7DB BDF436                JSR   B10B      ;OUTPUT - **WORKING**
F7DE 39                    RTS
           ;THIS SECTION CALLED UPON SOFT ERROR CONDITION
F7DF BDF431 SOFTERR        JSR   B10A      ;INCREMENT SOFT ERROR COUNTER
F7E2 20DD                  BRA   U1
           ;THIS SECTION UPDATES CYCLE COUNTER
F7E4 DE6B   COUNT          LDX   @TEMP8    ;LOAD COUNT VALUE
F7E6 08                    INX             ;INC. COUNT VALUE
F7E7 DF6B                  STX   @TEMP8    ;SAVE COUNT VALUE
F7E9 2605                  BNE   J9
F7EB DE69                  LDX   @TEMP7    ;LOAD MS. BYTES OF COUNT
F7ED 08                    INX             ;INC. COUNT VALUE
F7EE DF69                  STX   @TEMP7    ;SAVE COUNT VALUE
F7F0 39     J9             RTS

           ;THIS ROUTINE PERFORMS A MULTI-PAGE INITIALIZATION

F7F1 8610   MULTIPGE       LDA A #MULTI
F7F3 9783                  STA A CONT      ;INITIALIZE MULTI PAGE READ
F7F5 4F                    CLR A
F7F6 9780                  STA A PGSEL     ;INIT. BEGINNING PAGE FOR TRANSFER
F7F8 9781                  STA A PGSEL+1
F7FA 9786                  STA A PGCNT     ;SET UP NO. OF PAGES TO TRANSFER
F7FC C602                  LDA B #02       ;LS. BYTE OF PAGE COUNTS
F7FE D787                  STA B PGCNT+1
F800 39                    RTS

           ;TEST 3         -MAJOR LOOP NEAREST NEIGHBOR PATTERN TESTING
           ;               - PATTERN USED - BAA8255700

F801 CEFD2D TEST3          LDX   #M36      ;MESSAGE - TEST3
F804 BDF1BE                JSR   MESSA     ;OUTPUT MESSAGE
F807 BDF436                JSR   B10B      ;OUTPUT - **WORKING**
F80A CEFB0C U7             LDX   #PAT3     ;POINT TO BEGINNING OF PATTERN
F80D DF6D                  STX   @TEMP9    ;SAVE POINTER
F80F 8D90                  BSR   TRANSFER  ;FILL PAGE BUFFER WITH PATTERN
F811 8DDE                  BSR   MULTIPGE  ;INIT. FOR MULTI-PAGE
F813 CE0021                LDX   #BUF      ;POINT TO BEGINNING OF BUFFER
F816 A600                  LDA A 0,X       ;GET DATA BYTE
F818 9785                  STA A WR        ;PLACE DATA BYTE INTO WRITE FIFO
F81A 8614                  LDA A #MPWRCMD  ;LOAD WITH COMMAND
F81C 9783                  STA A CONT      ;ISSUE MULTI PAGE WRITE CMD.
F81E 0E     N1             CLI
F81F 3E                    WAI             ;WAIT FOR OPERATION
F820 08                    INX             ;POINT TO NEXT BYTE IN BUFFER
F821 8C0033                CPX   #ENDBUF   ;CHECK IF END OF BUFFER REACHED
F824 270F                  BEQ   N6        ;EXIT IF DONE
F826 A600                  LDA A 0,X       ;LOAD DATA
F828 9785                  STA A WR        ;STORE DATA
F82A C101                  CMP B #01H      ;ARE WE AT THE LAST PAGE
```

```
F82C 2605                   BNE   N7        ;IF NOT CONTINUE
```

```
F82E 8C0032                CPX   #ENDBUF-1;ARE WE AT THE END OF THE MULTI-PAGE
F831 2708                  BEQ   N8        ;BUFFER, EXIT IF YES
F833 20E9   N7             BRA   N1        ;CONTINUE IF NOT
F835 CE0020 N6             LDX   #BUF-1    ;REINITIALIZE BUFFER POINTER
F838 5A                    DEC B           ;CHECK IF ENTIRE BUFFER
F839 26E3                  BNE   N1        ;IS TRANSFERED
F83B BDF315 N8             JSR   A16       ;CHECK IF CONTROLLER BUSY
F83E 8680                  LDA A #RESETINT
F840 9783                  STA A CONT      ;RESET INTERRUPT
F842 8D23   U4             BSR   READ1     ;PERFORM MULTI PAGE READ
F844 8D9E                  BSR   COUNT     ;COUNT NO OF OPERATIONS
F846 8D3C                  BSR   COMPARE1  ;COMPARE RESULTS
F848 2505                  BCS   BELOW1    ;BRANCH ON ERROR CONDITION
F84A BDF3EE ABOVE1         JSR   IO        ;CHECK FOR ANY KEYBOARD IO
F84D 20F3                  BRA   U4        ;READ DATA
            ;SOFT ERROR SECTION
F84F C606   BELOW1         LDA B #06       ;SOFT ERROR COUNTER
F851 5A     U6             DEC B           ;DEC COUNTER
F852 270F                  BEQ   U5        ;CHECK IF MAX. NO. OF ERRORS REACHED
F854 37                    PSH B           ;SAVE COUNTER
F855 8D4E                  BSR   SOFTERR1  ;INCREMENT COUNTER
F857 BDF7A1                JSR   TRANSFER  ;FILL BUFFER WITH DATA
F85A 8D0B                  BSR   READ1     ;READ PAGE
F85C 8D26                  BSR   COMPARE1  ;COMPARE DATA
F85E 33                    PUL B           ;RESTORE COUNTER
F85F 25F0                  BCS   U6        ;REREAD IF ANOTHER ERROR
F861 20E7                  BRA   ABOVE1    ;EXIT IF NO ERROR
            ;THIS CALLED UPON HARD ERROR CONDITION
F863 8D45   U5             BSR   HARDERR1  ;UPDATE COUNTER
F865 20A3                  BRA   U7        ;RESTART TEST ON HARD ERROR
F867 8D88   READ1          BSR   MULTIPGE  ;INITLIALIZE MULTI PAGE PARAMETERS
F869 8612                  LDA A #MPRDCMD
F86B 9783                  STA A CONT      ;ISSUE MULTI PAGE READ CMD.
F86D CE0034                LDX   #BUF1     ;POINT TO BEGINNING OF PAGE BUFFER
F870 0E     J12            CLI
F871 3E                    WAI             ;WAIT FOR OPERATION
F872 9682                  LDA A RE        ;GET DATA BYTE
F874 A700                  STA A 0,X       ;STORE DATA BYTE
F876 08                    INX             ;INC. BUFFER POINTER
F877 8C0058                CPX   #ENDBUF1  ;FINAL BYTE TRANSFERED?
F87A 26F4                  BNE   J12       ;READ IF NOT COMPLETE
F87C BDF315                JSR   A16       ;CHECK FOR CONTROLLER BUSY
F87F 8680                  LDA A #RESETINT
F881 9783                  STA A CONT      ;RESET INTERRUPT
F883 39                    RTS

            ;THIS ROUTINE COMPARES READ DATA WITH ACTUAL DATA

F884 37     COMPARE1       PSH B           ;SAVE REGISTER
F885 9F65                  STS   @TEMP5    ;SAVE STACK POINTER
F887 CE0034                LDX   #BUF1     ;POINT TO BEG. OF BUFFER
F88A 8E0020 J15            LDS   #BUF-1    ;POINT TO BEG. OF SECOND BUFFER
F88D 32     J17            PUL A           ;RECALL DATA
F88E A100                  CMP A 0,X       ;COMPARE DATA
```

```
F890 2705                    BEQ   J14       ;CONTINUE IF OK
```

```
F892 0D                     SEC             ;SET FLAG ON ERROR
F893 9E65   J16             LDS   @TEMP5    ;RESTORE STACK POINTER
F895 33                     PUL B           ;RESTORE REGISTER
F896 39                     RTS
F897 08     J14             INX             ;POINT TO NEXT DATA BYTE
F898 8C0046                 CPX   #BUF1+18  ;IS THE END OF THE BUFFER REACHED
F89B 27ED                   BEQ   J15       ;POINT TO BEG. IF YES
F89D 8C0058                 CPX   #ENDBUF1  ;END OF PAGE TWO?
F8A0 0C                     CLC             ;RESET ERROR FLAG
F8A1 27F0                   BEQ   J16       ;EXIT IF FINISHED
F8A3 20E8                   BRA   J17       ;CONTINUE IF NOT
            ;SOFT ERROR SECTION
F8A5 BDF431 SOFTERR1        JSR   B10A      ;INCREMENT SOFT ERROR COUNTER
F8A8 2003                   BRA   J18
            ;HARD ERROR SECTION
F8AA 7C0073 HARDERR1        INC   LOC2      ;INCREMENT HARD ERROR COUNTER
F8AD CEFD19 J18             LDX   #M35      ;DISPLAY MESSAGE "ACTUAL DATA PATTERN"
F8B0 BDF1BE                 JSR   MESSA     ;OUTPUT MESSAGE
F8B3 CE0000                 LDX   #0        ;POINT TO PAGE ZERO
F8B6 DF5C                   STX   @TEMP2    ;SAVE PAGE POSN
F8B8 BDF525                 JSR   OUTPUT    ;OUTPUT PAGE DATA
F8BB C602                   LDA B #02H      ;OUTPUT ACTUAL NO. OF PAGES
F8BD CEFD0F                 LDX   #M34      ;DISPLAY MESSAGE "DATA READ"
F8C0 BDF1BE                 JSR   MESSA     ;OUTPUT MESSAGE
F8C3 9F65                   STS   @TEMP5    ;THE FOLLOWIND SECTION
F8C5 8E0033                 LDS   #BUF1-1   ;TRANSFERS THE PAGES READ
F8C8 CE0021 J21             LDX   #BUF      ;INTO A BUFFER SUCH
F8CB 32     J19             PUL A           ;THAT THE DATA MAY
F8CC A700                   STA A 0,X       ;BE OUTPUTED
F8CE 08                     INX             ;INC. BUFFER POINTER
F8CF 8C0033                 CPX   #ENDBUF   ;END OF BUFFER?
F8D2 26F7                   BNE   J19       ;FILL BUFFER IF NOT
F8D4 9E65                   LDS   @TEMP5    ;RESTORE STACK POINTER
F8D6 DE5C                   LDX   @TEMP2    ;RECALL PAGE POSN
F8D8 BDF525                 JSR   OUTPUT    ;OUTPUT PAGE
F8DB 5A                     DEC B           ;DEC PAGE COUNT
F8DC 270A                   BEQ   J20       ;EXIT IF DONE
F8DE 7C005D                 INC   TEMP2+1   ;INCREMENT TO DISPLAY THE SECOND PAGE
F8E1 9F65                   STS   @TEMP5    ;SAVE STACK POINTER
F8E3 8E0045                 LDS   #BUF1+18-1;POINT TO BEG. OF BUFFER
F8E6 20E0                   BRA   J21       ;REPEAT ABOVE PROCEDURE FOR PAGE TWO
F8E8 9E65   J20             LDS   @TEMP5    ;RETORE STACK POINTER
F8EA BDF43F                 JSR   VERFY     ;OUTPUT RUNNING RESULTS
F8ED BDF436                 JSR   B10B      ;OUTPUT - **WORKING**
F8F0 39                     RTS

            ;CHECK TO SEE IF PAGE BUFFER SHOULD BE CLEARED OR SET

F8F1 36     CHECK1          PSH A           ;SAVE REGISTER
F8F2 DF5E                   STX   @TEMP3    ;SAVE INDEX REGISTER
F8F4 CE0021                 LDX   #BUF      ;POINT TO BEG. OF BUFFER
F8F7 58                     ASL B           ;SHIFT DATA TO CHECK BIT VALUE
F8F8 2406                   BCC   K1        ;BRANCH IF LOW
```

```
F8FA 8D14                  BSR   SETBUF    ;FILL BUFFER WITH ONES
F8FC DE5E                  LDX   @TEMP3    ;RESTORE REGISTERS
F8FE 32                    PUL A
F8FF 39                    RTS
r900 8D04   K1             BSR   CLEARBUF  ;FILL BUFFER WITH ZEROS
F902 32                    PUL A           ;RETORE REGISTERS
F903 DE5E                  LDX   @TEMP3
F905 39                    RTS

            ;FILL BUFFER WITH 00 HEX

F906 4F     CLEARBUF       CLR A           ;CLEAR REGISTER
F907 A700   K2             STA A 0,X       ;STORE IN PAGE BUFFER
F909 08                    INX             ;POINT TO NEXT POSITON
F90A 8C0033                CPX   #ENDBUF   ;IS END REACHED?
F90D 26F8                  BNE   K2        ;CONTINUE IF NOT
F90F 39                    RTS

            ;FILL BUFFER WITH FF HEX

F910 86FF   SETBUF         LDA A #0FFH     ;SET REGISTER TO ONE
F912 20F3                  BRA   K2        ;FILL BUFFER WITH ONES

            ;TEST 4        -MINOR LOOP FAILURE MODE TESTING
            ;TEST 4A       -MINOR LOOP NEAREST NEIGHBOR TESTING
            ;              -PATTERN USED - FF08800F77
            ;                            - BAA8255700

r914 CEFD34 TEST4          LDX   #M37      ;MESSAGE - TEST4A
F917 BDF1BE                JSR   MESSA     ;OUTPUT MESSAGE
F91A CE0000                LDX   #0        ;SET TO PAGE ZERO
F91D DF60                  STX   @TEMP4    ;INITIALIZE STARTING PAGE
F91F CEFB07                LDX   #PAT2     ;POINT TO BEGINNING OF PATTERN
F922 200E                  BRA   K3        ;BRACH AROUND
F924 CEFD56 TEST4A         LDX   #M40      ;MESSAGE - TEST 4B
F927 BDF1BE                JSR   MESSA     ;OUTPUT MESSAGE
F92A CE0000                LDX   #0        ;SET TO PAGE ZERO
F92D DF60                  STX   @TEMP4    ;INITIALIZE STARTING PAGE
F92F CEFB0C                LDX   #PAT3     ;POINT TO BEGINNING OF PATTERN
F932 DF6D   K3             STX   @TEMP9    ;SAVE PATTERN POINTER
F934 DF6F                  STX   @TEMP10   ;SAVE IN ANOTHER TEMP. REG.
F936 08                    INX             ;INCREMENT POINTER TO END OF
r937 08                    INX             ;DATA PATTERN
r938 08                    INX
F939 08                    INX
F93A 08                    INX             ;REG. CONTAINS END ADDRESS
F93B DF71                  STX   @TEMP11   ;SAVE POINTER FOR PATTERN END
F93D BDF436                JSR   B10B      ;OUTPUT - **WORKING**
F940 DE6D   K6             LDX   @TEMP9    ;GET PATTERN POINTER
r942 E600   K5             LDA B 0,X       ;GET DATA PATTERN
F944 8608                  LDA A #08H      ;SET UP BIT COUNTER
F946 8DA9   K4             BSR   CHECK1    ;SET OR CLEAR BUFFER
F948 DE60                  LDX   @TEMP4    ;SET UP WRITE PAGE
```

```
F94A BDF31C                JSR   SWRITE    ;WRITE PAGE BUFFER
F94D 08                    INX              ;INC. PAGE POSN INDICATOR
F94E DF60                  STX   @TEMP4     ;SAVE NEW PAGE COUNTER
F950 8C0281                CPX   #ENDMEM    ;CHECK IF FILLED
F953 2712                  BEQ   K7         ;EXIT IF DONE, NOW COMPARE
F955 4A                    DEC A            ;ROTATED ALL BITS
F956 26EE                  BNE   K4         ;CONTINUE IF NOT FINISHED
F958 DE6D                  LDX   @TEMP9     ;FETCH DATA POINTER
F95A 08                    INX              ;POINT TO NEW DATA
F95B DF6D                  STX   @TEMP9     ;SAVE NEW DATA POINTER
F95D 9C71                  CPX   @TEMP11    ;ARE WE AT THE END?
F95F 26E1                  BNE   K5         ;CONTINUE IF NOT
F961 DE6F                  LDX   @TEMP10    ;GET BEGINNING DATA POINTER
F963 DF6D                  STX   @TEMP9     ;SAVE BEGINNING DATA POINTER
F965 20D9                  BRA   K6         ;REPEAT FROM BEGINNING
F967 CE0000 K7             LDX   #0         ;POINT TO PAGE ZERO
F96A DF60                  STX   @TEMP4     ;SAVE PAGE POINTER
F96C DE6F                  LDX   @TEMP10    ;GET BEGINNING DATA POINTER
F96E DF6D                  STX   @TEMP9     ;REINITIALIZE DATA POINTER
F970 E600    K10           LDA B 0,X        ;GET DATA
F972 8608                  LDA A #08H       ;SET UP BIT COUNTER
F974 DE60    K9            LDX   @TEMP4     ;GET PAGE COUNTER
F976 BDF2EB                JSR   SREAD      ;READ PAGE
F979 08                    INX              ;UPDATE PAGE POINTER
F97A DF60                  STX   @TEMP4     ;SAVE NEW PAGE COUNTER
F97C 8C0281                CPX   #ENDMEM    ;ENTIRE MEMORY READ?
F97F 27E6                  BEQ   K7         ;REPEAT OPERATION
F981 DE6D                  LDX   @TEMP9     ;GET DATA POINTER
F983 BDF7E4                JSR   COUNT      ;INCREMENT CYCLE COUNTER
F986 8D42                  BSR   COMPARE2   ;COMPARE READ DATA WITH ACTUAL DATA
F988 2515                  BCS   BELOW2     ;BRACH ON ERROR CONDITION
F98A BDF3EE ABOVE2         JSR   IO         ;CHECK FOR TERMINAL ENTRY
F98D 4A                    DEC A            ;ALL BITS ROTATED
F98E 26E4                  BNE   K9         ;CONTINUE IF NOT FINISHED
F990 DE6D                  LDX   @TEMP9     ;UPDATE DATA PATTERN POINTER
F992 08                    INX
F993 DF6D                  STX   @TEMP9     ;SAVE PATTERN POINTER
F995 9C71                  CPX   @TEMP11    ;END OF PATTERN REACHED?
F997 26D7                  BNE   K10        ;IF END OF PATTERN POINTER
F999 DE6F                  LDX   @TEMP10    ;REACHED THEN RESET POINTER
F99B DF6D                  STX   @TEMP9     ;SAVE POINTER
F99D 20D1                  BRA   K10        ;REPEAT
          ;SOFT ERROR SECTION
F99F 36      BELOW2        PSH A            ;SAVE REGISTERS
F9A0 37                    PSH B
F9A1 C606                  LDA B #06H       ;SOFT ERROR COUNTER
F9A3 5A      K12           DEC B
F9A4 2714                  BEQ   K11        ;CHECK IF MAX. ERRORS ENCOUNTERED
F9A6 37                    PSH B            ;SAVE COUNT VALUE
F9A7 8D4E                  BSR   SOFTERR2   ;OUTPUT READ AND COMPARED DATA
F9A9 DE60                  LDX   @TEMP4     ;GET PAGE POSN
F9AB 09                    DEX              ;POINT TO CORRECT PAGE
F9AC BDF2EB                JSR   SREAD      ;REREAD PAGE
```

```
r9AF D674                   LDA B @LOC3     ;REREAD OLD BIT TEST VALUE
r9B1 8D17                   BSR   COMPARE2 ;COMPARE DATA
r9B3 33                     PUL B          ;GET COUNT VALUE
F9B4 25ED                   BCS   K12      ;REPEAT ON ERROR
F9B6 33                     PUL B          ;RESTORE REGISTERS
F9B7 32                     PUL A
F9B8 20D0                   BRA   ABOVE2   ;RETURN IF FINE
r9BA 8D40   K11             BSR   HARDERR2 ;OUTPUT DATA
r9BC 33                     PUL B          ;RESTORE STACK POINTER
r9BD 32                     PUL A
F9BE DE6F                   LDX   @TEMP10  ;REINITIALIZE POINTERS
F9C0 DF6D                   STX   @TEMP9
F9C2 CE0000                 LDX   #0       ;REINITIALIZE PAGE POSN
F9C5 DF60                   STX   @TEMP4
r9C7 7EF940                 JMP   K6       ;RESTART TEST
            ;COMPARE READ DATA WITH ACTUAL DATA
F9CA 36     COMPARE2        PSH A          ;SAVE ACCUMULATORS
F9CB D774                   STA B @LOC3    ;STORE DATA PATTERN
F9CD DF5E                   STX   @TEMP3   ;SAVE INDEX REGISTER
F9CF 58                     ASL B          ;ROTATE ACC. B, CHECK TO
r9D0 CE0021                 LDX   #BUF     ;SEE IF BUFFER SHOULD BE
F9D3 2411                   BCC   K21      ;FF HEX OR 00 HEX
F9D5 A600   K8              LDA A 0,X      ;LOAD BUFFER DATA
F9D7 81FF                   CMP A #0FFH    ;COMPARE WITH FF HEX
F9D9 2619                   BNE   ERROR3   ;BRANCH ON ERROR
F9DB 08                     INX            ;POINT TO NEXT DATA BYTE
F9DC 8C0033                 CPX   #ENDBUF  ;HAS END OF BUFFER BEEN REACHED?
F9DF 26F4                   BNE   K8       ;CONTINUE IF NOT
r9E1 0C     N2              CLC            ;CLEAR ERROR FLAG
F9E2 32     N4              PUL A          ;RESTORE REGISTERS
F9E3 DE5E                   LDX   @TEMP3
F9E5 39                     RTS

            ;THIS ROUTINE COMPARES DATA WITH 00 HEX

F9E6 A600   K21             LDA A 0,X      ;LOAD DATA
r9E8 8100                   CMP A #0H      ;COMPARE WITH 00 HEX
F9EA 2608                   BNE   ERROR3   ;BRANCH ON ERROR CONDITON
F9EC 08                     INX            ;POINT TO NEXT DATA BYTE
F9ED 8C0033                 CPX   #ENDBUF  ;HAS END OF BUFFER BEEN REACHED
F9F0 26F4                   BNE   K21      ;CONTINUE IF NOT
r9F2 20ED                   BRA   N2       ;EXIT IF DONE
F9F4 0D     ERROR3          SEC            ;SET ERROR FLAG
F9F5 20EB                   BRA   N4       ;RETURN
F9F7 BDF431 SOFTERR2        JSR   B10A     ;INCREMENT SOFT ERROR COUNTER
F9FA 2003                   BRA   K13      ;OUTPUT RELEVANT DATA
F9FC 7C0073 HARDERR2        INC   LOC2     ;INCREMENT HARD ERROR COUNTER
F9FF CEFD0F K13             LDX   #M34     ;DISPLAY MESSAGE "DATA READ"
rA02 BDF1BE                 JSR   MESSA    ;OUTPUT MESSAGE
rA05 DE60                   LDX   @TEMP4   ;GET PAGE POSN
rA07 09                     DEX            ;POINT TO CORRECT PAGE
rA08 BDF525                 JSR   OUTPUT   ;OUTPUTED PAGE DATA
FA0B D674                   LDA B @LOC3    ;LOAD WITH ACTUAL DATA PATTERN
FA0D BDF8F1                 JSR   CHECK1   ;CHECK IF ONES OR ZEROS
```

```
FA10 CEFD19                LDX   #M35      ;DISPLAY MESS. "ACTUAL DATA PATTERN"
```

```
FA13 BDF1BE                JSR   MESSA     ;OUTPUT MESSAGE
FA16 DE60                  LDX   @TEMP4    ;GET PAGE POSN
FA18 09                    DEX              ;POINT TO CORRECT PAGE
FA19 BDF525                JSR   OUTPUT    ;OUTPUT PAGE OF DATA
FA1C BDF43F                JSR   VERFY     ;OUTPUT RUNNING RESULTS
FA1F BDF436                JSR   B10B      ;OUTPUT - **WORKING**
FA22 39                    RTS

            ;TEST 5        -TEST EACH MINOR LOOP WITH EITHER ALL 1s OR 0s

FA23 CEFD3B TEST5          LDX   #M38      ;MESSAGE  - TEST5
FA26 BDF1BE                JSR   MESSA     ;OUTPUT MESSAGE
rA29 BDF436                JSR   B10B      ;OUTPUT - **WORKING**
FA2C CE0000                LDX   #0        ;INITIALIZE PAGE POSN
FA2F DF5C                  STX   @TEMP2    ;SAVE PAGE POSN.
FA31 BDF4D3                JSR   CHNGE1    ;FILL PAGE BUFFER WITH DATA PATTERN
rA34 BDF436                JSR   B10B      ;OUTPUT - **WORKING**
rA37 08      K23           INX             ;INC. PAGE COUNTER
            ;THIS SECTION WRITES THE SAME PATTERN INTO THE ENTIRE MEMORY
FA38 8C0281                uPX   #ENDMEM   ;IS END OF THE MEMORY REACHED?
rA3B 2705                  uEQ   N3        ;EXIT IF YES
FA3D BDF31C                JSR   SWRITE    ;WRITE OUT PAGE
FA40 20F5                  uRA   K23       ;CONTINUE
            ;THIS SECTION RESTORES THE WRITE BUFFER CONTENTS
FA42 9F65    N3            STS   @TEMP5    ;SAVE STACK POINTER
FA44 8E0020                LDS   #BUF-1    ;POINT TO BEG. OF BUFFER
FA47 CE0034                LDX   #BUF1     ;POINT TO OTHER BUFFER
FA4A 32      K22           PUL A           ;GET DATA
FA4B A700                  STA A 0,X       ;STORE DATA
rA4D 08                    INX             ;INCREMENT POINTER
rA4E 8C0046                CPX   #BUF1+18  ;IS END REACHED?
rA51 26F7                  uNE   K22       ;CONTINUE IF NOT
FA53 9E65                  LDS   @TEMP5    ;RESTORE STACK POINTER
FA55 CE0000 K24            LDX   #0        ;STORE STARTING PAGE NUMBER
FA58 FF0071 K25            STX   TEMP11
FA5B BDF2EB                JSR   SREAD     ;READ PAGE
FA5E BDF7E4                JSR   COUNT     ;INCREMENT THE NO. OF OPERATIONS
FA61 8D2C                  uSR   COMPARE3  ;COMPARE RESULTS
FA63 250E                  uCS   BELOW3    ;ACCOUNT FOR ERROR
FA65 BDF3EE ABOVE3         JSR   IO        ;CHECK FOR KEYBOARD IO
rA68 FE0071                LDX   TEMP11    ;RECALL  PAGE NUMBER
FA6B 08                    INX             ;UPDATE PAGE POSN
FA6C 8C0281                uPX   #ENDMEM   ;HAS END BEEN REACHED YET?
FA6F 26E7                  BNE   K25       ;CONTINUE IF NOT FINISHED
rA71 20E2                  BRA   K24       ;REPEAT ENTIRE PROCEDURE
            ;SOFT ERROR SECTION
FA73 C606    BELOW3        LDA B #06H      ;SOFT ERROR COUNTER
FA75 5A      K15           DEC B           ;DEC COUNTER
FA76 270E                  BEQ   K14       ;EXIT IF MAX. NO EXCEED
FA78 37                    PSH B           ;SAVE COUNTER
FA79 BDFAC5                JSR   SOFTERR3  ;UPDATE COUNTER
rA7C BDF2EB                JSR   SREAD     ;REREAD PAGE
FA7F 8D0E                  BSR   COMPARE3  ;COMPARE READ DATA
```

```
FA81 33                     PUL B           ;RECALL COUNTER
FA82 25F1                   BCS   K15       ;REPEAT ON ERROR COND.
FA84 20DF                   BRA   ABOVE3    ;CONTINUE TEST
          ;HARD ERROR SECTION
FA86 8D42 K14               BSR   HARDERR3  ;INCREMENT COUNTERS
FA88 8D23                   BSR   TRANSFR1  ;REINITIALIZE BUFFER
FA8A CEFFFF                 LDX   #0FFFFH   ;SET UP FOR PAGE POSN
FA8D 20A8                   BRA   K23       ;REPEAT TEST
          ;COMPARE ACTUAL DATA WITH READ DATA
FA8F DF5E COMPARE3          STX   @TEMP3    ;SAVE INDEX REGISTER
FA91 9F65                   STS   @TEMP5    ;SAVE STACK POINTER
FA93 8E0033                 LDS   #BUF1-1   ;POINT TO BEGINNING OF ORIGINAL PATTERN
FA96 CE0021                 LDX   #BUF      ;POINT TO READ PATTERN
FA99 32     K17             PUL A           ;GET DATA
FA9A A100                   CMP A 0,X       ;COMPARE RESULTS
FA9C 260C                   BNE   K16       ;SET FLAG ON ERROR COND.
FA9E 08                     INX             ;POINT TOI NEXT DATA
FA9F 8C0033                 CPX   #ENDBUF   ;HAS BUFFER BEEN CHECKED?
FAA2 26F5                   BNE   K17       ;CONTINUE IF NOT
FAA4 0C                     CLC             ;CLEAR FLAG IF ALL IS OK!
FAA5 DE5E K18               LDX   @TEMP3    ;RESTORE REGISTERS
FAA7 9E65                   LDS   @TEMP5
FAA9 39                     RTS
          ;THIS SECTION SETS THE ERROR FLAG
FAAA 0D     K16             SEC             ;SET FLAG IF ERROR DETECTED
FAAB 20F8                   BRA   K18       ;RETURN

          ;FILL PAGE BUFFER WITH DATA STORED AT ALTERNATIVE PAGE BUFFER

FAAD DF5E TRANSFR1          STX   @TEMP3    ;SAVE REGISTERS
FAAF 9F65                   STS   @TEMP5
FAB1 8E0033                 LDS   #BUF1-1   ;POINT TO BEG. OF BUFFER
FAB4 CE0021                 LDX   #BUF      ;POINT OT FIRST BUFFER
FAB7 32     K19             PUL A           ;GET DATA
FAB8 A700                   STA A 0,X       ;STORE DATA
FABA 08                     INX             ;INCREMENT BUFFER POINTER
FABB 8C0033                 CPX   #ENDBUF   ;IS END REACHED?
FABE 26F7                   BNE   K19       ;CONTINUE IF NOT
FAC0 DE5E                   LDX   @TEMP3    ;RESTORE REGISTERS
FAC2 9E65                   LDS   @TEMP5
FAC4 39                     RTS
          ;INCREMENT SOFTERROR COUNTER
FAC5 BDF431 SOFTERR3        JSR   B10A      ;INCREMENT SOFT ERROR COUNTER
FAC8 2003                   BRA   K20       ;CONTINUE
          ;HARD ERROR SECTION
FACA 7C0073 HARDERR3        INC   LOC2      ;INCREMENT HARD ERROR COUNTER
FACD CEFD0F K20             LDX   #M34      ;DISPLAY MESS. "DATA READ"
FAD0 BDF1BE                 JSR   MESSA     ;OUTPUT MESSAGE
FAD3 DE71                   LDX   @TEMP11   ;GET PAGE POSN.
FAD5 BDF525                 JSR   OUTPUT    ;OUTPUT BUFFER CONTENTS
FAD8 8DD3                   BSR   TRANSFR1  ;FILL BUFFER WITH OTHER PATTERN
FADA CEFD19                 LDX   #M35      ;DISPLAY MESS. "ACTUAL DATA PATTERN"
FADD BDF1BE                 JSR   MESSA     ;OUTPUT MESSAGE
FAE0 DE71                   LDX   @TEMP11   ;GET PAGE POSN
```

```
FAE2 BDF525                JSR   OUTPUT   ;OUTPUT READ & ACTUAL DATA
```

```
FAE5 BDF43F                JSR   VERFY     ;OUTPUT RUNNING RESULTS
FAE8 BDF436                JSR   B10B      ;OUTPUT - **WORKING**
FAEB 39                    RTS

             ;THIS IS THE INTERRUPT ROUTINE USED FOR
             ;MULTI PAGE READ/WRITE OPERATION

FAEC 32      INTERR        PUL A           ;GET COND. CODES
FAED 8A10                  ORA A #10H      ;SET INTERRUPT MASK
rAEF 36                    PSH A           ;RESTORE COND. CODES
FAF0 3B                    RTI             ;RETURN

             ;POWER FAIL ROUTINE

FAF1 0F      INTERR1       SEI
FAF2 CE2000                LDX   #DELAY    ;LOAD FOR BROWN OUT DELAY
rAF5 09      A5            DEX
FAF6 26FD                  BNE   A5
rAF8 760100                ROR   PIADATAA  ;CHECK IF REAL POWER FAILURE
FAFB 2504                  BCS   BELOW4
FAFD B601F0                LDA A DISABLE   ;DISABLE BUBBLE MODULE FOR POWER FAIL
FB00 3E                    WAI             ;WAIT FOR POWER TO GO DOWN
rB01 3B      BELOW4        RTI             ;RETURN IF NO POWER FAIL

             ;DATA PATTERNS FOR TESTS

FB02 F1150EEPAT1           BYTE  0F1H,15H,0EH,0EAH,00H
rB06 00
FB07 FF08800PAT2           BYTE  0FFH,08H,80H,0FH,77H
FB0B 77
FB0C BAA8255PAT3           BYTE  0BAH,0A8H,25H,57H,00H
FB10 00

             ;MESSAGE TABLE
F
rB11 5245434M1     ASCII  "RECEPTION ERROR@"
rB15 5054494
FB19 4E20455
FB1D 524F524
FB21 58414D4M2     ASCII  "XAM@"
FB25 554D504M3     ASCII  "UMP@"
FB29 424F532PRMPT1 ASCII  "BOS - VERSION 2.1@"
rB2D 2D20564
FB31 5253494
FB35 4E20322
FB39 3140
FB3B 434F4D3PROMPT ASCII  "COM:@"
FB3F 40
FB40 4255424MESAGE ASCII  "BUBBLE I/O IN ASCII OR HEX?"
FB44 4C45204
FB48 2F4F204
FB4C 4E20415
FB50 4349492
```

```
FB54 4F52204
FB58 45583F
FB5B 0D             BYTE  0DH
FB5C 454E544        ASCII  "ENTER <A> OR <H>@"
FB60 52203C4
FB64 3E204F5
FB68 203C483
FB6C 40
FB6D 4F4E544M4      ASCII  "ONTINOUS BUBBLE I/O@"
FB71 4E4F555
FB75 2042554
FB79 424C452
FB7D 492F4F4
FB81 5354415M5      ASCII  "START ADDRESS:@"
FB85 5420414
FB89 4452455
FB8D 533A40
FB90 454E442M6      ASCII  "END ADDRESS:@"
FB94 4144445
FB98 4553533
FB9C 40
FB9D 4441544M7      ASCII  "DATA PATTERN:@"
FBA1 2050415
FBA5 5445524
FBA9 3A40
FBAB 5245414M8      ASCII  "READ ERROR@"
FBAF 2045525
FBB3 4F5240
FBB6 4541444M9      ASCII  "EAD@"
FBBA 494C4C4M10     ASCII  "ILLEGAL ADDRESS@"
FBBE 47414C2
FBC2 4144445
FBC6 4553534
FBCA 3D3D3E4M11     ASCII  "==>@"
FBCE 5041474M12     ASCII  "PAGE:@"
FBD2 3A40
FBD4 494C4C4M13     ASCII  "ILL@"
FBD8 4D454D4M14     ASCII  "MEMORY FILLED@"
FBDC 5259204
FBE0 494C4C4
FBE4 4440
FBE6 4953544M15     ASCII  "IST@"
FBEA 52414E5M16     ASCII  "RANSFER@"
FBEE 4645524
FBF2 46524F4M17     ASCII  "FROM RAM TO BUBBLE-ENTER <B> OR"
FBF6 2052414
FBFA 20544F2
FBFE 4255424
FC02 4C452D4
FC06 4E54455
FC0A 203C423
FC0E 204F52
FC11 0D             BYTE  0DH
```

```
FC12 454E544       ASCII  "ENTER <R> IF TRANSFER FROM BUBBLE TO RAM ==>@"
FC16 52203C5
FC1A 3E20494
FC1E 2054524
FC22 4E53464
FC26 5220465
FC2A 4F4D204
FC2E 5542424
FC32 4520544
FC36 2052414
FC3A 203D3D3
FC3E 40
FC3F 5452414M18    ASCII  "TRANSFER PAGE NO:@"
FC43 5346455
FC47 2050414
FC4B 45204E4
FC4F 3A40
FC51 5452414M19    ASCII  "TRANSFER COMPLETE@"
FC55 5346455
FC59 20434F4
FC5D 504C455
FC61 4540
FC63 5354415M20    ASCII  "START PAGE:@"
FC67 5420504
FC6B 47453A4
FC6F 454E442M21    ASCII  "END PAGE:@"
FC73 5041474
FC77 3A40
FC79 5452414M22    ASCII  "TRANSFER RAM ADDRESS:@"
FC7D 5346455
FC81 2052414
FC85 2041444
FC89 5245535
FC8D 3A40
FC8F 4E49542M23    ASCII  "NIT. CONT.@"
FC93 20434F4
FC97 542E40
FC9A 08      M24   BYTE  08H
FC9B 4441544       ASCII  "DATA INPUT MODE@"
FC9F 20494E5
FCA3 5554204
FCA7 4F44454
FCAB 4441544M25    ASCII  "DATA TRANSFERED@"
FCAF 2054524
FCB3 4E53464
FCB7 5245444
FCBB 454C504M26    ASCII  "ELP@"
FCBF 4F20465M27    ASCII  "O FROM:@"
FCC3 4F4D3A4
FCC7 4E4F2E2M30    ASCII  "NO. OF SOFT ERRORS=@"
FCCB 4F46205
FCCF 4F46542
FCD3 4552524
```

```
FCD7 52533D4
FCDB 2A2A205M29     ASCII  "** WORKING **@"
FCDF 4F524B4
FCE3 4E47202
FCE7 2A40
FCE9 4E4F2E2M31     ASCII  "NO. OF R/W OPERATIONS =@"
FCED 4F46205
FCF1 2F57204
FCF5 5045524
FCF9 54494F4
FCFD 53203D4
FD01 5445535M32     ASCII  "TEST 1@"
FD05 203140
FD08 5445535M33     ASCII  "TEST 2@"
FD0C 203240
FD0F 4441544M34     ASCII  "DATA READ@"
FD13 2052454
FD17 4440
FD19 4143545M35     ASCII  "ACTUAL DATA PATTERN@"
FD1D 414C204
FD21 4154412
FD25 5041545
FD29 45524E4
FD2D 5445535M36     ASCII  "TEST 3@"
FD31 203340
FD34 5445535M37     ASCII  "TEST 4@"
FD38 203440
FD3B 5445535M38     ASCII  "TEST 5@"
FD3F 203540
FD42 4E4F2E2M39     ASCII  "NO. OF HARD ERRORS=@"
FD46 4F46204
FD4A 4152442
FD4E 4552524
FD52 52533D4
FD56 5445535M40     ASCII  "TEST 4A@"
FD5A 2034414
FD5E 5448452M28     ASCII  "THE FOLLOWING CMDS. ARE AVAILABLE"
FD62 464F4C4
FD66 4F57494
FD6A 4720434
FD6E 44532E2
FD72 4152452
FD76 4156414
FD7A 4C41424
FD7E 45
FD7F 0D             BYTE   0DH
FD80 43202D4        ASCII  "C -CONT. BUBBLE R/W OPER."
FD84 4F4E542
FD88 2042554
FD8C 424C452
FD90 522F572
FD94 4F50455
FD98 2E
```

```
FD99 0D            BYTE  0DH
FD9A 44202D4       ASCII  "D -MEM DUMP"
FD9E 454D204
FDA2 554D50
FDA5 0D            BYTE  0DH
FDA6 45202D4       ASCII  "E -EXAM MEM LOC."
FDAA 58414D2
FDAE 4D454D2
FDB2 4C4F432
FDB6 0D            BYTE  0DH
FDB7 20202D5       ASCII  "  -WITH C-CHANGE, <B>-BACK, CR.-NEXT"
FDBB 4954482
FDBF 432D434
FDC3 414E474
FDC7 2C203C4
FDCB 3E2D424
FDCF 434B2C2
FDD3 43522E2
FDD7 4E45585
FDDB 0D            BYTE  0DH
FDDC 46202D4       ASCII  "F -FILL A BLOCK OF MOD. WITH A DATA PATTERN"
FDE0 494C4C2
FDE4 4120424
FDE8 4F434B2
FDEC 4F46204
FDF0 4F442E2
FDF4 5749544
FDF8 2041204
FDFC 4154412
FE00 5041545
FE04 45524E
FE07 0D            BYTE  0DH
FE08 47202D4       ASCII  "G -EXEC PROGRAM"
FE0C 5845432
FE10 50524F4
FE14 52414D
FE17 0D            BYTE  0DH
FE18 48202D4       ASCII  "H -HELP FACILITY"
FE1C 454C502
FE20 4641434
FE24 4C49545
FE28 0D            BYTE  0DH
FE29 49202D4       ASCII  "I -INIT. CONT."
FE2D 4E49542
FE31 20434F4
FE35 542E
FE37 0D            BYTE  0DH
FE38 4C202D4       ASCII  "L -LIST BLOCK OF BUBBLE MEMORY TO TERM."
FE3C 4953542
FE40 424C4F4
FE44 4B204F4
FE48 2042554
FE4C 424C452
```

```
FE50 4D454D4
FE54 5259205
FE58 4F20544
FE5C 524D2E
FE5F 0D             BYTE  0DH
FE60 52202D5        ASCII  "R -READ A PAGE FROM MODULE"
FE64 4541442
FE68 4120504
FE6C 4745204
FE70 524F4D2
FE74 4D4F445
FE78 4C45
FE7A 0D             BYTE  0DH
FE7B 20202D5        ASCII  "  -WITH C-CHANGE, <B>-BACK, N-NEXT PAGE"
FE7F 4954482
FE83 432D434
FE87 414E474
FE8B 2C203C4
FE8F 3E2D424
FE93 434B2C2
FE97 4E2D4E4
FE9B 5854205
FE9F 414745
FEA2 0D             BYTE  0DH
FEA3 20202D5        ASCII  "  -WITH CHANGE, <^> RETAINS PRESENT DATA"
FEA7 4954482
FEAB 4348414
FEAF 47452C2
FEB3 3C3E205
FEB7 4554414
FEBB 4E53205
FEBF 5245534
FEC3 4E54204
FEC7 415441
FECA 0D             BYTE  0DH
FECB 53202D4        ASCII  "S -DATA INPUT INTO MODULE,HALF DUPLEX COMM."
FECF 4154412
FED3 494E505
FED7 5420494
FEDB 544F204
FEDF 4F44554
FEE3 452C484
FEE7 4C46204
FEEB 55504C4
FEEF 5820434
FEF3 4D4D2E
FEF6 0D             BYTE  0DH
FEF7 20202D4        ASCII  "  -DATA BLOCK MUST BE PRECEDED WITH A <W>, AND ENDED WITH </>
FEFB 4154412
FEFF 424C4F4
FF03 4B204D5
FF07 5354204
FF0B 4520505
```

```
FF0F 4543454
FF13 4544205
FF17 4954482
FF1B 41203C5
rF1F 3E2C204
FF23 4E44204
FF27 4E44454
rF2B 2057495
FF2F 48203C2
rF33 3E
FF34 0D             BYTE  0DH
rF35 54202D5        ASCII  "T -TRANSFER DATA FROM BUBBLE TO RAM OR FROM"
rF39 52414E5
FF3D 4645522
rF41 4441544
rF45 2046524
FF49 4D20425
FF4D 42424C4
rF51 20544F2
FF55 52414D2
rF59 4F52204
FF5D 524F4D
FF60 0D             BYTE  0DH
FF61 20202D5        ASCII  "  -RAM TO BUBBLE"
rF65 414D205
FF69 4F20425
rF6D 42424C4
rF71 0D             BYTE  0DH
FF72 56202D5        ASCII  "V -VER. ERRORS DURING CONT. OPER."
rF76 45522E2
FF7A 4552524
FF7E 5253204
rF82 5552494
FF86 4720434
rF8A 4E542E2
FF8E 4F50455
FF92 2E
FF93 0D             BYTE  0DH
FF94 0D             BYTE  0DH
rF95 4E4F544        ASCII  "NOTE BUBBLE I/O MAY BE IN ASCII OR HEX@"
rF99 2042554
FF9D 424C452
rFA1 492F4F2
FFA5 4D41592
FFA9 4245204
FFAD 4E20415
FFB1 4349492
FFB5 4F52204
FFB9 455840
     FFF8           ORG    0FFF8H
FFF8 FAEC           WORD  INTERR
rFFA F01C           WORD  START
FFFC FAF1           WORD  INTERR1
```

```
FFFE F000           WORD  BEGIN
     F000           END   BOS
```