# A HARDWARE COMPARISON OF INTERMICROPROCESSOR COMMUNICATIONS TECHNIQUES

**C. Korgel, T. Lohaus, D. Pape**
**Martin Marietta Corporation**
**Denver, Colorado**

## ABSTRACT

A generalized multimicroprocessor utilizing the TI9900 and AMD2901 microprocessors is presented. Different types of microprocessors are used to provide extensive computational capability, versatile interprocessor communications, high reliability, and system flexibility. The system is especially suitable for high speed signal processing, data processing, and data handling. Measured data on the speed of interprocessor communication for three techniques is presented along with a comparative assessment of flexibility and reliability for the techniques.

## INTRODUCTION

Microprocessor systems are currently being proposed for numerous applications because of their potential to provide extensive computational capability, versatile interprocessor communications, high system reliability, and system control flexibility. Many of the systems that have been built are segments of a system such as microprocessor based computers. Some systems have been built that use microprocessors for spacecraft data handling and a large computer for guidance and control computations. Some data handling systems using one type of microprocessor have been completed such as one developed at Martin Marietta for commercial applications using the INTEL 8080 exclusively. Typically, in microprocessor systems currently being designed to perform data handling plus computations different types of microprocessors are being proposed for the two tasks. Examples of this include the Galileo planetary mission spacecraft, and the military systems. It appears that a large percentage of microprocessor systems discussed throughout the industry are in some phase of development. This is especially true for systems using combinations of types of microprocessors and in space system applications. Performance values for these systems are therefore design values rather than demonstrated performance values. In this paper we present the design and some measured data taken from a multimicroprocessor system we have built. We also present comparative advantages of three methods of interprocessor communications. Comparative performance data is given for the TI 9900, AMD 2901, and RCA 1802.

The technical problems in the development of a distributed microprocessor system include coordination and communication between individual microprocessors and the sharing of required functions to produce optimum performance. These problems are compounded when the system contains a large number of microprocessors with different word sizes, operating speeds and architectures. Three interprocessor communication techniques are presented, which attempt to solve these problems. The three interprocessor communication techniques are double latch, multiplexed input/output, and shared memory with service microprocessor. Also presented is a technique to majority vote three microprocessors to increase system reliability. This scheme allows one microprocessor to fail without affecting system operation.

## INTERPROCESSOR COMMUNICATION TECHNIQUES

Both serial and parallel data busses were considered for transferring data between microprocessors. An 8-bit parallel bus approach is desirable. The parallel bus approach is faster and compatible with most microprocessor bus architectures. Processors having 12 or 16 bit words are required to reduce data to be sent to another processor or send two consecutive words. This scheme allows communication between 8, 12, or 16 bit microprocessors. Slower speed and increased hardware requirements are the main disadvantages of a serial data bus.

### Double Latch

One of the simplest interprocessor communication techniques is the double latch method shown in Figure 1. The hardware required consists of two eight-bit data latches and several gates. One latch accepts data from microprocessor No. 1 and passes data to microprocessor No. 2. The other latch accepts data from microprocessor No.2 and passes data to microprocessor No. 1. The latch interrupts the receiving microprocessor when it has accepted data. The software required by the two microprocessors consists of simple read and write routines initiated by interrupts.

The double latch technique is asynchronous and compatible with any word length microprocessor. Any two microprocessors within a system can exchange data if connected together. However, this method would not be suitable for providing interprocessor communication between all microprocessors in a system of three or more. The double latch technique is one of the fastest methods of transferring both single and multiple word data between two microprocessors. Using vectored interrupts data can be read out of the latch immediately.

The double latch technique is reliable because of the low parts count and simple software. The double latch isolates the processors such that if one processor fails the other is not affected.

**Multiplexed Input/Output**

The multiplexed input/output technique for interprocessor communications uses a dedicated data bus and address bus as shown in Figures 2 and 3. The data bus carries data between microprocessors and the address bus contains the address of the microprocessor for which the data is intended. A sequencer provides time periods during which each microprocessor in turn may transmit data to another processor.

The multiplexed input/output technique is a very flexible method. It allows each microprocessor within a system to communicate with every other microprocessor. It is well suited for microprocessor systems containing five or more processors. Up to 100 processors could be interconnected by simple expansion of the sequencer section of the multiplexed input/output hardware. Microprocessors can easily be added to an existing multiplexed input/output system with only minor modifications to the sequencer without affecting existing software. The hardware required by each microprocessor is very simple consisting of only three 8-bit latches and five IC circuits. The software required by each microprocessor is also very simple, consisting of simple read and write routines. The sequencer is hardwired but can be easily modified to provide various allocation schemes for access to the multiplexed input/output busses.

The multiplexed input/output technique is one of the fastest methods of transferring both single and multiple word data between any two microprocessors within a system. Each microprocessor may use the multiplexed input/output busses according to a fixed schedule determined by the sequencer. In a system containing less than one hundred microprocessors, the time each microprocessor has to wait for access to the busses is insignificant. The rate that data can be transferred is mostly dependent on how fast the sending and receiving microprocessors can put data on and take data off the bus. Vectored interrupts are used to inform the receiving processor data is available. If the processors are programmed to remove data as soon as it arrives, fast transfer rates (20 us/word) can be achieved. As an example, in a six microprocessor system a transfer rate of 50K words/second could be carried out between microprocessors 1 and 2, 3 and 4 and 5 and 6, simultaneously.

The multiplexed input/output technique is the most reliable when used in large microprocessor systems. The parts count is appreciably lower and the software for each microprocessor is simple. No system software to synchronize or control the

microprocessors is required. If a microprocessor fails, no other microprocessor would be affected.

**Shared Memory - Service Microprocessor**

A shared memory with service microprocessor concept is shown in Figure 4. Conflicts in accessing shared memory are resolved by the service microprocessor which has the ability to grant any microprocessor access by controlling the bus switches. The bus switches are simple circuits which connect a particular microprocessor's address, data, and control lines to the shared memory. Once a microprocessor has been connected to shared memory the microprocessor itself performs the data transfer with shared memory.

The shared memory with service microprocessor technique is most suited for transferring large blocks of data between a microprocessor and shared memory or between two microprocessors. This technique is very useful in systems where common data is stored for access by several microprocessors. Additional processors can be added to the system by expanding the software of the service microprocessor. Access to shared memory is determined by the program stored in the service microprocessor. Therefore, access schemes can be easily changed or modified. Various schemes may be used such as a priority system, time shared system, a demand system, or a combination of these. These schemes could be changed while the system is operating. The shared memory technique is most useful in systems containing from three to ten processors. Systems with more than ten processors would not have fast access to shared memory and could require complicated software to control access to shared memory.

The shared memory with service microprocessor technique is the slowest of the techniques studied. To transfer data between two processors each processor must request and be granted access to shared memory. Access to shared memory (depending on the access control scheme) may be slow if shared memory is being used by another microprocessor. In the double latch and multiplexed input/output techniques, the sending processor is executing a transmit routine while the receiving processor is executing a receive routine. These two routines are executed simultaneously. But in the shared memory technique these two routines are executed serially.

The reliability of the shared memory technique is lower than the other techniques. This is primarily due to the additional memory (shared) and added microprocessor (service microprocessor) required to perform data transfers.

## Comparison of Interprocessor Communication Techniques

A comparison of interprocessor communication techniques is presented in Tables 1 through 3. The flexibility, throughput and reliability of each are compared under various system configurations. The share memory using arbitrator logic and adjacent memory interprocessor communication techniques have been included. These two techniques have been studied but are not discussed in this paper.

Table 1 makes a general comparison of the usefulness of each interprocessor communication technique in a particular system configuration. In establishing the rating, the following functions were considered: software, throughput, flexibility, expansion, etc.

Table 2(a) makes a general comparison of the throughput of the interprocessor communication techniques studied. Each technique was rated using calculated throughput rates. Table 2(b) presents actual measured data of throughput for the multiplexed input/output, double latch, and shared memory interprocessor communication techniques. These measurements were obtained from the breadboard which contains these three interprocessor communication techniques along with a AMD 2901 and TI 9900 microprocessor. Figure 7 is a redrawn oscilloscope picture of the TI 9900 microprocessor transmitting data to the AMD 2901 microprocessor using the multiplexed input/output technique. Eight bit words are sent over an 8-bit parallel data bus, as they are for the other communication techniques. For comparison purposes Table 2(c) presents calculated throughput rates for the RCA 1802 CMOS microprocessor.

Table 3 makes a general comparison of the reliability of each interprocessor communication technique for a particular system configuration. In establishing the rating, both hardware and software complexity were considered.

## RELIABILITY

Five reliability techniques were studied hardware fail-operate majority vote, synchronous fail operate majority vote, software fail-operate majority vote with feedback, standby redundancy, and various software reliability schemes. Only the hardware fail-operate majority vote technique was incorporated into the breadboard and only it will be discussed further.

## Hardware Fail-Operate Majority Vote

A functional block diagram of a hardware fail-operate majority vote system is shown in Figure 5. The majority vote logic compares the outputs of three independent microcomputer systems. Each microcomputer system contains its own local memory,

clock, and data, address, and control busses. The microprocessors are all performing the same task and store results in an output register. When each microprocessor has loaded its output register the majority vote logic compares the three outputs and selects the contents of a particular output register to be output. The multiplexers are redundant such that if one fails the logic which compares their outputs will detect an error and data will not be output from the system.

Although the microprocessors are all performing the same task they are not required to execute identical software routines. For example, each may be involved in a particular computation but each could be executing a different Algorithm for the solution.

Each microcomputer system is independent in hardware and their outputs do not have to occur at the same time. The majority vote logic will not compare the outputs until at least two microprocessors have output their results. Since each microcomputer system is hardware independent the microprocessors may be different types, have different word lengths and be running on different clock speeds.

The hardware fail-operate majority vote system protects against the following types of failures: microprocessor, memory, data and address busses, clock and software programming errors. This redundancy scheme uses only minimal additional control software. Software is required to start each microprocessor on the next portion of the task after agreement is reached between processors on the results of the previous subtask. This redundancy scheme uses 32 IC’s to implement the required hardware.

The reasons for selecting the majority vote technique for incorporation into the breadboard were as follows: The additional hardware required was about the same as the other schemes investigated, only minimal additional software was required, different microprocessor types could be utilized, by using different algorithms programming errors can be detected, and protection against more types of hardware errors can be realized. Another advantage of this scheme is that hardware or software errors do not have to be detected by the microprocessor themselves. This is accomplished within the majority vote logic which also selects the correct data to be output from the appropriate microprocessor. However, this scheme does require more power, larger size and greater weight because of the three independent microcomputer systems.

## BREADBOARD

The interprocessor communication techniques and reliability concepts which were incorporated in the breadboard are the following: double latch, multiplexed input/output, shared memory with service microprocessor, and hardware majority vote. Other hardware developed and incorporated into the breadboard to support the above concepts were bus

switches, front panel controls with single instruction execution capability, an interval timer, interrupt logic, an ASR 733 terminal interface, and a display panel.

Figure 6 shows the generalized multiprocessor system constructed utilizing available microprocessor technology. It accommodates different types of microprocessors and provides extensive computational capability, versatile interprocessor communications, high reliability and system control flexibility. The system is especially suitable for high speed signal processing, data processing and data handling. The system can be considered for application in place of a central processor, for data bus applications or for implementing special functions within a larger system. A particular application is in a highly accurate and reliable pointing system in a satellite space sextant.

Computational speed, accuracy and power consumptions can be optimized in a system performing a number of functions by using different types of processors for different functions. In the system an Advanced Micro Devices 2901A bit slice processor is used for increased computational speed and accuracy. For functions that do not require bit slice speed but do require high accuracy, an MOS Texas Instruments TMS 9900 16-bit processor is used. The system can accommodate 8-bit processors, such as the RCA 1802 CMOS processor, to provide further power reduction in performing functions requiring only 8-bit words.

System control involves two aspects. One is the control of communications between microprocessors. Communications between microprocessors via a shared memory is under the control of a designated processor. A mode of communication between a large number of microprocessors providing maximum speed is provided by a common bus. For controlling communications between particular pairs of processors, the processors are connected by latches in a predetermined manner. The second aspect of system control is with regard to shared resources that can include memory, I/O devices, special computations processors, etc. The allocation of shared resources is under control of the 2901 service microprocessor. The control software can be written to maximize the shared resources for a particular application.

The distributed microprocessor system was built with standard modular cards featuring high density packaging of dual in-line packages. All interconnections were machined wire wrapped. The layout of components was found not to be critical, therefore, only standard techniques and practices are required.

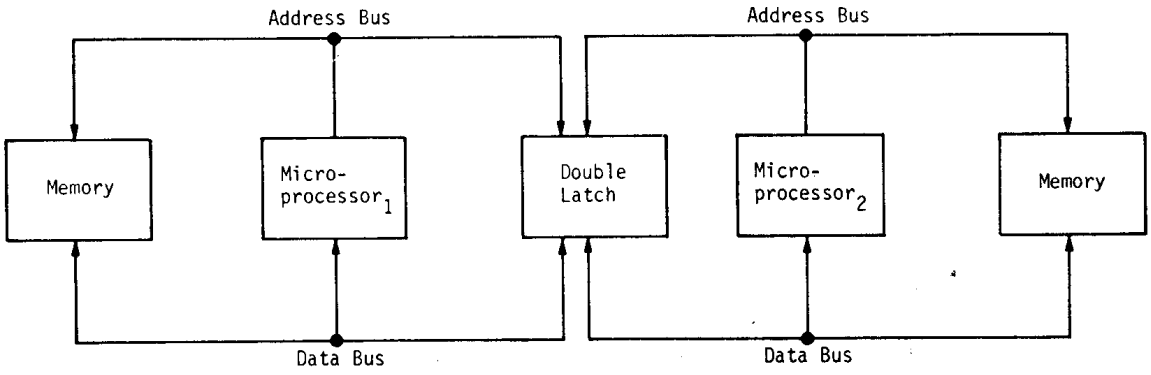


**Figure 1 Double-Latch Block Diagram**

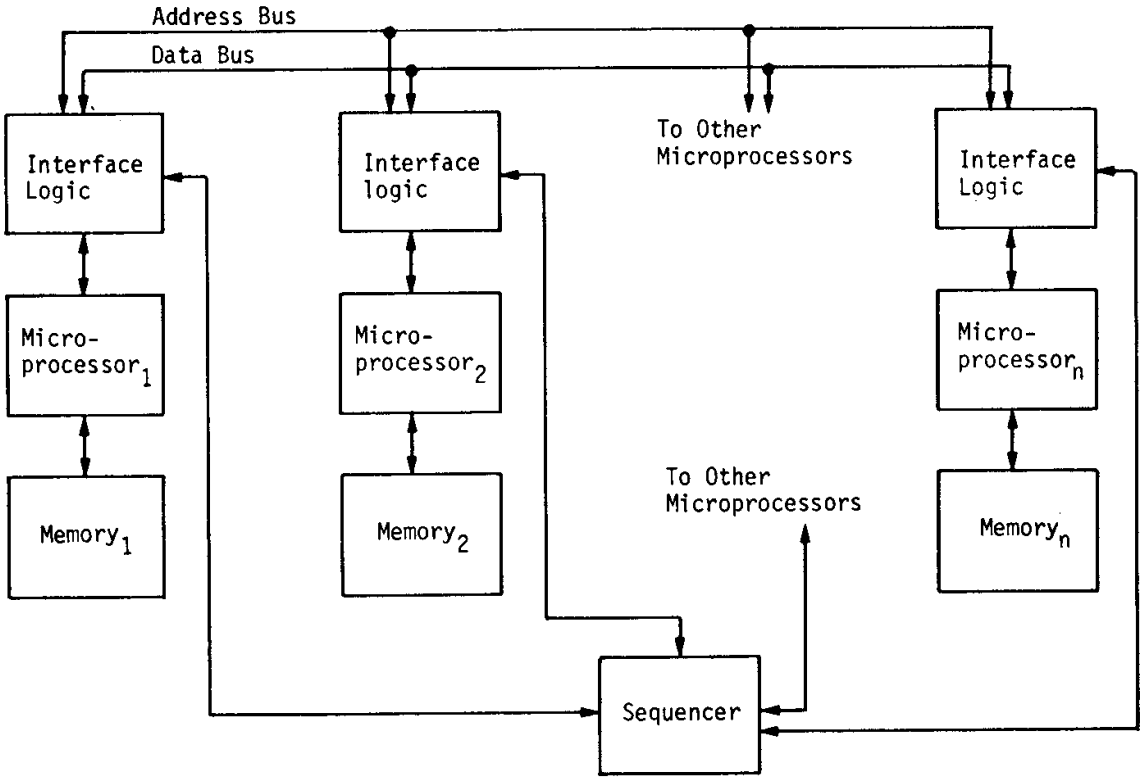


**Figure 2 Multiplexed Input/Output Block Diagram**

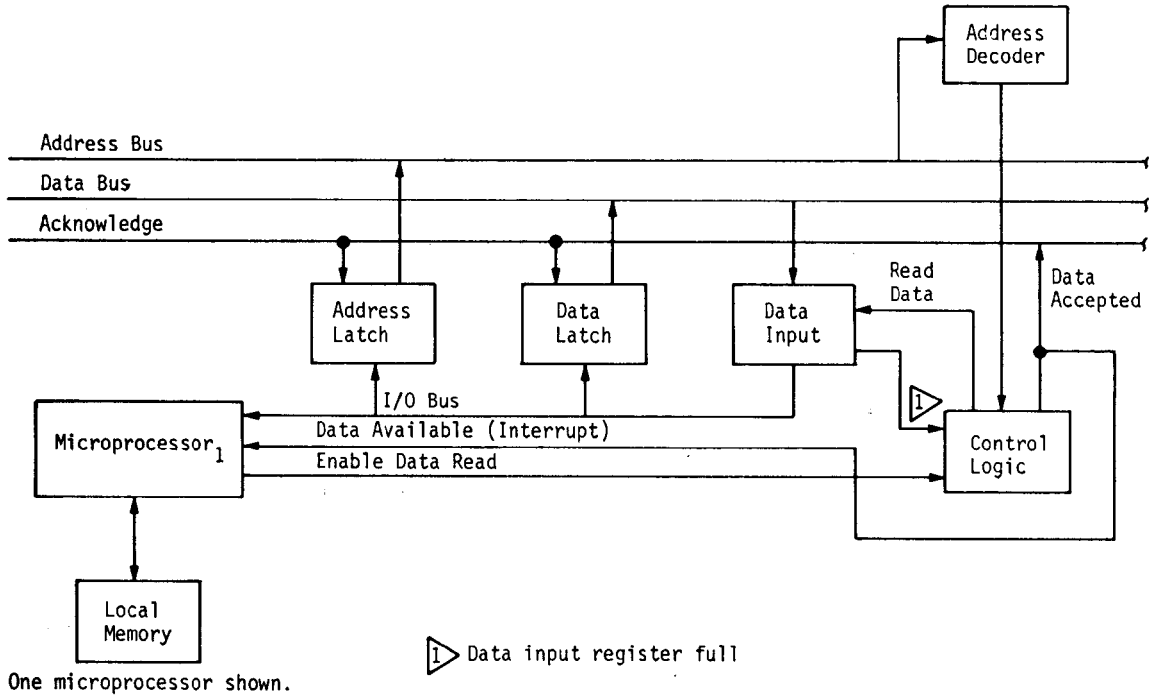


**Figure 3 Multiplexed Input/Output Functional Block Diagram, Transmit and Receive Section**

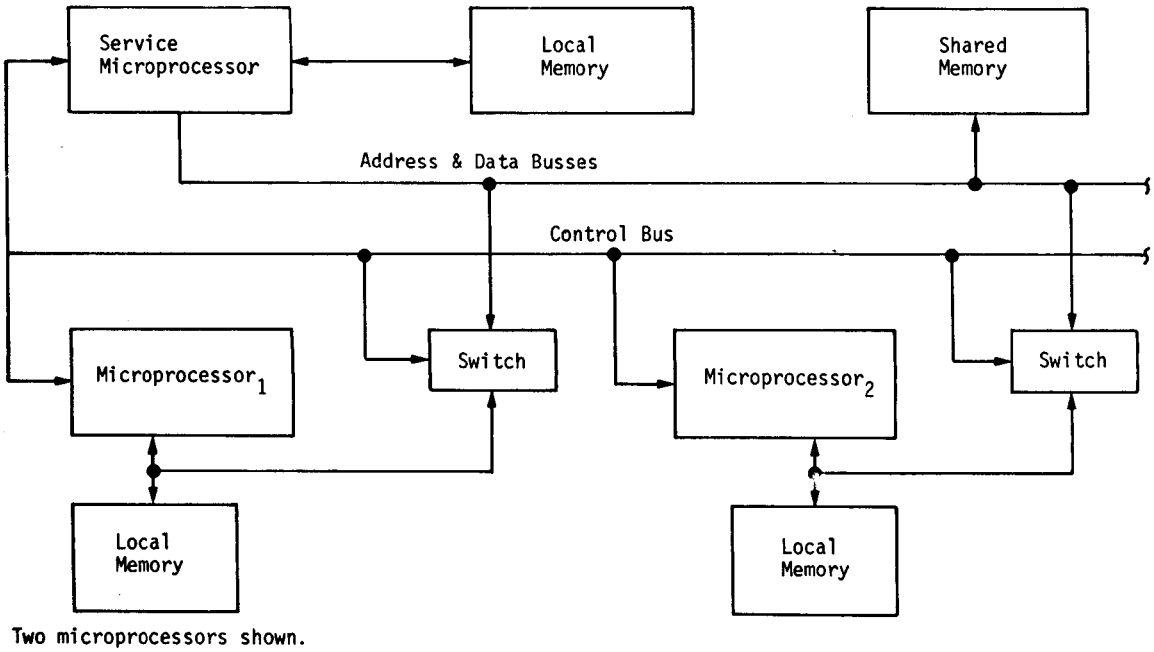


**Figure 4 Shared Memory--Service Microprocessor Block Diagram**

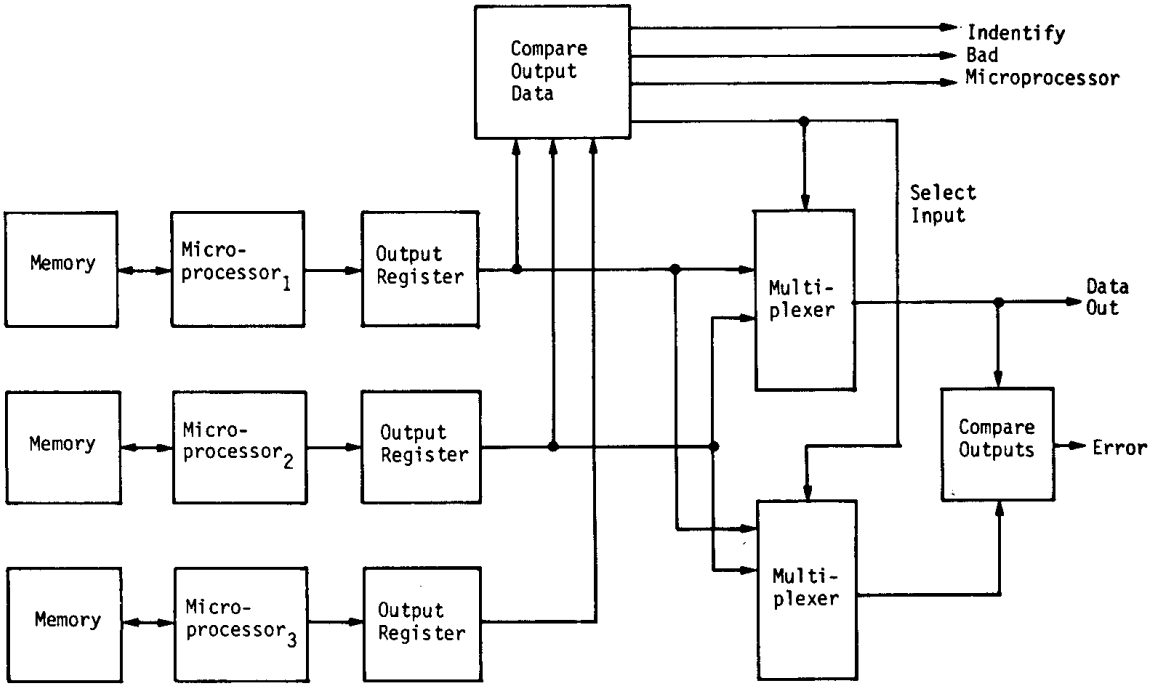


**Figure 5 Hardware Fail-Operate Majority-Vote System**

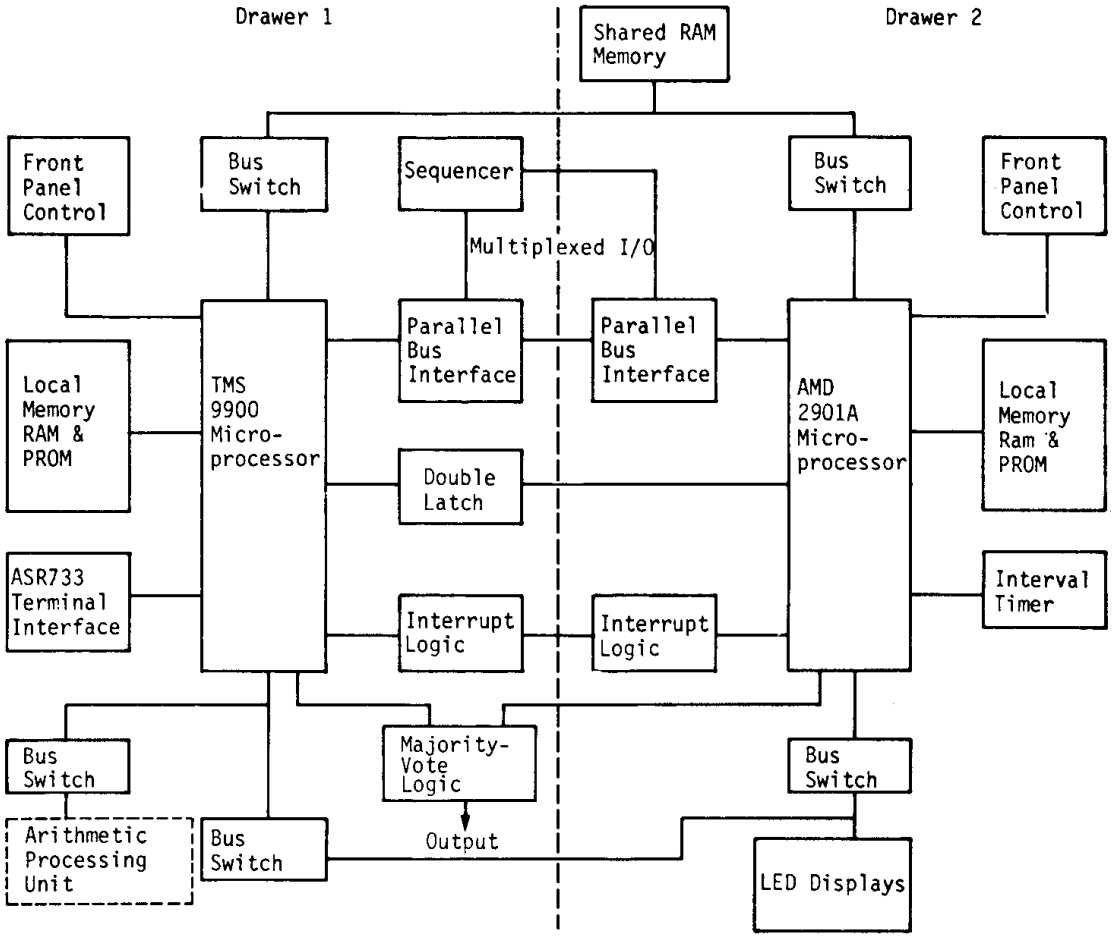


**Figure 6 Distributed Microprocessor Breadboard**

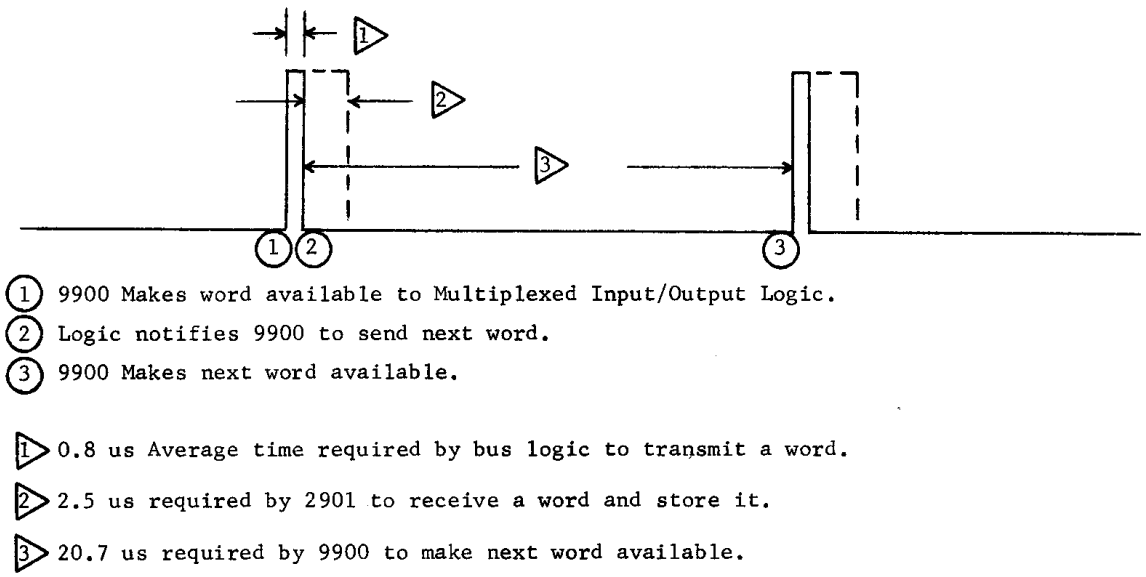


**Figure 7. Data Obtained on Multiplexed Input/Output (9900 Transmitting to 2901)**

**Table 1 Comparison of Adaptability--Interprocessor Cummunication Techniques**

| | Double Latch | Multiplexed Input/Output | Shared Memory | | Adjacent Memory |
|---|---|---|---|---|---|
| | | | Service Micro-processor | Arbitrator Logic | |
| Two-Microprocessor System | Excellent | Poor | Fair | Fair | Excellent |
| Five-Microprocessor System | Poor | Good | Good | Good | Poor |
| Greater than 20-Microprocessor System | Very Poor | Execellent | Poor | Poor | Very Poor |

**Table 2(a) Comparison of Throughput-Interprocessor Communication Techniques**

| | Double Latch | Multiplexed Input/Output | Shared Memory | | Adjacent Memory |
|---|---|---|---|---|---|
| | | | Service Micro-processor | Arbitrator Logic | |
| Transfer one Word between Microprocessors | Excellent | Excellent | Fair | Good | Good |
| Transfer 32 Words between Microprocessors | Excellent | Excellent | Good | Good | Excellent |
| Read 32 Words from Common Memory | N/A | N/At | Good | Good | Excellent |

**Table 2(b) Measured Throughput Rates for TI 9900 and AMD 2901**

1. Multiplexed Input/Output
   9900 Transmitting to 2901 -- 24 us/word (Note 1)
   - 20.7 us required by 9900
   - 2.5 us required by 2901
   - 0.8 us required by bus

2. Double Latch
   9900 Transmitting to 2901 -- 22 us/word (Note 1)
   - 19.5 us required by 9900
   - 2.5 us required by 2901

3. Shared Memory 2901 Transmitting to 9900 -- 231 us 8 words (Note 2)
   - 65 us required by 2901 to load shared memory.
   - 166 us required by 9900 to read shared memory.

NOTE 1: Although not measured this time should apply when the transmission between processors is reversed.

NOTE 2: Calculated Value -- Actual time should equal calculated time ±10% (based on comparisons between calculated and measured times on double latch and multiplexed input/output).

**Table 2(c) Calculated Throughput Rates for RCA 1802**

1. Multiplexed Input/Output
   1802 Transmitting to 2901 -- 18.0 us/word (Note 1)
   - 14.7 us required by 1802
   - 2.5 us required by 2901
   - 0.8 us required by bus

2. Double Latch
   1802 Transmitting to 2901 -- 17.2 us/word (Note 1)
   - 14.7 us required by 1802
   - 2.5 us required by 2901

3. Shared Memory
   2901 Transmitting to 1802 -- 235 us 8 words (Note 1)
   - 65 us required by 2901 to load shared memory.
   - 170 us required by 1802 to read shared memory.

NOTE 1: Although not calculated this time should apply when the transmission between processors is reversed.

**Table 3 Comparison of Reliability--Interprocessor Communication Techniques**

| | Double Latch | Multiplexed Input/Output | Shared Memory | | Adjacent Memory |
|---|---|---|---|---|---|
| | | | Service Micro-processor | Arbitrator Logic | |
| Two-Microprocessor System | Excellent | Good | Good | Good | Excellent |
| Five-Microprocessor System | Good | Excellent | Good | Good | Good |
| Greater than 20-Microprocessor System | Fair | Excellent | Good | Fair | Fair |