interface TECHNOLOGY

Document# 432-2

THE MICROPROCESSOR CONTROLLED DATA AND TIMING GENERATOR

MODEL RS-432, MODEL RSM-432

USER'S GUIDE

INTERFACE TECHNOLOGY
Document Control #432-UG-001
Copyright 1976

Revised: 24 May 1976
16 August 1976
30 April 1980

MODEL RS-432, RSM-432 DOCUMENTATION

This sheet is prepared on a per serial number basis. It describes your specific model number, serial number and configuration. It identifies which drawings are applicable from the drawing package supplied in this manual. It also identifies the applicable options. It is recommended that this page not be removed from this manual and shall be kept by the customer for future reference in cases of service, applications assistance, etc.

Your Model Number is: Model ______________________________

Your Serial Number is: Serial # ____________________________

Applicable sections of this manual are appropriately marked below:

______ Basic RS-432 User's Guide

______ RSM-432 Users Guide Supplement

______ 16 Bit Parallel Interface User's Guide

______ IEEE 488-1978 Interface User's Guide

______ RS-232/TTY Interface User's Guide

______ 8 Bit Parallel Interface User's Guide

______ Card Reader with Interface User's Guide

______ RS-232/C, TTY Interface Drawings/Schematics

______ Dual 8 Bit Parallel 16 Bit Parallel Interface

______ Top Assembly RS-432, RS-432E (Dwg# 10011287)

______ A1, Top Logic Panel, Standard (Dwg# 10011062)

______ A1, Top Logic Panel, RSM (Dwg# 10011062-001)

______ A2, Bottom Logic Panel, Standard (Dwg# 10011061)

______ A2, Bottom Logic Panel, RSM (Dwg# 10011061-001)

______ A3, Front Panel (Dwg# 10011016)

______ A4, Rear Bottom Logic Panel (Dwg# 10011095)

______ 256 Word/Program Memory Card (Dwg# 10011035)

______ 1024 Word Memory Card (Dwg# 10011049)

______ 256/1024 ROM Card (Dwg# 10011007)

______ Output Register Card (Dwg# 10011014)

______ 2K x 16 Word Memory Card (Dwg# 10011334)

______ ALU Card (Dwg# 10011044)

______ Timing Simulator Card (Dwg# 10011043)

______ Input Comparator Card (Dwg# 10011047)

______ Card Reader Interface Card (Dwg# 10011002)
(Also 8 Bit Parallel Interface Card)

______ IEEE 488-1978 Interface Card #1 (Dwg# 10011013)

______ IEEE 488-1978 Interface Card #2 (Dwg# 10011012)

______ RS-232/TTY Interface Card (Dwg# 10011066)

______ RS-232/TTY DC Converter (Dwg# 10011336)

______ Power Supply, Transformer Schematics (Dwg# 2835)

______ Power Supply, Transformer Schematics (Dwg# 6015-S01)

PREPARED BY: ______________________________ DATE: ______________

TABLE OF CONTENTS

TABLE OF CONTENTS

APPENDECIES

LIST OF FIGURES

LIST OF TABLES

WARRANTY

Interface Technology warrants each instrument manufactured to be free of defects in material and workmanship for a period of one year from the date of shipment to the original purchaser. Interface Technology will service, replace, or adjust any defective part or parts, free of charge, when the instrument is returned to Interface Technology freight prepaid, and when examination reveals that the fault has not occurred because of misuse or abnormal conditions of operation. Instruments repaired beyond the effective date of warranty or when abnormal useage has occurred will be charged at applicable rates. Interface Technology will submit an estimate for such charges commencing repair if so requested. For any questions concerning this warranty or shipping, call Interface Technology or our Sales Representative in your area.

PERSONNEL SAFETY

The equipment described in this manual contains voltages hazardous to human life and safety and which is capable of inflicting personal injury.

For all procedures involving component insertion or withdrawal, the equipment must be powered off to prevent component damage. It is also recommended for such procedures that the primary power cord be disconnected from the equipment's rear panel connector to prevent accidental contact with primary power circuits.

While physical measures have been built into the equipment to prevent accidental contact with high voltages during maintenance and troubleshooting procedures, the user should still exercise caution. Careless probing inside the equipment may result in the exposure of high voltage terminals.

Before operating the unit ensure that the primary power outlet includes a functional protective ground (earth) circuit. Do not defeat the unit's internal protective ground circuit to chassis by using a two conductor adapter plug or other such device.

I. INTRODUCTION

The RS-432 Microprocessor Controlled Data & Timing Generator provides the user with a self-contained instrument that can be programmed to generate special-purpose digital test signals. It is a highly flexible digital test system, enabling the user to custom-program all his required test data, as well as required timing, control, and handshake signals.

The User's Guide describes the RS-432 entirely from an operational stand point. It provides programming procedures and specific applications and describes the various options that can be added to the basic RS-432. Organizationally, the Guide starts with a general introduction of the RS-432 from a block diagram stand point. It then proceeds to the descriptions of the various major sections of the RS-432 and provides examples of various programs.

1.1 GENERAL DESCRIPTION

The RS-432 integrates two digital concepts into a single machine. First, it incorporates a high-speed programmable microprocessor that the user programs to generate all his required timing, control, and handshake signals. The microprocessor also provides the overall decision-making and control of the test system. Second, it uses an internal high-speed word generator to store the user's required test data. The word generator receives operating parameters and commands from the microprocessor in such a way that the two sections work together to generate the required test data and control signals.

Figure 1 illustrates the RS-432 architecture. Data is input into the microprocessor program memory and word generator memory from either the front panel or an external device (card, tape reader, etc.). Once the program and word memories are loaded and the machine is started (START PROGRAM switch on front panel is activated), the microprocessor begins executing instructions. These instructions, in turn, generate the required test signals.

The user cannot directly control the word generator from the front panel; the word generator must receive commands from the microprocessor to generate the test data. A program must be written and executed in the microprocessor to generate the required word generator control.

The microprocessor section is the heart of the RS-432. It controls/monitors output and input flags and pulses to and from the user's unit under test (UUT). It executes 16 basic instructions at a rate of 200 nanoseconds per instruction.

The word generator section contains a clock generator that develops the basic output frequency or word rate. It also contains additional logic that controls the output shift registers and generates the various clocks output by the RS-432. The RS-432 can generate as many as 64 parallel bits.

Several options are available to expand the RS-432 capability. The ALU option provides for output pattern generation capability. The timing simulator option allows nonperiodic output rates. The input comparator option provides the RS-432 with the capability of testing input data against data prestored in word memory.

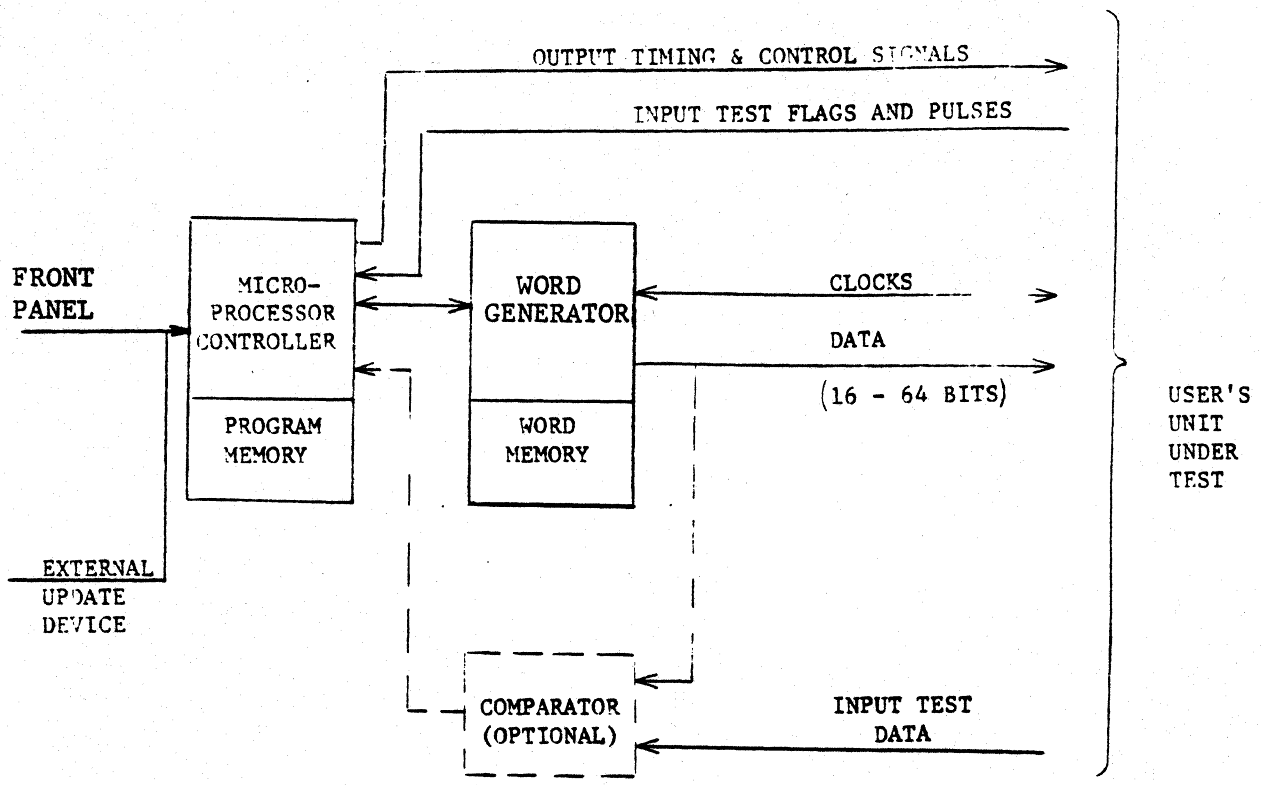


Figure 1 RS-432 BLOCK DIAGRAM

1.2 PHYSICAL DESCRIPTION

The RS-432 is packaged in a metal enclosure. The front panel contains control and data switches to allow manual communication with the internal logic of the RS-432. The back panel houses the input/output connectors and BNC jacks that allow the physical interconnect between the RS-432 and the UUT. Figures 2 and 3 are photographs of the front and back panels respectively.

1.3 SPECIFICATIONS

Table 1 summarizes the general specifications of the RS-432.

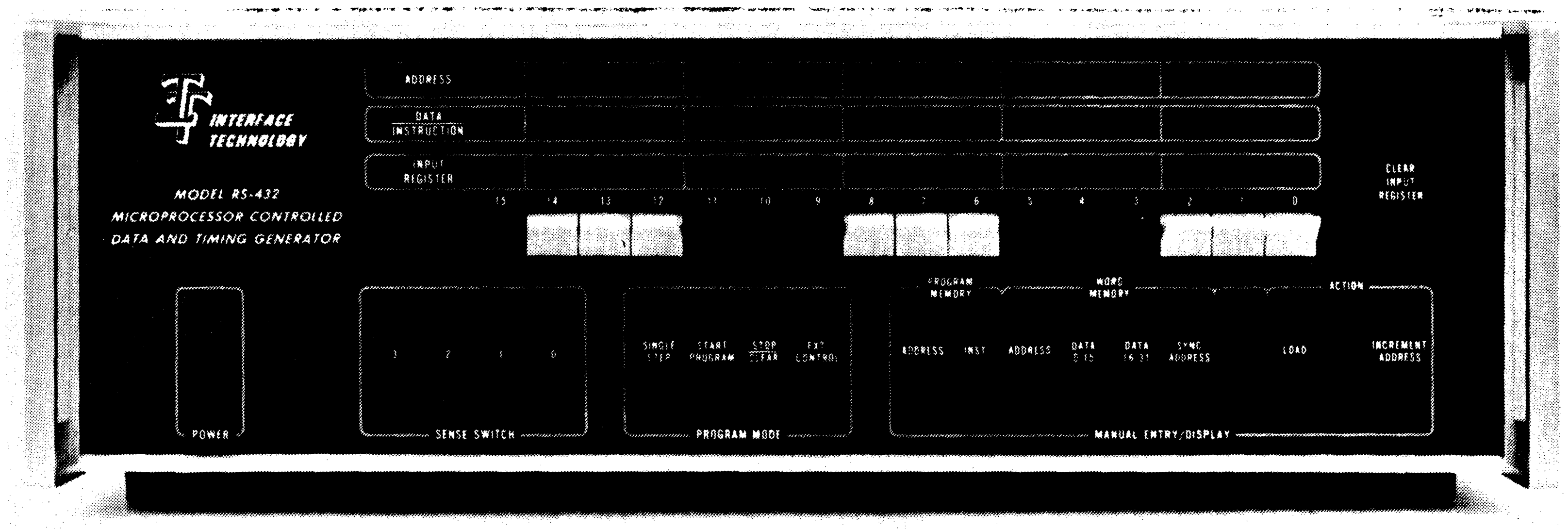


Figure 2. RS-432 MICROPROCESSOR CONTROLLED DATA AND TIMING GENERATOR FRONT PANEL

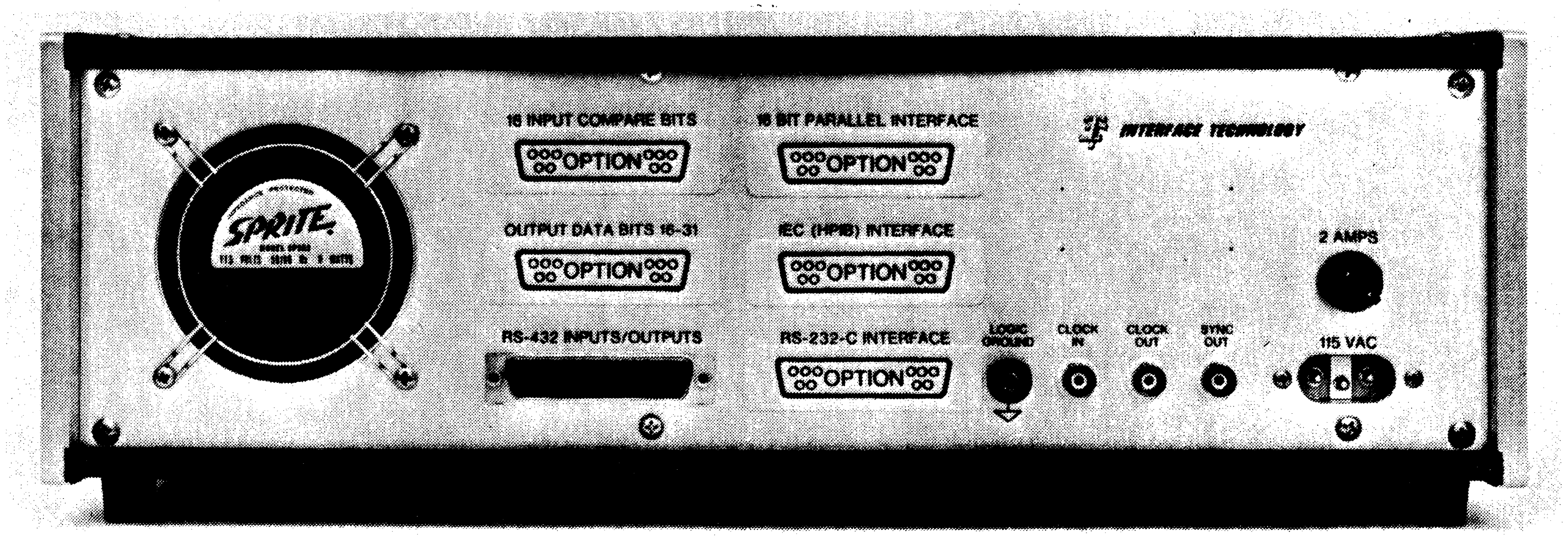


Figure 3. RS-432 MICROPROCESSOR CONTROLLED DATA AND TIMING GENERATOR BACK PANEL

Table 1. RS-432 GENERAL SPECIFICATIONS

MEMORY SIZE:

Program Memory: 64 or 256 words; each word (instruction) is a 16-bit parallel word.

Word (Generator) Memory: Word memory may range from 64 to 4K words, and word size may range from 16 to 64 bits. Refer to the Word Generator Section (Page 37) for a table of allowable configurations.

MEMORY SPEED:

Program Memory: Instruction execution speed is 5 MHz (200 nsec per instruction).

Word (Generator) Memory: Maximum data access time is 10 MHz (100 nsec per word).

OUTPUT SIGNALS:

All output signals are SN74128N 50-ohm TTL line drivers. Each output is capable of driving 30 TTL gates. Each output signal drives a twisted pair wire and is accessible on the back panel connector or BNC connector. Typical rise times (assuming 50-ohm terminated line) for each driver is 3 nsec. Drive characteristics of the SN74128N are presented in Texas Instrument's Data Book.

INPUT SIGNALS:

All input signals are one TTL input load. Each input is a twisted pair wire, accessible on the back panel connector. Input signal low and high limits are -0.5 and +5.5 volts, respectively.

POWER:

115 VAC @ 60 Hz (±10%). Typical dissipation is 80 watts. The AC is fused at 2 amp.

DIMENSIONS AND WEIGHT:

Height: 5.25 inches
Depth: 13.00 inches (extended 22 inches)
Width: 17.00 inches (rack mountable)
Weight: 24 pounds (extended 34 pounds)

OPERATING TEMPERATURE:

Operating temperature range is 0° to 65°C. The RS-432 is cooled by means of a fan mounted on the rear panel.

II. INSTALLATION

Installation of the RS-432 consists simply of rack mounting, if required, and mating to the proper input/output signal connectors.

2.1 MECHANICAL

The RS-432 is shipped ready for use as a stand-alone benchtop signal generator. The folding stand option, if ordered, is mounted and ready for use on the bottom side of the RS-432.

Rack-mounting kit material, if ordered, is packaged with the RS-432. Installation of the rack mounting hardware requires removal of the RS-432 side plates to expose the appropriate holes to which the rack-mount ears can be attached. Slide hardware can also be attached to the side plates if desired.

The RS-432 weighs approximately 24 pounds, with its center of gravity to the left and rear of the enclosure's geometrical center. Appropriate measures should be taken to assure proper mounting within the equipment rack.

2.2 ELECTRICAL

The RS-432 is shipped ready for operation. The enclosed AC cord should be plugged into the 115 VAC 60 Hz receptacle located on the RS-432 back panel. A two-amp 115 VAC fuse is preinstalled in the fuse holder located on the back panel.

All RS-432 input/output signal interconnections are made via the back panel. These I/O signals can be categorized in three groups: (1) those considered to be the standard RS-432 I/O signals, (2) those that are options to the standard RS-432, and (3) those concerned with external loading and control of the RS-432.

Group 1 signals are accessible, via the back panel, through a 78-pin Cannon-type connector and three BNC-type connectors. The 78-pin connector contains all the standard RS-432 input and output signals that are under program control. These signals and their pin assignments are listed in the Appendix. Typically, these signals will be cabled to the user's UUT with a special cable built by the user for his application.

The BNC connectors contain signals that interface with the word generator clock section of the RS-432 - - SYNC OUT, CLOCK OUT, and CLOCK IN. Briefly, SYNC OUT occurs each time the word generator address counter accesses a user defined address (refer to Page 16 for detailed explanation); CLOCK OUT and CLOCK IN are equivalent to GESCK- and GINCK-, respectively, found on the 78-pin connector discussed previously.

Group 2 signals are options to the standard RS-432, and are also available via the back panel through appropriately marked connectors. Signals in this group include additional parallel data outputs and input data compare bits. These signals and their pin assignments are also found in the Appendix.

Group 3 signals are also optional and would be provided for either a 16-bit parallel interface, an RS-232C connector compatible interface (for RS-232C or TTY current loops), an IEEE 488-1975 standard (HPIB) compatible interface, or a mark/sense hand-feed card reader interface. Appropriate connector locations are marked on the RS-432 back panel. Refer to the Appendix for a detailed listing of pin assignments.

2.3 INITIAL CHECKOUT

The following is a short initial checkout procedure that can be performed when the RS-432 is first received.

First, with power off, manually exercise all front panel paddle switches (except the POWER ON switch) to verify that they are operational. Note that some switches are alternate action, whereas others are momentary action.

Next, perform all the steps listed in Table 2. Detailed descriptions of all front panel controls and indicators appear in Section 3.1.

This procedure is a quick check "go/no go" of the RS-432. It verifies that the RS-432 is operational from the standpoint of front panel loading, including the application of AC power, DC power supply, and internal control logic. It does not verify the instruciton execution or I/O data generation capability.

Table 2. INITIAL CHECKOUT PROCEDURE

STEP	ACTION	VERIFY
1	Turn POWER ON switch on	LED above POWER switch should be lit.
1a	Put all sense switches to	
2	Depress STOP/CLEAR switch	LED above START PROGRAM and EXT CONTROL should go OFF.
3	Depress PROGRAM MEMORY ADDRESS switch	LED above switch should be lit. ADDRESS, DATA/INSTRUCTION LED may have several LEDs lit.
4	Depress CLEAR INPUT REGISTER switch	INPUT REGISTER LEDs should be all zero.
5	Depress data toggle switches 0 thru 6	INPUT REGISTER LEDs should be lit in bit positions 0 thru 6.
6	Depress LOAD switch	ADDRESS LEDs should be lit in bit positions positions 0 thru 6 only.
7	Depress PROGRAM MEMORY DATA switch	LED above switch should be lit.
8	Depress LOAD switch	DATA LEDs should be lit in bit positions 0 thru 6 only.
9	Depress INCREMENT switch	ADDRESS LED 7 should be lit, all others should be zero; DATA LEDs may change.
10	Repeat steps 3 thru 9 but replace the PROGRAM MEMORY ADDRESS switch with the DATA MEMORY ADDRESS switch, and replace the PROGRAM MEMORY DATA switch with the DATA 0-15 switch.	

2.4 CALIBRATION

Prior to shipment, the RS-432 has been calibrated and functionally tested. Since the RS-432 is a digital signal generator, calibration is limited to periodic (3 months) verification of the power supply voltage. Refer to the power supply and transformer schematics for location of the various controls referenced below. The necessary adjustments are as follows:

a. Voltage

Connect voltmeter to power supply connections on the lower logic panel. Adjust the power supply voltage to +5.000 +/- 0.050 Vdc.

b. Current Limiting

Connect voltmeter to power supply connections on the lower logic panel. Rotate current limit potentiometer CCW until the output voltage decreases approximately 15 mVdc. Rotate the current limit potentiometer CW approximately 1/8 turn.

In order to troubleshoot the unit, the user should reference the maintenance drawing set which has been configured to each unit's option complement. Verification of the RS-432's fundamental functions may be accomplished by performing the operations of Sections 2.3 and 3.1 of the User's Guide. If the fault cannot be located and corrected, the unit should be returned to Interface Technology.

III. OPERATION

3.1 CONTROLS AND INDICATORS

This section describes the function of the front panel switches. Figure 2 shows the position of the switches on the front panel. It is suggested that the reader become familiar with the microprocessor and word generator before studying this section in detail.

POWER: This alternate-action power switch supplies 115 VAC to the 5-volt power supply.

Lower position = POWER OFF
Upper position = POWER ON (The LED above the switch will light, verifying power on).

INPUT REGISTER AND SWITCHES: The 16 input register momentary action switches labeled 0 thru 15 input data into the input register. Pressing a switch transfers a "1" into the respective register bit which, in turn, lights its associated LED. The 16-bit input register supplies manual input data for the RS-432.

CLEAR INPUT REGISTER: This momentary action switch clears all input register bits to the zero state.

MANUAL ENTRY/DISPLAY: These momentary action switches define which register or memory location is to be updated and/or displayed on the ADDRESS and DATA/INSTRUCTION LEDs. The six switches to the left select the destination register, and the two to the right define the action to be performed on the designated register. Depressing any of the six switches on the left lights the respective LED and turns off the remaining five (i.e., these switches are bailed). Table 3 indicates the data displayed on the ADDRESS and DATA/INSTRUCTION LEDs and the action that may take place for each of the six destination registers.

PROGRAM MODE: These four momentary action switches determine the program mode:

STOP/CLEAR: Pressing the STOP/CLEAR switch performs the following functions:

#1. Stops program execution; START PROGRAM LED will go off.
#2. Stops word generator from sequencing. Word generator can then be started only by executing a start word generator loop (STL) instruction.
#3. Clears external control; EXT CONTROL LED will go off (if on); i.e., gives update control to the front panel.
#4. Clears input pulse falgs.

START PROGRAM: Pressing the START PROGRAM switch starts program execution. The START PROGRAM LED lights, and program starts at the present address stored in the PMA register.

EXT CONTROL: Pressing the EXT CONTROL switch enables an external device (computer, card reader, etc) to load, update, and control the RS-432. The EXT CONTROL LED turns on. Programming from the front panel is inhibited, although the six register display LEDs can still be selected to display their respective data. Note that an external device can also request and gain control without the depressing of the EXT CONTROL switch. The EXT CONTROL led can only be exstinguished with STOP CLEAR switch.

SINGLE STEP: If both the START PROGRAM and EXT CONTROL LEDs are OFF, pressing the SINGLE STEP switch will execute one program instruction. By continual pressing, the generator can single-step through the entire program.

If one of the instructions executed during this Single Step process is an STL instruction, the word generator will receive one clock pulse for each activation of the SINGLE STEP switch. Simultaneously, the program will also execute one instruction for each activation of the SINGLE STEP switch. If the instruction sequence is a start word generator followed by a halt until the word generator is complete instruction, the operator could literally spend hours attempting to single step through the word generator if the generator required many clock pulses to complete its task. This effort can be bypassed by pressing the STOP/CLEAR switch, which will reset the Word Generator Busy flag. Continued pressing of the SINGLE STEP switch will then continue to Single Step through the program.

SENSE SWITCHES: These alternate-action switches enable on-line communication with the program. Each switch is debounced and synchronized with the program to eliminate erroneous inputs. The jump-on sense switch (JSS) instruction can then test each sense switch for the upper or lower position. Some sense switches are uses for display purposes as well as program control.

3.1.1. Front Panel Loading

The following description provides step-by-step procedures for loading both the program memory and word memory. Program Example 4 (page 58) is used as input data.

3.1.1.1. Program Memory Loading Procedure

Step	Action
1	Press STOP/CLEAR switch.
1a	Put all sense switches to lower position.

NOTE: Steps 2 thru 5 provide the "fetch" program memory address function.

Step	Action
2	Press PROGRAM MEMORY ADDRESS switch. This selects the program memory address counter as the destination register.
3	Press CLEAR INPUT REGISTER switch.
4	Input into the Input Register (press appropriate data switches for these bit positions which are to be a "1") the first program memory address location. For Program Example 4, all zeroes will be the input data (i.e., do not press any input register switches).

Step Action

5 Press the LOAD switch. This action loads the Input Register data into the program memory address counter (i.e., program address 0 has been "fetched").

NOTE: Steps 6 thru 9 will load the first instruction into the program memory.

6 Press the PROGRAM MEMORY INST (instruction) switch. This selects the program memory as the destination register. The particular program memory word that is the destination register is the one at the location defined by the program memory address counter being displayed on the ADDRESS lights (in this case, address 0).

7 Press the CLEAR INPUT REGISTER switch.

8 Input into the input register the desired program instruction data for program address 0. For Program Example 4, this would be as shown below:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0
DEP				1 usec		Value 10									

9 Press the LOAD switch. The data in Step 8 will appear on the DATA/ INSTRUCTION lights.

NOTE: Steps 10 thru 13 will load the second instruction into the program memory

10 Press the INCREMENT ADDRESS switch. The program memory address counter will increment by 1, as can be verified by observing the ADDRESS lights.

11 Press the CLEAR INPUT REGISTER switch.

12 Input the next program instruction data into the input register.

13 Press the LOAD switch

NOTE: Step 14 will load the remaining memory locations.

14 Repeat Steps 10 thru 13 until all desired program memory locations are loaded. For this example, the last program memory location to be loaded is at program address 6 (00000110). Note that unused program memory locations need not be cleared out; i.e., they are "don't care" locations.

3.1.1.2 Word Memory Loading

Step Action

1 Press the STOP CLEAR switch.

NOTE: Steps 2 thru 5 provide the "fetch" word memory address function.

2 Press the WORD MEMORY ADDRESS switch. This selects the word memory address counter as the destination register.

3 Press the CLEAR INPUT REGISTER switch.

Step	Action
4	Input into the input register (press appropriate data switches for the bit positions that are to be a "1") the first word memory address location.
5	Press the LOAD switch. This function loads the input register data into the word memory address counter.
NOTE:	Steps 6 thru 9 will load the first data word into the word memory.
6	Press the WORD MEMORY DATA 0-15 switch, which selects the word memory data bits 0-15 as the destination register. However, since there may be 4K memory words, the particular word that is the destination register is the one at the location defined by the word memory address counter (which is now displayed on the ADDRESS lights).
7	Press the CLEAR INPUT REGISTER switch.
8	Input into the input register the desired word data for the first data word.
9	Press the LOAD switch. Input data will appear on the DATA/INSTRUCTION lights.
NOTE:	Steps 10 thru 13 will load the second data word into the word memory.
10	Press the INCREMENT ADDRESS switch. The word memory address counter will increment by 1, as can be verified by observing the ADDRESS lights.
11	Press the CLEAR INPUT REGISTER switch.
12	Input the next data word into the input register.
13	Press the LOAD switch.
NOTE:	Step 14 will load the remaining program memory location.
14	Repeat Steps 10 thru 13 until all desired word memory locations are loaded.

3.2 INPUTS AND OUTPUTS

This section describes the RS-432 signal inputs and signal outputs, exclusive of the memory load interfaces such as the 16-bit parallel interface, RS-232C compatible interface, etc. It describes, in general terms, how these various signals are controlled. For a detailed description and explanation of these signals, refer to Section IV.

Figure 4 is a simplified diagram depicting the source and termination points for all inputs and outputs to be discussed in this section. All inputs present one TTL unit load to the UUT, whereas all outputs are driven from a 74128-type, 50-ohm line driver capable of driving 30 TTL unit loads.

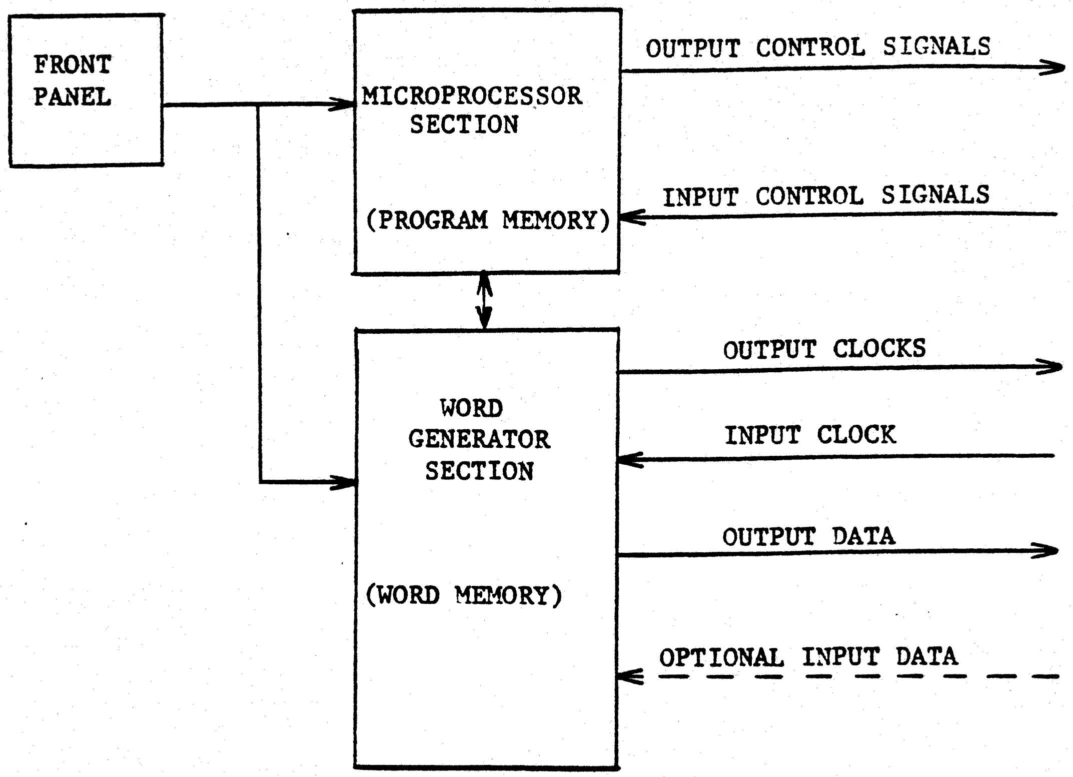


Figure 4. SIMPLIFIED I/O BLOCK DIAGRAM

3.2.1 Output Control Signals

The microprocessor section directly controls two types of output control signals: eight output flags and two output pulses. The states of these ten lines are simultaneously controlled by a single microprocessor instruction. This instruction sets ("1") or resets ("0") the output flag lines, and can set, reset, pulse low, or pulse high the output pulse lines. Setting or resetting any flag or pulse line can occur as fast as 200 nanoseconds. All pulse lines have durations of 100 nanoseconds. Refer to Section 4 for detailed explanation of the generate output flags (GOF) instruction.

These output control signals are available on the standard 78 pin back panel connector. Pin assignments are indicated in the Appendix.

3.2.2 Input Control Signals

The microprocessor section can directly sense two types of input control signals: four input flag lines and four input pulse lines. These input control signals are available on the standard 78-pin back panel connector. Refer to the Appendix for pin assignments. Quanitity may vary because some input control signals are used for options.

A microprocessor instruction can check the states of the four input flag lines for their set or reset conditions. External devices under test can use these lines to indicate a variety of conditions to the microprocessor section.

The input pulse lines feed latches internal to the microprocessor section. The microprocessor monitors latch outputs for pulse occurrence or absence. After interrogation, the microprocessor resets the particular latch that has been tested. Refer to Section 4 for a detailed explanation of the jump-on input flag (JIF) and jump-on input pulse (JIP) instructions.

3.2.3 Output Clocks

Several clocks are available at the RS-432 back panel: the free-running clocks, gated clocks, and the sync signal. These clocks are generated by the word generator section.

3.2.3.1 Free-Running Clocks

The user has access on the back panel 78-pin connector to both a free-running clock (FEXCK+) and an inverted free-running clock (FEXCK-) (Figure 5). Both clocks are square waves, with a frequency defined by the period counter in the word generator.

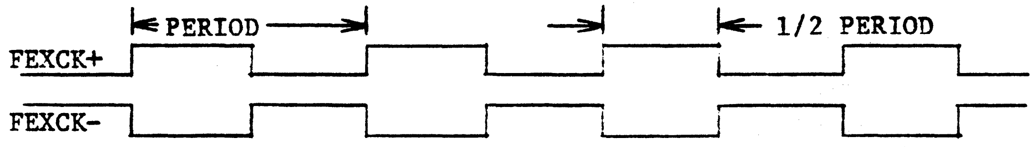


Figure 5. FREE-RUNNING CLOCKS

The plus sign on the signal mnemonic for the clock indicates that the output register will load new data or shift data when the FEXCK+ clock goes from low to high. The minus sign indicates that data will change when FEXCK- goes from high to low (refer to Figure 6).

3.2.3.2 Gated Clocks

Two gated clocks (GEXCK+, GEXCK-) are also available on the back panel 78-pin output connector. These clocks are precisely synchronized with the free-running clocks, but they are gated on only when the word generator has been started by a start word generator loop (STL) instruction. They remain running until the word generator completes the transfer of the data block and the Word Generator Busy flag is reset. The gated clocks enable the user to transfer test data from the word generator into his system or UUT.

Figure 6 illustrates the relationship of the free-running clocks, gated clocks, and the word generator output data. The timing shown assumes the following:

1. First word memory address = 0
2. Last word memory address = 3
3. Parallel data transfer
4. Period counter = 10 microseconds
5. The word generator received an STL instruction that selects the period counter clock as the clock source and requests a loop count of 1.

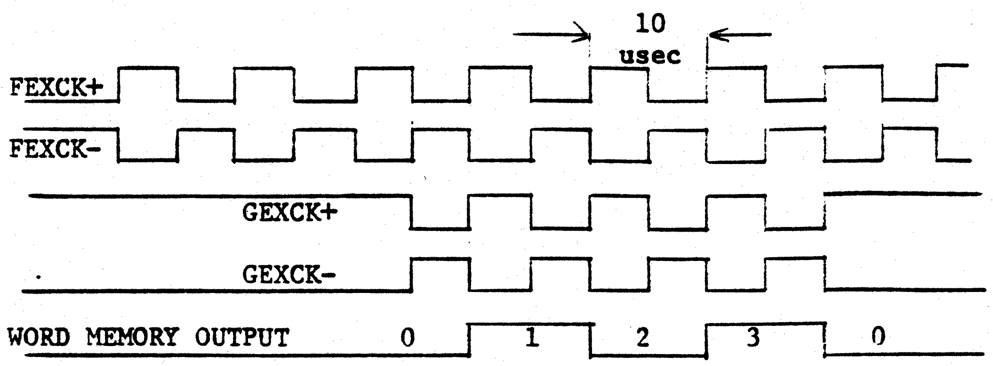


Figure 6. OUTPUT CLOCKS AND DATA RELATIONSHIP

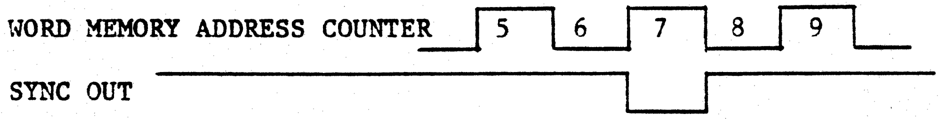
Again, note that the plus sign on the clock mnemonic indicates that the word data will change coincident with the low-to-high transition, whereas the minus sign indicates that the word data will change coincident with the high-to-low transition.

Note that the signal CLOCK OUT, which is available at a back panel BNC-type connector, is identical to the gated clock GEXCK- on the 78-pin I/O connector. However, it is rebuffered. Figure 6 shows the timing relationship.

3.2.3.3 Sync Signal

The SYNC OUT signal available at a BNC-type connector on the back panel generates a sync pulse whenever the word memory address counter equals the word memory sync address register. This register can be loaded by activating the front panel WORD MEMORY SYNC ADDRESS switch in conjunction with the LOAD switch (see Controls and Indicators description, Section 3.1).

EXAMPLE: Assume the WORD MEMORY SYNC ADDRESS register equals 7 (000 000 000 111).



3.2.4 Input Clock

The gated input clock GINCK- enables an external device to clock the word generator. The word generator fetches a new word or shifts the output register on the negative edge (high-level to low-level transition ‾_) of each clock.

As shown in Figure 7, GINCK- must be in the low (zero) state when clocks are not generated to ensure that the RS-432 and the input clock will be synchronized on the first input clock, after the STL instruction. Otherwise, erroneous data may result for the first output word. All subsequent output words will be synchronized with GINCK-.

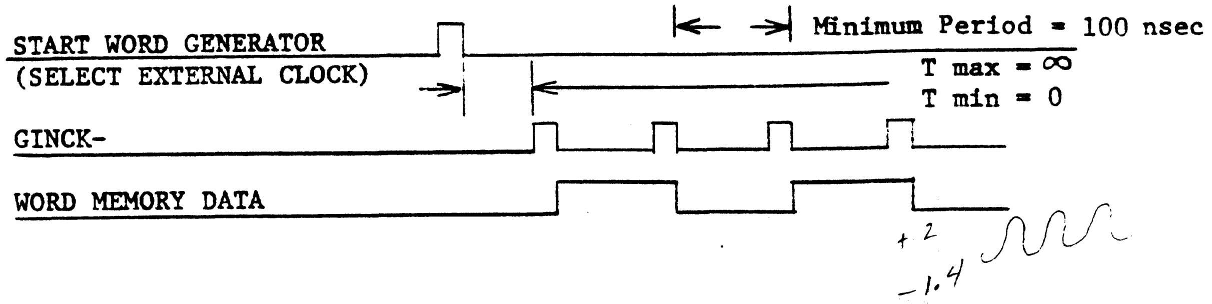


Figure 7. INPUT CLOCK RELATIONSHIP

Figure 8 portrays the input clock to output data timing relationship.

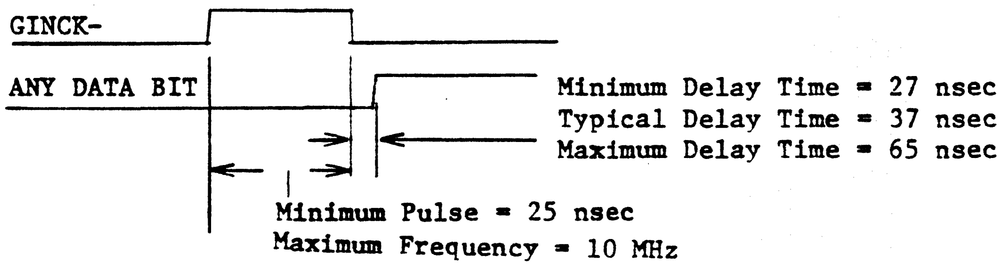

GINCK-
ANY DATA BIT
Minimum Delay Time = 27 nsec
Typical Delay Time = 37 nsec
Maximum Delay Time = 65 nsec
Minimum Pulse = 25 nsec
Maximum Frequency = 10 MHz

Figure 8. INPUT CLOCK/DATA OUT TIMING RELATIONSHIP

Note that the CLOCK IN signal available at the back panel BNC-type connector is identical to the input gated clock GINCK- on the 78-pin I/O connector. It is logically "ORed" with GINCK-. Thus, the same rules apply for this clock as for the input clock GINCK- discussed in the previous paragraph.

The CLOCK OUT and CLOCK IN signals permit two RS-432's to be run in parallel: for example, if 128 parallel data bits are required, two RS-432's with 64 parallel data bits each can be tied together, yielding 128 fully synchronous parallel data bits. This is accomplished by using one RS-432 as the master generator and the other as the slave. The master RS-432 starts its word generator selecting its internal period counter, and the slave starts the word generator selecting an external clock. The CLOCK OUT of the master RS-432 is then connected to the CLOCK IN of the slave RS-432. Note that a program "handshake" between the two machines must also be performed to ensure that the slave RS-432 is ready to transmit its data. This handshake is accomplished by using a flag from the slave machine to inform the master machine that it has started its word generator and is awaiting input clocks.

3.2.5 Output Data

This set of output signals consists of 16 parallel signal lines representing the data stored in an output register internal to the word generator section. The output register is fed from the word generator word memory in which user-defined data patterns or words have been prestored. Since the output register is also shift register, output data can be considered either parallel or serial. Serial shifting and parallel loading are controlled within the word generator via instructions from the microprocessor.

Available as options are additional output registers that can increase the total number of parallel output bits from 16 to 64. Note that standard on all RS-432's are all I/O control signals discussed in Sections 3.2.1 through 3.2.4, along with 16 output data lines. Output data lines 32 through 64 are optional. These additional data bits are also accessible via back panel connectors. Refer to the Appendix for their pin assignments. Refer to Figure 6, Figure 10, and Section 3.2.7 for timing relationships of these output signals.

3.2.6 Input Data

This set of optional input data bits consists of 16 or 32 parallel input data lines. These lines are routed internally to a set of digital compare circuits that indicate compare/no compare to values previously stored in the word generator memory. This input data is typically provided by the UUT in response to data and control signals previously provided by the RS-432. Bit masking capabilities exist to allow masking of input bits not pertinent to the go/no go comparison being made. Masking can be performed either manually via switches mounted on the input circuit card or dynamically via programmed microprocessor instructions.

Note that this input data section is an option to the basic RS-432. Check your specific RS-432 configuration for applicability. This option is provided with the comparator output wired to one of the input flag line, thus allowing the microprocessor to check the compare operation. Refer to Section 4.3 for a full explanation of the input compare option.

3.2.7 Output Timing Relationship

Figure 9 shows the timing relationship of the rising edges of FEXCK+ and GEXCK+ with respect to any data from the word memory. The falling edges of FEXCK- and GEXCK- have similar timing delays to those of FEXCK+ and GEXCK+.

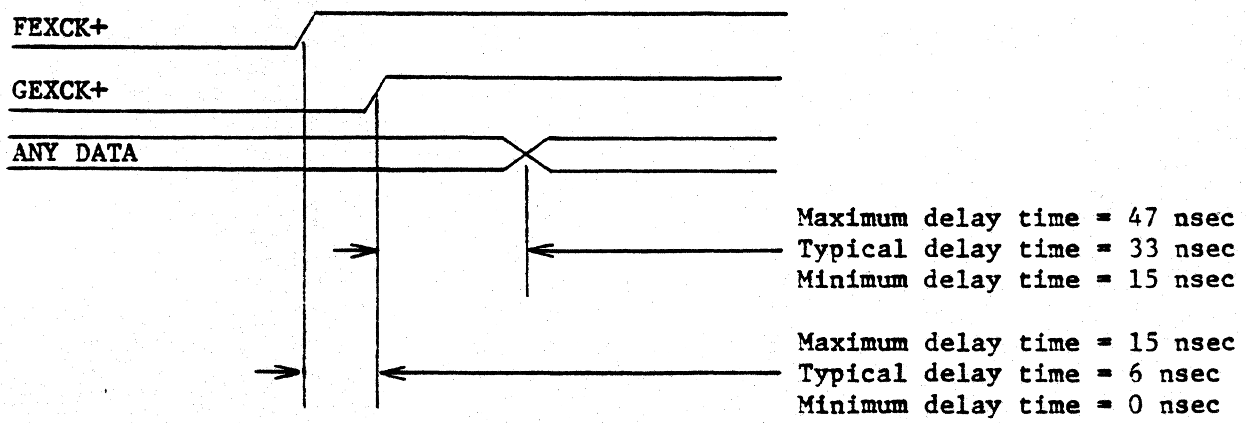

FEXCK+
GEXCK+
ANY DATA
Maximum delay time = 47 nsec
Typical delay time = 33 nsec
Minimum delay time = 15 nsec
Maximum delay time = 15 nsec
Typical delay time = 6 nsec
Minimum delay time = 0 nsec

Figure 9. CLOCK/DATA TIMING RELATIONSHIP

Figure 10 shows the output timing relationship between any two data bits.

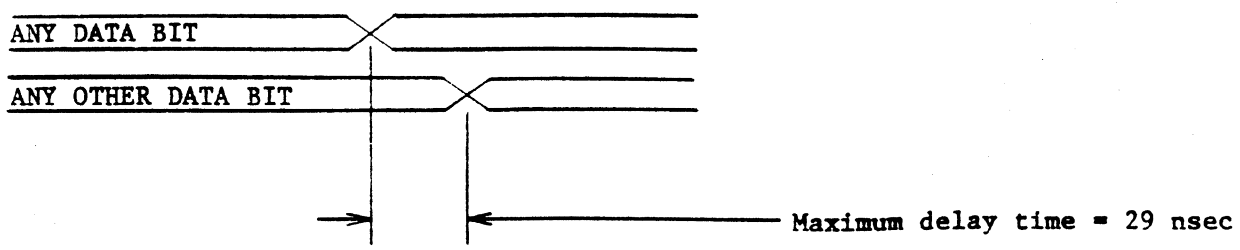


Figure 10. DATA TIMING RELATIONSHIP

3.2.8 Logic Ground

The logic ground signal is wired to the ground plane of the logic boards internal to the RS-432. This ground signal is <u>not</u> tied to the RS-432 chassis ground. The user may tie logic ground to chassis ground if desired.

3.3 OPERATING PROCEDURE AND EXAMPLE

This section discusses operation of the RS-432 from the practical, "how to" application standpoint. It is presented to provide an overall view of the RS-432 and its application to any signal generation test problem. The section concludes with an example that can be loaded into and executed on any standard RS-432. Details of progamming and circuit operation are covered in Section IV.

3.3.1 Operating Procedure

Operation of the RS-432 involves several key steps: (1) problem definition and assignment of I/O signals, (2) program definition and writing, (3) program loading and execution (machine operation), (4) program debug/signal analysis, and (5) program documentation. Sequential progression through these steps assures a high probability of success with the least investment of time.

Problem definition yields such information as number of control signals and their timing requirements, number of parallel or serial output bits, output data rates, number of word memory locations required, where in word memory the data will reside, and other details required for programming. This information is used to assign the RS-432 inputs and outputs to the signal lines at the UUT with which they interface. This allows construction of a cabling arrangement to mate the RS-432 to the specific UUT. It also defines the signals that must be controlled by the microprocessor and word generator sections of the RS-432.

The second step, program definition, started during problem definition, where decisions were made concerning where and how the various signals were to be controlled. This step entails determining how the RS-432 program actually effects control of the I/O signals. Flow charting is a useful tool in finalizing the program definition in that it allows visualization of what steps the microprocessor must go through to accomplish its given task. The completed flow chart forms the skeleton of the detailed program that can then be written. The net result of a written program is normally a coding sheet filled with the machine language "1" and "0" that must be loaded into the RS-432. This step also involves the generation of data or data patterns to be stored in the word memory.

Program loading, which consists of the transfer of machine language program into program memory, can be accomplished either manually via the front panel or under external control from a card reader, TTY, etc. (For manual entry refer to Section 3.1, Controls and Indicators, which describes the manual operation of the RS-432.) Upon completion of program loading, the program memory and word memory are loaded with their respective instructions and await execution.

Program execution and, thus, starting of the RS-432 signal generation, consists of placing the program memory address counter at the proper starting program memory address and depressing the START PROGRAM control switch. Depressing the STOP/CLEAR switch will immediately halt the RS-432, and clear all input pulse latches. The RS-432 can be started from this stopped state by depressing the START switch again.

Program debugging is aided by use of the SINGLE STEP switch and front panel LED readouts. The STEP switch allows single instruction steps to be taken with each switch activation for checking program logic and sequencing. Erroneo instructions or data words can be modified via the front panel.

Finally, when all signal generation tasks are accomplished, final program instructions, word memory patterns, and input/output waveforms must be documented. Interface Technology provides coding sheets to assist in the program development and documentation. These sheets appear in the Appendix and are used throughout this manual as examples.

3.3.2 RS-432 Example

The following example is presented to further acquaint the user with the operating procedure for the RS-432. Even though detailed explanations of the RS-432 program instructions have not been provided, the user can still use the coding sheets provided in this example, load the RS-432 program and word memory, and execute this example. His example also affords insight into the RS-432's capabilities, and is executable on any standard configuration RS-432.

3.3.2.1 Problem

Continuously output five 16-bit parallel words at a 300 millisecond-per-word rate if sense switch 0 is set. Set a control signal to indicate that the 300 millisecond data is being generated.

3.3.2.2 I/O Signals

In this example, the 16 parallel bits will be generated on BIT00+ through BIT15 and the output control signals on FLG00+ and FLG01+. As a result of this program's execution, the following signals will also be generated:

1. Free running clock FEXCK+, FEXCK-
2. Gated clock GEXCK+, GEXCK-

When executing, the word generator can also generate the SYNC OUT signal at one of the five word memory addresses. Figure 11 depicts the RS-432 and the various output signals. Refer to the Appendix for the pin assignments of these signals.

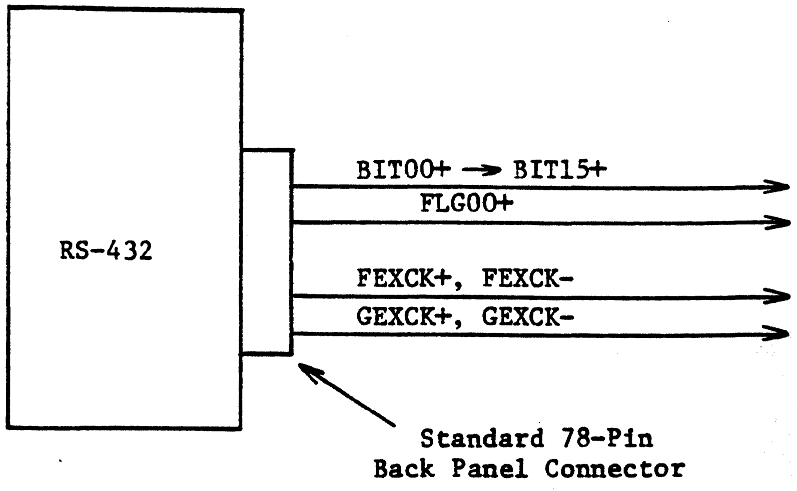


Figure 11. RS-432 I/O SIGNAL DIAGRAM FOR EXAMPLE

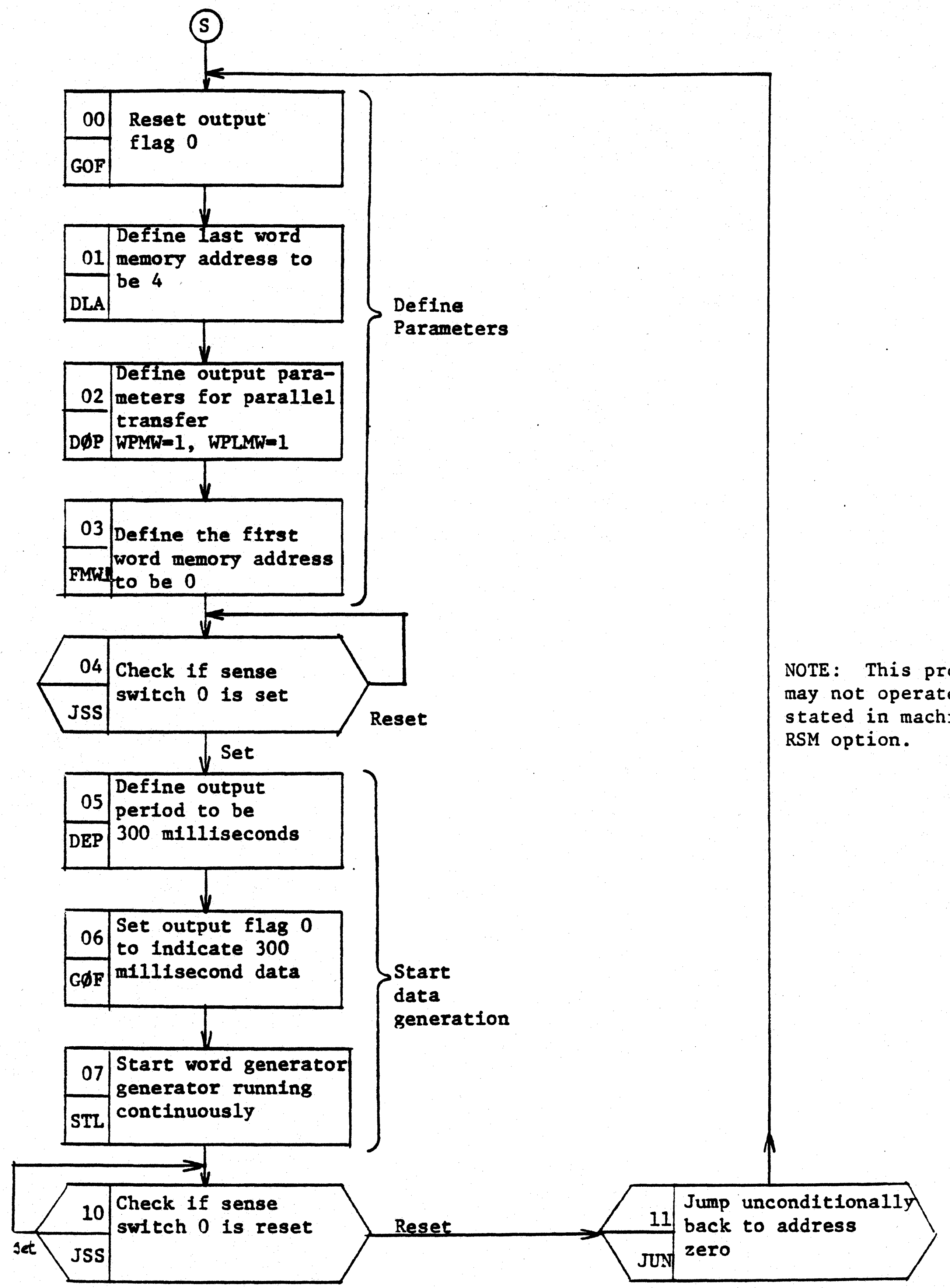
Refer to Figure 12 for a simplified flow chart of this example.

3.3.2.3 Program Coding Sheet, Program Loading and Execution

Refer to Tables 4 and 5 for the program memory and word memory data coding sheets, respectively. By following instructions set forth in Section 3.1, Controls and Indicators, along with the program memory coding sheet and the word memory coding sheet, the RS-432 can manually be loaded. Once loaded, place the program memory address to location zero, reset all sense switches, and depress the START PROGRAM control switch.

The program will now be running and will be checking the sense switch; no data is being generated as the sense switch is not yet set. Setting Sense Switch 0 (SS0) and depressing the WORD MEMORY DATA 0-15 manual entry/display switch should allow visual verification that the five memory words are being generated at a rate of 300 milliseconds per word. Resetting SS0 will stop the data transfer. The various output signals can be monitored on an oscilloscope. If the Sync Address register is loaded with the value 000, a synchronizing pulse occuring once every five words at word memory address 0 can be used to synchronize the oscilloscope. Note that at 300 milliseconds per word, one should be able to verify on the DATA LEDs the output pattern of Table 5.

Program Example 7: Parallel Data Transfer, Page 63, is a follow-up to this example. In it, the microprocessor will check two sense switches and generate data at one rate for Sense Switch 0, or generate data at a faster rate for Sense Switch 1. Program Example 7 should be programmed in lieu of this example if faster data rates are desired for oscilloscope monitoring of the output signals.


S
00
GOF
Reset output flag 0
01
DLA
Define last word memory address to be 4
02
DØP
Define output parameters for parallel transfer WPMW=1, WPLMW=1
03
FMWL
Define the first word memory address to be 0
Define Parameters
04
JSS
Check if sense switch 0 is set
Reset
Set
05
DEP
Define output period to be 300 milliseconds
06
GØF
Set output flag 0 to indicate 300 millisecond data
07
STL
Start word generator generator running continuously
Start data generation
10
JSS
Check if sense switch 0 is reset
Set
Reset
11
JUN
Jump unconditionally back to address zero

NOTE: This progr
may not operate a
stated in machine
RSM option.

Figure 12. RS-432 FLOW CHART FOR EXAMPLE

DATE:______________

SHEET _____ OF __________

Table 4: RS-432 PROGRAMMING SHEET (CODING)

PROGRAM NAME: RS-432 USER'S GUIDE EXAMPLE

					INSTRUCTION																	
ADDRESS OCTAL					MNEMONIC	OP CODE				DATA FIELD												COMMENTS
						15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
			0	0	GØF	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
			0	1	DLA	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	DEFINE PARAMETERS
			0	2	DØP	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	
			0	3	FMWI	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	
			0	4	JSS	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	CHECK SENSE SWITCH
			0	5	DEP	1	0	0	0	0	0	0	1	0	0	1	0	1	1	0	0	
			0	6	GØF	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	START 300MS DATA
			0	7	STL	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	
			1	0	JSS	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	
			1	1	JUN	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	JUMP BACK
			1	2																		
			1	3																		
			1	4																		
			1	5																		
			1	6																		
			1	7																		
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		

Date:____________

Sheet______ of ______

Table 5. **RS-432 WORD MEMORY PROGRAMMING SHEET**

PROGRAM NAME: RS-432 USER'S GUIDE EXAMPLE

ADDRESS (OCTAL)						DATA																COMMENTS
						15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
			0	0		0	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	
			0	1		0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	
			0	2		1	0	1	1	0	0	0	1	1	0	0	0	1	1	0	1	DATA
			0	3		0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	PATTERN
			0	4		1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	0	
			0	5																		
			0	6																		
			0	7																		
			1	0																		
			1	1																		
			1	2																		
			1	3																		
			1	4																		
			1	5																		
			1	6																		
			1	7																		
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		

IV. CIRCUIT DESCRIPTION AND PROGRAMMING

4.1 MICROPROCESSOR DESCRIPTION

The RS-432 uses a special-purpose, high-speed microprocessor with an instruction set dedicated to digital test applications. The microprocessor contains its own program memory (seperate from the word generator memory) which stores the test program. This program, when executed, generates the test signals.

Figure 13 depicts the RS-432 microprocessor architecture. The microprocessor performs three basic functions:

1. Generates all required timing, control, and handshake signals to stimulate the user's UUT. This is accomplished by several instructions that generate output flags and pulses, as well as test the condition of input flags and pulses from the UUT.

2. Controls the RS-432 word generator. There are eight instructions dedicated to the word generator control.

3. Provides overall control of the test system. Items 1 and 2 may be considered part of this control. However, several other instructions enable further control. Load and test loop counter instructions enable the programming of multiple loop sequences, general-purpose counting, and delay timing routines. A conditional jump instruction enables the program to branch, dependent on front panel sense switches. Finally, an unconditional jump instruction enables the program to repeat operations and to jump to and from subroutines, or it can be used as a No Operation (NOP) or HALT instruction.

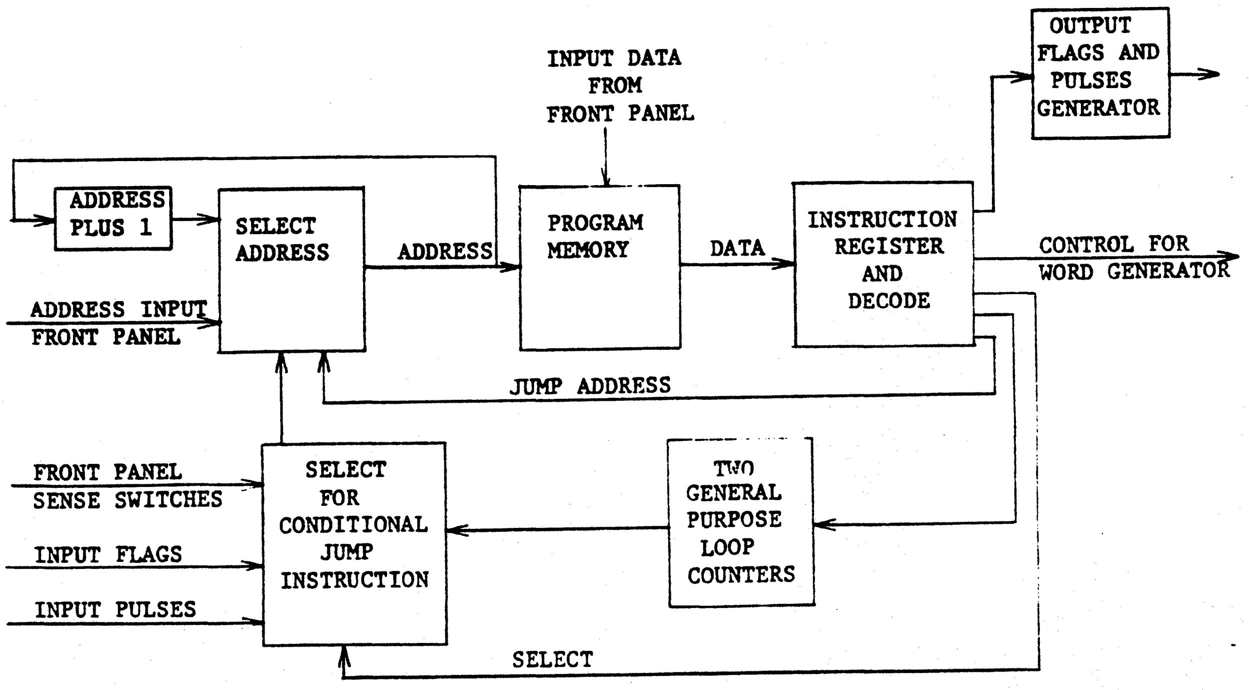

OUTPUT FLAGS AND PULSES GENERATOR
INPUT DATA FROM FRONT PANEL
ADDRESS PLUS 1
SELECT ADDRESS
ADDRESS
PROGRAM MEMORY
DATA
INSTRUCTION REGISTER AND DECODE
CONTROL FOR WORD GENERATOR
ADDRESS INPUT FRONT PANEL
JUMP ADDRESS
FRONT PANEL SENSE SWITCHES
INPUT FLAGS
INPUT PULSES
SELECT FOR CONDITIONAL JUMP INSTRUCTION
TWO GENERAL PURPOSE LOOP COUNTERS
SELECT

Figure 13. MICROPROCESSOR SECTION BLOCK DIAGRAM

The program memory can store as many as 256 program instructions, each 16 bits wide. The instruction set consists of 16 instructions, each performing a unique function, but having a similar format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation Code				Data for Operation											

The operation code (Op Code), consisting of the four most significant bits, defines the operation that will take place when that particular instruction is executed. The data field, consisting of the 12 least significant bits, supplies data for the given operation. For example:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0
Op Code				Data											

This Op Code specifies this instruction to load a loop counter. The data specifies in binary the value 16 to be loaded into the loop counter.

Detailed descriptions of the 16 executable instructions are provided in Table 6. A condensed instruction listing is provided in the Appendix for easy reference.

4.1.1 Operation/Program Considerations

The microprocessor requires a sequence of instructions written in the general format described previously. Each instruction sequence is dependent upon the specific application. The user enters these instructions into the program memory.

When the START PROGRAM switch is activated, the first instruction is executed (the first instruction is the one residing in the memory location defined by the program memory address LEDs). After the first instruction is executed, the program memory address counter steps to the second instruction. The second instruction is executed, then the third, fourth, etc.

The program memory address counter continues to sequentially execute instructions until a jump instruction is reached. The jump instructions have the same general format as all other instructions, except that they may force the program memory address counter to branch to an address other than the next sequential address. For example, assume that this instruction resides at program memory address 15:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	X	X	X	0	0	0	0	0	0	1	0	0
Op Code				Data											

The Op Code defines this instruction to be an unconditional jump instruction. The data field is where the next instruction to be executed resides; for example at program address 4, instead of following the normal sequence and executing the instruction at address 16 after 15 is executed, the program will jump to address 4 and execute that instruction. Note that, if an unconditional jump instruction specifies its own program address as the jump address, the program will effectively halt at that location. If an unconditional jump instruction specifies the next program memory address as the jump address, the instruction can be considered a No Operation (NOP).

With the two previous instruction types, a program could be written to sequentially perform several functions and then either halt or jump back and repeat the operation. For control operations, it is desirable to program the machine so that the instruction can make a decision to jump or not to jump. This is accomplished with the conditional jump instruction, in which the instruction specifies a condition that it would like to test. If the condition tested is true, the jump takes place. If not, the program address counter simply increments to the next instruction. Note that several conditions are testable: front panel sense switches, four input pulse lines, four input flag lines, two loop counters, and the state (on or off) of the word generator. For example, assume that this instruction resides at program memory address 15:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0
Op Code				Data											

The Op Code specifies this instruction to test the position of the four sense switches on the front panel. Bits 10 and 11 specify to test sense switch 0. Bit 9 specifies that the condition to test for is the upper position. Thus, the entire instruction says that if sense switch 0 is in the upper position, jump to address 8. Otherwise, step to program address 16.

Using these basic types of operation the microprocessor section can be programmed to perform the RS-432 control functions. Detailed explanations of each instruction appear in Table 6. The Appendix contains a condensed instruction listing.

NOTE: After becoming familiar with the Instruction Set, the user should read and understand the implications of Program Example 6, Program Timing Considerations.

Table 6. INSTRUCTION SET

JUMP-ON SENSE SWITCH (JSS)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	SS		T	J	JA							
Op Code															

Jump to the JUMP ADDRESS if sense switch SS is in position defined by T.

T=0; test for lower position
T=1; test for upper position

J=0; JUMP ADDRESS = JA (binary; bit 0=LSB)
J=1; JUMP ADDRESS = 8 LSBs of input register from front panel

SS=00; sense switch 0
SS=01; sense switch 1
SS=10; sense switch 2
SS=11; sense switch 3

Note that logic is provided to debounce each sense switch.

NOTE: Some switches have display uses as well as program use.

JUMP-ON INPUT FLAG (JIF)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	IF		T	J	JA							
Op Code															

Jump to the JUMP ADDRESS if input flag IF is in state defined by T.

T=0; test for 0 (0 volts)
T=1; test for 1 ($\geq$2.4 volts)

J=0; JUMP ADDRESS = JA (Binary; bit 0=LSB)
J=1; JUMP ADDRESS = 8 LSBs of input register from front panel

IF=00; input flag 0 (COMP+, input compare option)
IF=01; input flag 1
IF=10; input flag 2
IF=11; input flag 3 (ALOPD+, RSM option)

Reference the JIP instruction if more than four input flags are required. Input flags are not latched in the RS-432; they are simply buffered by one TTL gate with all logical inversions being accounted for by the RS-432.

NOTE: Some input flags are dedicated for some options.

Table 6. INSTRUCTION SET (Cont'd)

JUMP-ON INPUT PULSE (JIP)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	IP		T	J	JA							
Op Code															

Jump to the JUMP ADDRESS if input pulse IP is in the state defined by T.

T=0; pulse did not occur
T=1; pulse did occur

J=0; JUMP ADDRESS = JA (binary; bit 0=LSB)
J=1; JUMP ADDRESS = 8 LSBs of input register from front panel

IP=00; input pulse 0 (AEQLB, Alu option)
IP=01; input pulse 1
IP=10; input pulse 2
IP=11; input pulse 3 (START-, IEEE option)

Input pulse (defined as a high-to-low transition) logic latches the pulse until the selected pulse is sampled by a JIP instruction. After execution of the JIP instruction, the selected pulse latch is reset and awaits another pulse. The minimum pulse duration that is certain to set the latch is 25 nsec.

The RS-432 input pulses are wired to sample a negative edge ($^{1}\neg_{0}$). The user may change this to a positive edge ($_{0}\lrcorner^{1}$) by grounding a wire in the RS-432. Refer to the logic diagrams in the RS-432 Maintenance Guide for this change.

Note also that each input pulse can be converted into an input flag with an appropriate wiring change. Refer to the logic diagram in the RS-432 Maintenance Guide for this change.

NOTE: Some input pulses not available for some options.

JUMP-ON LOOP COUNT (JLC)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	1	LC		T	J	JA							
Op Code															

The JLC instruction interrogates several conditions, depending on the value of LC and T.

Jump to the JUMP ADDRESS if the conditions of LC and T are as defined below:

LC=00 T=0; jump if loop counter #0 ≠ 0 (not done)
T=1; jump if loop counter #0 = 0 (done)

LC=01 T=0; jump if loop counter #1 ≠ 0 (not done)
T=1; jump if loop counter #1 = 0 (done)

JUMP-ON LOOP COUNT (JLC) (Cont'd)

This instruction, with LC=00 or 01, automatically decrements the respective loop counter <u>after</u> execution of the instruction. Note that no other instructio or condition can decrement either of these two loop counters, and that these loop counters are completely independent of the word generator loop counter.

LC=10 T=0; jump if Word Generator Busy flag is not set
T=1; jump if Word Generator Busy flag is set

The Word Generator Busy (WGB) flag sets immediately after a start word generato loop (STL) instruction. This flag then resets when the word generator has completely transmitted its data; i.e., last memory address data (as defined by DLA instruction) has been transmitted, all shifting of the LMA data is complete (as defined by DØP instruction), the required number of loops (as defined by the STL instruction) are complete and finally, the word generator is back at the first memory address, awaiting another start command.

LC=11 T=0; jump if LMA·SCF flag is not set
T=1; jump if LMA·SCF flag is set

The LMA·SCF flag sets when the last word memory address (as defined by the DLA instruction) data is in the output register and all shifts (as defined by the DØP instruction) are complete. This instruction is useful when all word generator control remains within the microprogram. In other words, the word generator is incremented with a clock output shift register (CSR) instruction and <u>not</u> a STL instruction. Note that the STL loop counter does not affect this flag as in the LC=10 case. Reference Program Example 5.

J=0; JUMP ADDRESS = JA (binary; bit 0=LSB)
J=1; JUMP ADDRESS = 8 LSBs of input register from front panel

UNCONDITIONAL JUMP (JUN)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	X	X	X	J	JA							
Op Code															

Unconditionally jump to JUMP ADDRESS.

NOTE: Used as HALT instruction if JUMP ADDRESS = instruction address.
Used as NOP instruction if JUMP ADDRESS = instruction address + 1.

J=0; JUMP ADDRESS = JA (binary; bit 0=LSB)
J=1; JUMP ADDRESS = 8 LSBs of input register from front panel

X = don't care.

Table 6. INSTRUCTION SET (Cont'd)

LOAD LOOP COUNTER (LOC)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	L	LOOP COUNT										
Op Code															

Load loop counter 0 (L=0) or loop counter 1 (L=1) with data defined by Loop Count. Loop Count data is binary; bit 0 = LSB.

These two general-purpose loop counters permit the programming of multiple loop routines. They can also be used as delay counters, keeping in mind that each instruction requires 200 nanoseconds for execution.

The LOC instruction, used in conjunction with the JLC instruction, tests these counters for the value zero. Note that the selected loop counter is automatically decremented by 1, after the JLC instruction makes the test.

GENERATE OUTPUT FLAGS (GØF)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	P_1	P_1	P_0	P_0	F_7	F_6	F_5	F_4	F_3	F_2	F_1	F_0
Op Code															

Set output flag F_n high ($\geq$2.4 volts) if F_n=1 or reset the output flag F_n low (0 volts) if F_n=0, where n=0, 1, 2, ...7.

P_0P_0=00; reset output pulse 0 to 0 (0 volts)
P_0P_0=01; pulse output pulse 0 high for 100 nsec
P_0P_0=10; pulse output pulse 0 low for 100 nsec
P_0P_0=11; set output pulse 0 to 1 ($\geq$2.4 volts)

P_1P_1=00; reset output pulse 1 to 0 (0 volts)
P_1P_1=01; pulse output pulse 1 high for 100 nsec.
P_1P_1=10; pulse output pulse 1 low for 100 nsec
P_1P_1=11; pulse output pulse 1 to 1 ($\geq$2.4 volts)

Note: Output pulse one is used for SRQ with IEEE option.

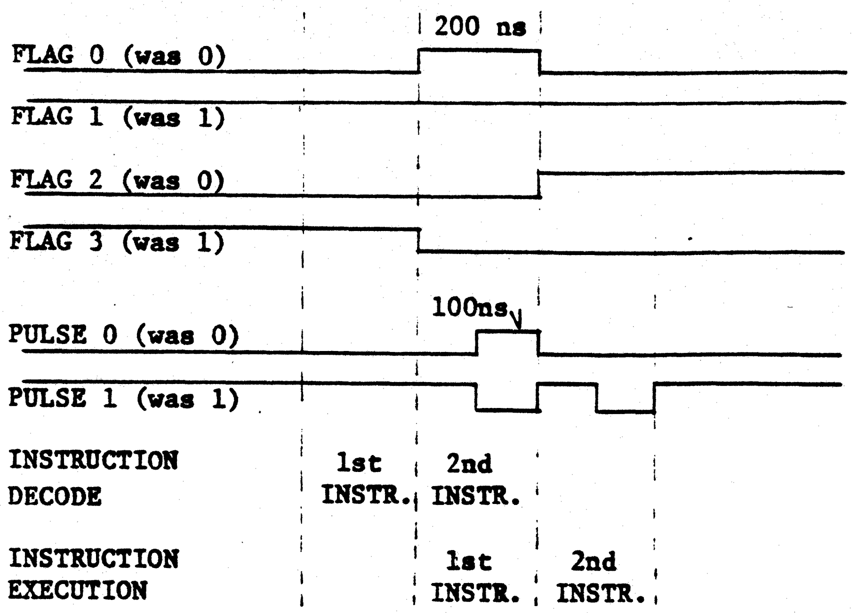
Example: The following depicts the output timing relationship if two GØF instructions are executed consecutively. It is assumed that a prior GØF instruction placed the flag and pulse lines in their present starting state:

	Op Code	
1st INSTRUCTION	0 1 1 0	1 0 0 1 X X X X 0 0 1 1
2nd INSTRUCTION	0 1 1 0	1 0 0 0 X X X X 0 1 1 0

NOTE: Some input flags and pulses are dedicated for certain options.

Table 6. INSTRUCTION SET (Cont'd)

GENERATE OUTPUT FLAGS (GØF) (Cont'd)



Refer to Program Example 6, Program Timing Considerations, for additional timing information on the GØF instruction. Note that, although not illegal, a pulse line can be high and commanded to pulse high or a pulse line can be low and commanded to pulse low. In both cases, a pulse will occur; however, the respective line will finally be quiescent, in the opposite state from which it started.

HALT UNTIL PERIOD COMPLETE (HPC)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	X	X	X	X	X	X	X	X	X	X	X	X
Op Code															

The HPC instruction is the only instruction that does not have a 200-nsec execution time.

The HPC instruction halts program execution until the rising edge of the free-running period counter clock (FEXCK+). It enables the output of word data to be in sync with the free-running period counter clock. NOTE: This instruction is not necessary if a start word generator loop (STL) has been executed. However, the HPC instruction is useful if it is desired to access word data under microprocessor program control (by executing FMW and CSR instructions).

HALT UNTIL PERIOD COMPLETE (HPC) (Cont'd)

RS-432 timing has been designed so that, if an FMW_1, FMW_2, CSR, or ALU instruction (described later) is executed <u>immediately</u> after an HPC instruction, the word memory data accessed will be output (loaded in the output shift register) at precisely the same time as the rising edge (_|‾) of the free-running period clock (FEXCK+). Refer to Word Generator Clock section (Page 14) for the relationship of period clocks and data. Refer to Program Example 5 for a typical application.

The following instructions supply parameter and control functions for the word generator.

DEFINE PERIOD (DEP)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	M	M	PERIOD									
Op Code															

Load the word generator section period counter with the multiplier and period data defined by MM and PERIOD.

MM=00; 1 msec	PERIOD: Period count is binary; bit 0=LSB
MM=01; 100 usec	
MM=10; 1 usec	
MM=11; 100 nsec	

The period counter will free-run at the specified period immediately after this instruction. NOTE: Once the period is defined, this instruction should <u>not</u> be repeated unless it is desired to change the period count because immediately after execution of the DEP instruction, the period counter is preset to the desired period count. Thus, if the DEP instruction is executed anywhere in the middle of the period, the period counter will be altered, causing an erroneous period count.

Refer to the Word Generator Clock section (Page 14) for the relationship of period clocks and data.

DEFINE WORD OUTPUT PARAMETERS (DØP)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	WPLMW						WPMW					
Op Code															

Table 6. INSTRUCTION SET (Cont'd)

DEFINE WORD OUTPUT PARAMETERS (DØP) (Cont'd)

The DØP instruction defines whether output data will be in parallel or serial format. If in serial format, it further defines the number of shifts that the output register makes before loading a new memory word. Two parameters control the output format, Words Per Memory Word (WPMW) and Words Per Last Memory Word (WPLMW). For parallel data, WPMW and WPLMW are set equal to 1, indicating one output word per memory word. For serial data, WPMW and WPLMW are set to indicate in binary the number of output words contained in one memory word; i.e., the number of bits to be shifted out of the output register The maximum value of WPMW and WPLMW for the standard RS-432 is 16.

Thus, the DØP instruction loads the word generator section with the words (shifts) per memory word register and the words (shifts) per last memory word register with data defined by WPMW and WPLMW.

WPMW and WPLMW count is binary; bit 0 and bit 6 are LSBs.

Refer to the Word Generator Output Shift Register section (Page 40) for a detailed description.

DEFINE LAST MEMORY ADDRESS (DLA)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	0						LMA						
	Op Code														

Load the word generator last memory address register with LMA data.

LMA data is binary; bit 0=LSB

This instruction defines the last word memory address of the word generator. The word generator, when cycling, steps sequentially through each address until the data from the last memory address is accessed. At the next clock, the address counter then steps to the first memory address (see instructions FMW_1 and FMW_2) and resumes its cycling. **The DLA instruction must be located prior to a FMW1 or FMW2 in an instruction sequence.**

FETCH MEMORY WORD 1 (FMW_1)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1							FA					
	Op Code														

Fetch the word generator word memory data at the address defined by FA. This data is automatically loaded into the output shift register.

FA address data is binary; bit 0=LSB

Table 6. INSTRUCTION SET (Cont'd)

FETCH MEMORY WORD 1 (FMW_1) (Cont'd)

This instruction performs a dual function: (1) it enables the program to fetch (output) any word from the word memory, and (2) when it is executed, the address defined by FA is loaded into the first word memory address register. Thus, the last FMW instruction executed before the word generator is started defines the location of the first word memory address.

FETCH MEMORY WORD 2 (FMW_2)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	X	X	X	X	X	X	X	X	X	X	X	X
Op Code															

This instruction fetches the word generator memory data at the address defined by the 12 LSBs of input register. The data is automatically loaded into the output shift register.

Input register data is binary; bit 0=LSB

This instruction is identical to the FMW_1 instruction, except that the address for the desired data is defined by the front panel switches via the input register. Note that the FMW_2 instruction enables the user to manually define the start address of a word memory block. Thus, with proper programming, the user can access different blocks of data by simply changing the data on the input register.

NOTE: This instruction has been deleted and a DFMA instruction used with this opcode for RSM option.

CLOCK OUTPUT SHIFT REGISTER (CSR)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	0	0	0	0	0	0	X	X	X	X	X	X
Op Code															

The CSR instruction is used when the program desires complete control of the word generator. Normally the word generator rate is controlled by the period counter (refer to the DEP instruction). However, the CSR instruction enables the program to provide clocks to the word generator rather than the period counter providing the clocks. When program control is desired, the STL instruction will not be executed.

Before CSR instructions are executed, the user must still (via the program) define the first memory address (FMW_1 or FMW_2 instruction), last memory address (DLA instruction), and the word output parameters (DØP instruction). The period may also be defined (DEP instruction) if the program is to be synchronized with the period counter via an HPC instruction.

CLOCK OUTPUT SHIFT REGISTER (CSR) (Cont'd)

Once all parameters are defined, the program may clock the word generator with the CSR instruction. This clock, in turn, will cause the output register to either shift or load the next memory word, depending on the WPMW count. With each clock, the word generator will sequentially output the defined block of data until the last memory address is reached. If CSR instructions continue, the first memory address data is fetched and the process repeats. Note that the program can monitor the word memory address counter of the word generator by sensing the Last Memory Address and Shifts Complete Flag (LMA.SCF) via the JLC instruction (refer to Program Example 5).

For configurations of RS'432's contailng ALU options, the CSR instruction can can simultaneously control both ALU output registers and the normal output registers. This is accomplished by executing CSR instructions with the following format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	SA		SM				Cn	M	S_3	S_2	S_1	S_0
Op Code															

SA = Select ALU

SM = Select Mode

ALU Function
(Refer to Table 8, pg. 51)

11	10	
0	0	Normal CSR
0	1	ALU in bit position 0-15 is selected
1	0	ALU in bit position 32-47 is selected
1	1	ALU in bit position 0-15 and 32-47 is selected

9	8	7	6	
0	0	0	0	Word Memory to ØR(AØR)
0	0	0	1	No transfer
0	0	1	0	" "
0	0	1	1	" "
0	1	0	0	ØR(AØR) to GPR1
0	1	0	1	" " GPR2
0	1	1	0	" " GPR3
0	1	1	1	" " GPR4
1	0	0	0	GPR1 ALU AØR results to ØR(AØR)
1	0	0	1	GPR2 " " " " "
1	0	1	0	GPR3 " " " " "
1	0	1	1	GPR4 " " " " "
1	1	0	0	GPR1 ALU AØR
1	1	0	1	GPR2 " "
1	1	1	0	GPR3 " "
1	1	1	1	GPR4 " "

Refer to Page 47 for a description of the ALU Option.

By executing CSR instructions in this format, those output registers connected to the ALU are loaded from the ALU as commanded by the CSR instruction bit field, and the other output registers are loaded from the Word Memories in the normal fashion. The CSR also increments the Word Memory address counters thus allowing data patterns to be output from word memory simultaneously with ALU data generation on another output register.

Table 6. INSTRUCTION SET (Cont'd)

START WORD GENERATOR LOOP (STL)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	P	C	NL									
Op Code															

The STL instruction starts the word generator by defining the clock source (as defined by P). It also defines the number of times the entire word block (as defined by the first memory address and the last memory address) is to be repeated. The repeat count is defined by NL, the number of loop counts.

Note that when the word generator has accessed all memory locations and repeated them if the NL count so specifies, it will stop at the first memory address location.

NL: number of loops binary; bit 0=LSB

P=0; select period counter clock	C=0; perform number of loops specified by NL
P=1; select external clock	C=1; run continuously (ignore NL count)

The STL instruction immediately sets the Word Generator Busy (WGB) flag. The WGB flag is reset by two conditions: (1) when the number of loops, (defined by NL) are completed and the LMA data has been processed, and the data from the first memory address is reloaded into the output shift register; or (2) when running continously, i.e., C=1, the WGB flag is reset by executing a DLA instruction. The program can monitor the WGB flag via the JLC instruction.

Note that P=1 allows the user to supply an external clcok as a substitute for the internal period counter clock. Immediately after an STL instruction is executed with P=1, the word generator awaits external clocks. These clocks then activate the word memory address logic and output shift register exactly as if they were receiving period counter clocks. When the word generator determines its sequence to be complete (all data has been transmitted and repeated the required number of times), the WGB flag is reset and the microprocessor resumes control of the word generator. Refer to the Input Clock section, Page 18 for timing details.

The user shall note that the microprocessor is still executing program instructions after executing an STL. That is, the STL instruction starts the word generator section and does not affect the microprocessor sections' ability to execute instructions. Thus, the user should plan to insert appropriate instructions to control the microprocessor while the word generator is running. Refer to the Program Examples section, Page 48 and 63 for STL examples.

NOTE: Modifications have been made for RSM option.

ARITHMETIC OPERATION (ALU)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	SA		SM				Cn	M	S_3	S_2	S_1	S_0
Op Code															

The ALU instruction should be executed only when the RS-432 contains the ALU option. Refer to the Options section, page 48 for a detailed explanation of this instruction and its capabilities.

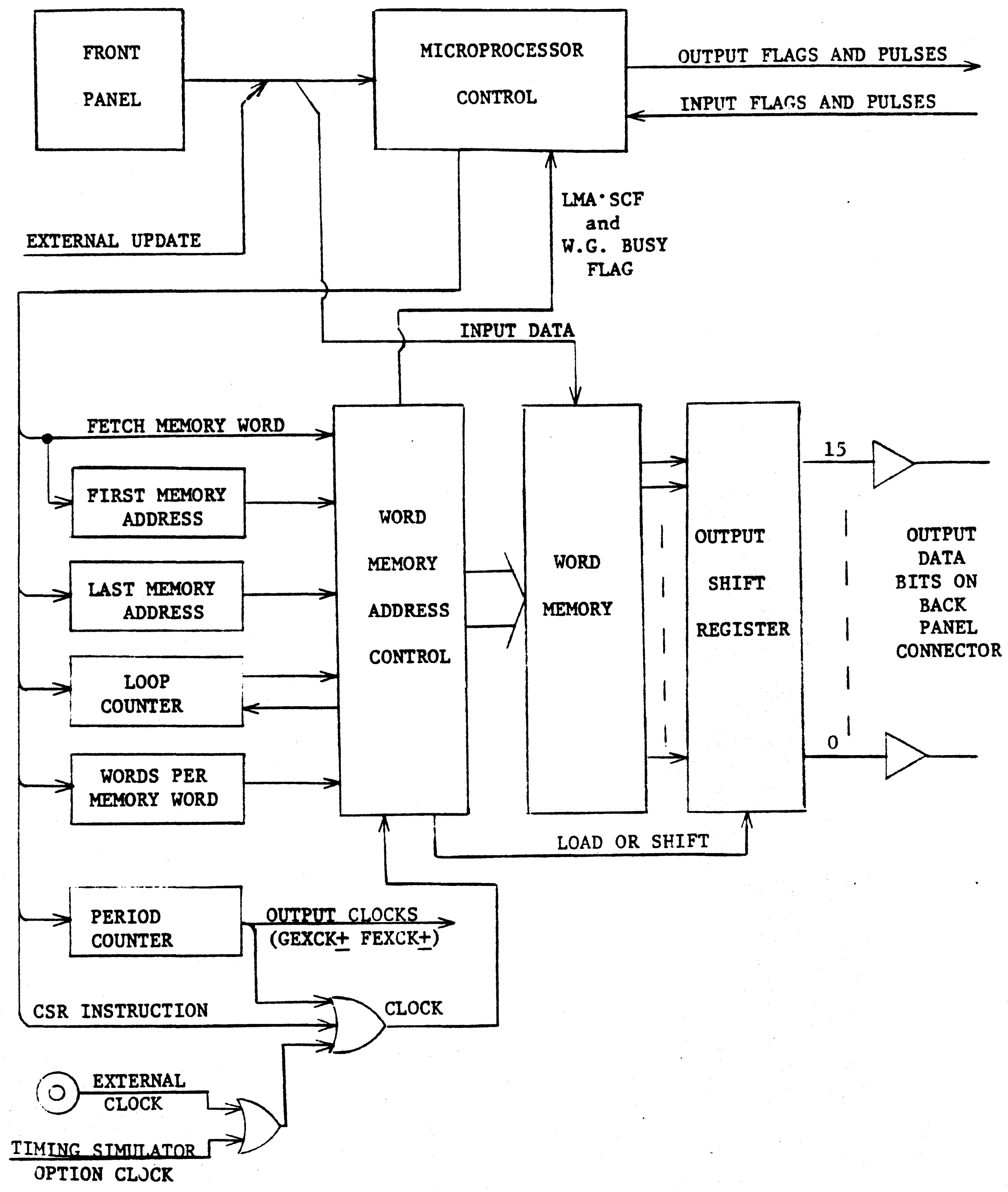


Figure 14. WORD GENERATOR ARCHITECTURE

a clock shift register (CSR) instruction is executed. Again, the word memory address counter starts at the first memory address, and the address counter increments each time it receives the CSR clock. After the last memory address location is reached, the address counter presets back to the first memory location if another CSR clock is received. The WGB flag is not set when under CSR control. Also, as in the STL mode, word generator parameters must be defined before CSR clocks are generated. Program Examples 4 and 5, show how identical data patterns can be generated by using the two clocking methods.

4.2.3 Output Shift Register

The output shift register is a 16-bit, parallel-in, serial- or parallel-out register with line drivers attached to each output. The output register performs two functions: (1) it simply stores each memory word as it is fetched from memory, and (2) it can be programmed to convert each 16-bit parallel memory word into a serial data train.

The parallel-to-serial conversion is accomplished by shifting the output register one place to the right (MSB$\rightarrow$LSB) with each register clock. The number of shifts is determined by the words per memory word (WPMW) registers (reference the DØP instruction, Page 33). Thus, if 16-bit parallel data is desired, the WPMW register is loaded, via the microprocessor program, with the value 1 (i.e., consecutively load the output register, never shift data). If single-bit serial data is desired, the WPMW register is loaded with the value 16 (i.e., load the output register and shift all register data bits out on register bit 0). When all shifts are complete, the output register is loaded with the next memory word.

The WPMW count, in effect, determines the number of data bits output before a new word is loaded into the output shift register. Refer to Figure 15 for representation of the output shift register.

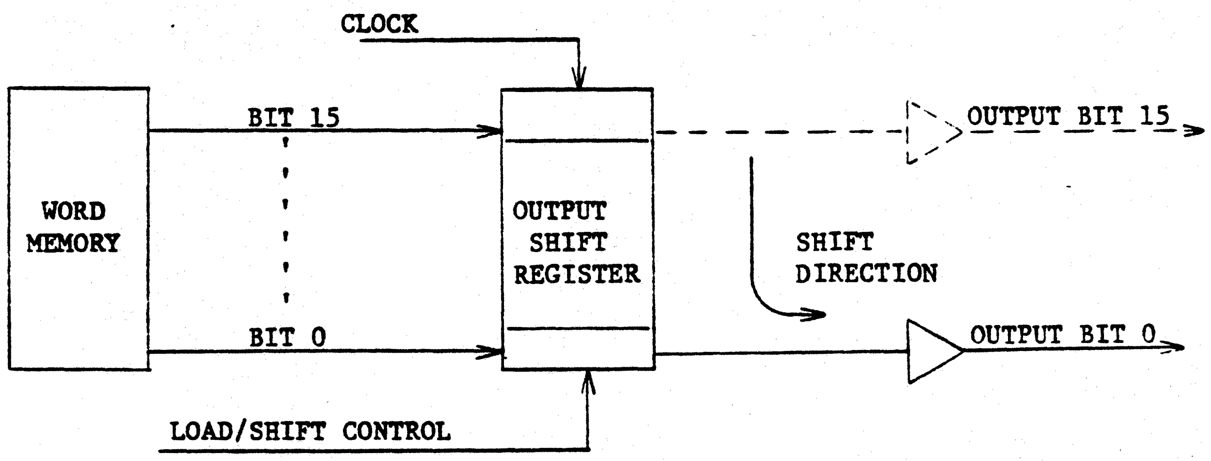


Figure 15. OUTPUT SHIFT REGISTER

Therefore, with different combinations of loading and shifting the output register, any output word size can be achieved while maximizing the word generation efficiency of the word memory. The WPLMW register also stores the shift value for the last memory address data, allowing any odd or even total number of shifts to be achieved by controlling the number of bits generated by the word stored in the last memory address location.

The clocks received by the output register are the same as those received by the word memory address control logic. However, internal address control logic assures that the address counter will increment only when it is time to fetch the next memory word. It will automatically gate off clocks whenever the output register is shifting its data.

NOTE: The user should not concern himself with the timing, loading, or shifting of the data or memory delays. He need only specify the word generator parameters via the proper instructions and then enable the clock pulses, either under program control (CSR instruction), from the period counter (STL instruction), or from an external source (STL instruction).

4.2.4 Period Counter and Output Clocks

The period counter may be set to any period count from 100 nanoseconds to 1.023 seconds (refer to the DEP instruction, Page 33). The basic clock driving the period counter is a 20-mHz crystal-controlled oscillator with an accuracy of ± 5 ppm/C° over 0 to 65°C. This clock is digitally counted down to derive the desired period count.

Once the period count is defined by a DEP instruction, the period counter free-runs at the specified period until either the period count is changed (another DEP instruction is executed) or the power is shut off.

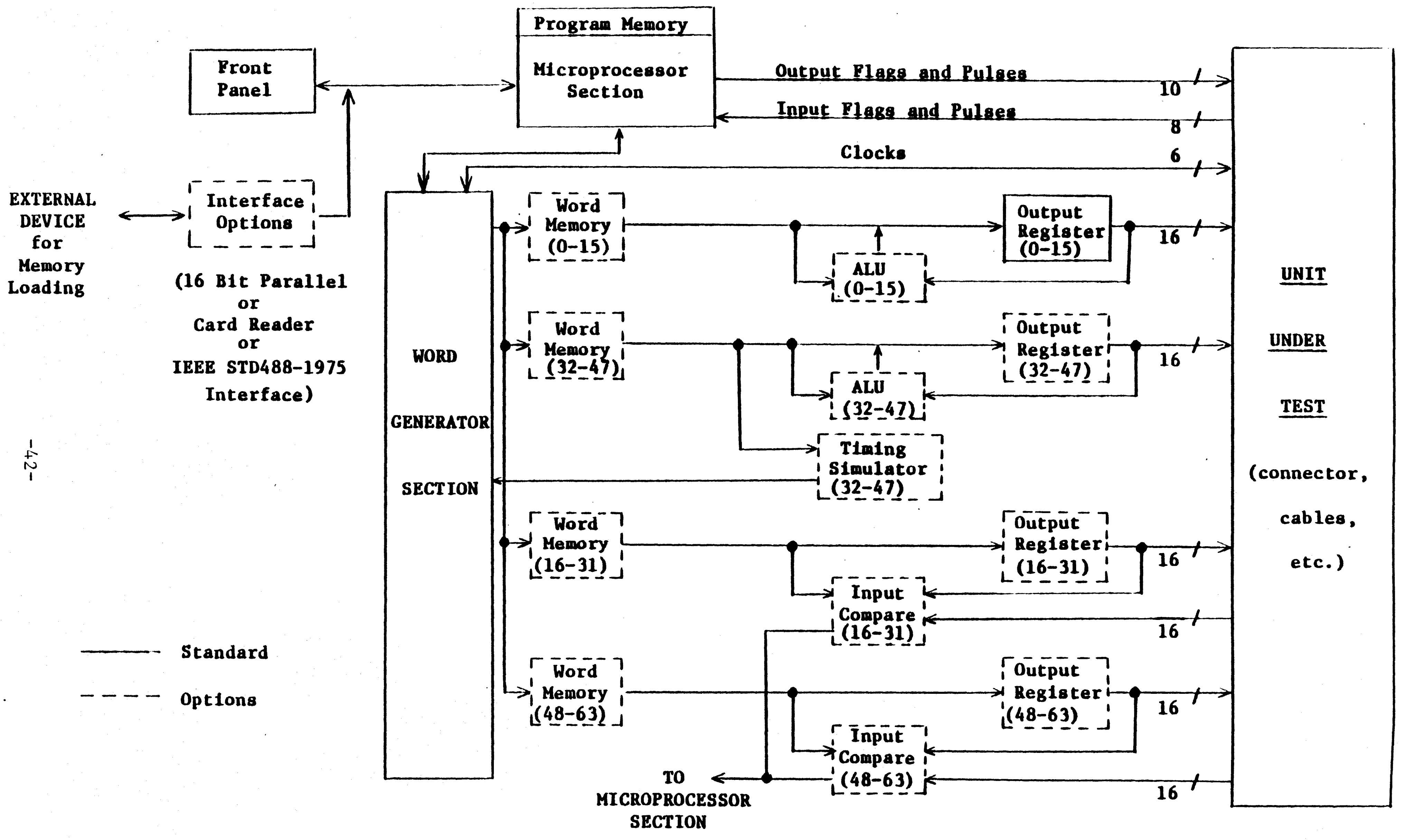
4.3 OPTIONS

Options available with the RS-432 include: additional output register cards, input data comparator cards, timing simulator card, and arithmetic/logic function output card. There are also several memory load devices that can be configured into the standard RS-432: a 16-bit parallel interface, an RS-232C compatible ASCII interface, an IEC (HPIB) or IEEE Standard 488-1975 compatible ASCII interface, and a card reader with interface.

Figure 16 depicts the RS-432's standard hardware, along with all available input/output options. This block diagram shows the maximum allowable option configuration, although the options need not be positioned as shown. For example, an input comparator could be installed in the ØR 0-15 position with or without the ALU option.

4.3.1 Additional Output Register Cards

The basic RS-432 is wired with one 16-bit output shift register card in the word generator. This card requires one 16-bit word memory card to supply the test data. The basic output register structure is shown in Figure 17.



interface TECHNOLOGY

Figure 16. RS-432 BLOCK DIAGRAM

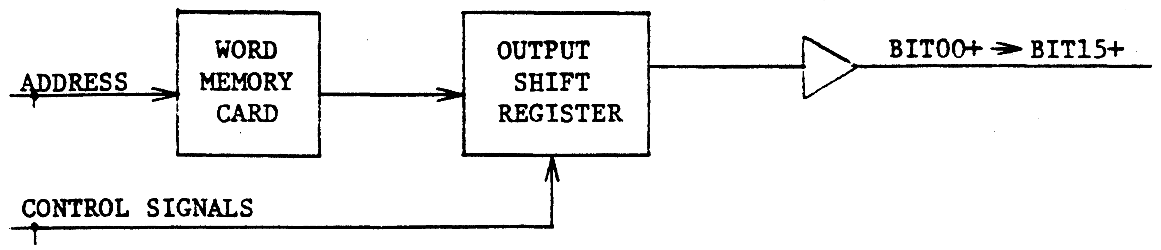


Figure 17. ONE OUTPUT REGISTER

Additional output register cards (Figure 18) can be installed individually to yield a maximum of 64 data bits. Each additional register requires a memory card. Since memory and output register cards share common address and control signals, the timing between any of the 64 data bits is the same as that between any two data bits in the first 16-bit output register. (Refer to Figure 10 for the data bit timing relationship.)

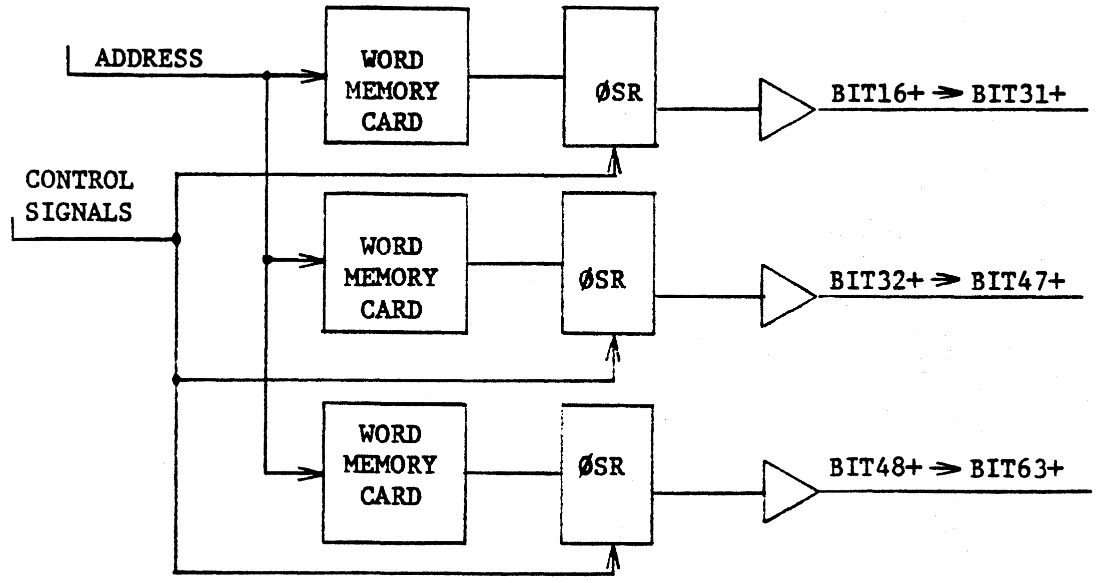


Figure 18. ADDITIONAL OUTPUT REGISTERS

Table 7 shows how data bits 32 thru 63 can be loaded, updated, and examined from the front panel. Sense switch 3 is used for memory selection when more than 32 bits of output data are required.

Table 7. FRONT PANEL SWITCH CONFIGURATION FOR GREATER THAN 32 BIT OPERATION

SENSE SWITCH 3	WORD MEMORY DATA 0-15 SWITCH	WORD MEMORY DATA 16-31 SWITCH	OUTPUT REGISTER DISPLAYED AND SELECTED AS THE DESTINATION REGISTER
LOW	ON	----	0-15
LOW	----	ON	16-31
HIGH	ON	----	32-47
HIGH	----	ON	48-63

The additional 32 output bits are wired to a similar back panel output connector, as are the other RS-432 inputs and outputs. Detailed pin assignments are included in the Appendix.

Unless specifically requested otherwise, RS-432's that have more than 16 output bits will be limited to 16 serial bits per memory word; that is, the serial output of the most significant 16 bits is not tied to the serial input of the least significant 16 bits. Thus, the capability of generating one 32-bit serial output data channel does not exist. The same is true for 48- or 64-bit configurations.

In summary, the words per memory word (WPMW, WPLMW) parameters have maximum values of 16. However, this value can be modified; if one references the DØP instruction, a maximum value of 63 can be achieved for each parameter. Factory contact should be made for applications requiring this modification.

4.3.2 TIMING SIMULATOR OPTION

The timing simulator option card enables the RS-432 to generate 16, 32, or 48 channels of timing signals with a resolution of 100 nanoseconds. It does this by converting the word generator portion of the RS-432 into a Timing Simulator. It is suggested that the reader become familiar with Interface Technology's Timing Simulator/Word Generator Model RS-648. In particular, the RS-648 Application Note #1, "What's A Timing Simulator", should be read.

Figure 19 is a block diagram of the word generator with the timing simulator option installed. Note that additional memory must be installed to store the time information required for timing simulation.

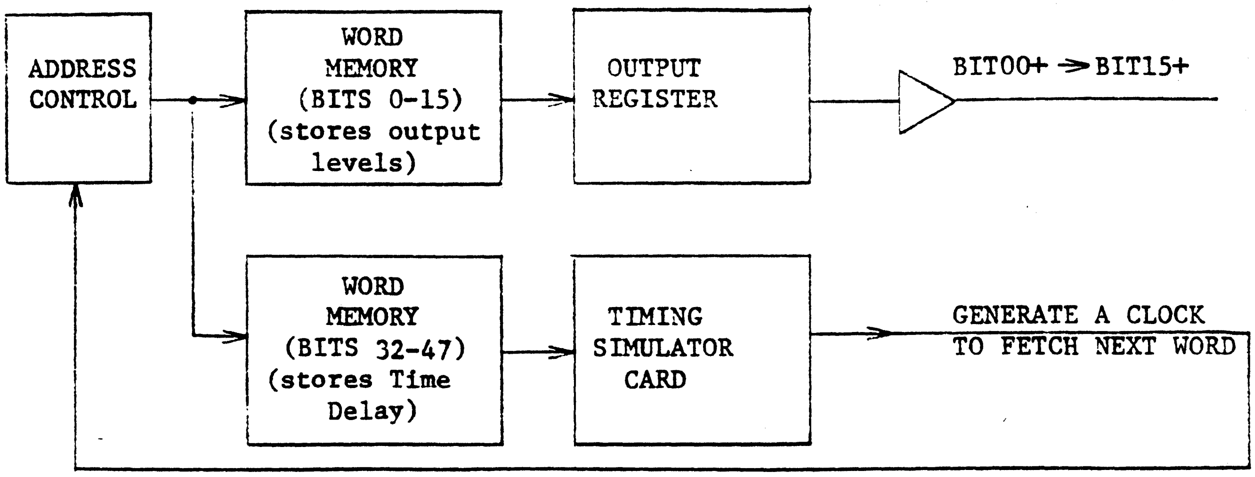

ADDRESS CONTROL
WORD MEMORY (BITS 0-15) (stores output levels)
OUTPUT REGISTER
BIT00+ → BIT15+
WORD MEMORY (BITS 32-47) (stores Time Delay)
TIMING SIMULATOR CARD
GENERATE A CLOCK TO FETCH NEXT WORD

Figure 19. TIMING SIMULATOR BLOCK DIAGRAM

The following notes explain how the timing simulator option affects the RS-432. It does not affect any instruction except as specified in these notes.

Note 1: For each output data word (bits 0-15, 16-31, 48-63), an additional word (bits 32-47) stores the time that the output data word remains at the generator outputs. Note that in the word generator mode, each data word is fetched at a fixed time period, whereas in the timing simulator mode, each word may be fetched with a different period. The format for storing the timing information is as follows:

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EACH MEMORY WORD BITS 32-47	X	X	X	X	M	M	TIME DELAY									
	47															32

TIME DELAY is in binary; Bit 0 (32) = LSB.

MM Time Delay Multiplier	
00	= 1 msec
01	= 100 usec
10	= 1 usec
11	= 100 nsec

X = don't care

Note 2: The memory word bit layout and multiplier time delay times have been changed for the RSM option. The timing simulator is started and stopped with an STL instruction precisely as in the word generator configuration. One stipulation, however, is that when starting the timing simulator with the STL instruction, the P value must equal 1 to select the timing simulator as an external clock. The user forfeits the external clock function when the timing simulator option card is installed.

However, he may revert to the word generator structure by simply executing an STL instruction with a P value of 0. Since C and NL functions are not affected by the timing simulator option, multiple loop and continuous run functions may be executed.

Note 3: When the timing simulator is started with an STL instruction and the C bit equals 0, the timing simulator starts and stops at the first memory address. Note that, when the define and fetch first memory (FMW) address instruction is executed, the RS-432 outputs are defined by the data stored at this FMW location. The time for this first word will be processed when the STL instruction is executed. Subsequent passes through the first memory word will be processed similarly if the STL instruction requests multiple loops. If one loop (single cycle) is specified the RS-432 will process the last word, fetch the first word, and then stop.

Note 4: The STL instruction immediately sets the Word Generator (timing simulator) Busy (WGB) flag. The WGB flag is reset when either the number of loops defined by NL are complete and the LMA data has been processed (signified by the data at first memory address being loaded into the output shift register), or when running continuous, whenever a DLA instruction is executed.

Note 5: Output timing relationships for the output channels are identical to the word generator. (See Figure 10).

4.3.3 INPUT COMPARATOR OPTION

The input comparator option card allows the program of the microprocessor section to compare as many as 32 bits of input data from the UUT against known correct data stored in the word generator. Each input comparator card has 16-bit compare capability, requiring two cards to generate a full 32-bit compare. Further, each bit of the 16-bit comparator can be masked off, permitting compare routines with fewer than 16 data bits; for example, a serial data train can be tested by inhibiting 15 of the data bits. Figure 20 depicts the comparator card in block diagram form.

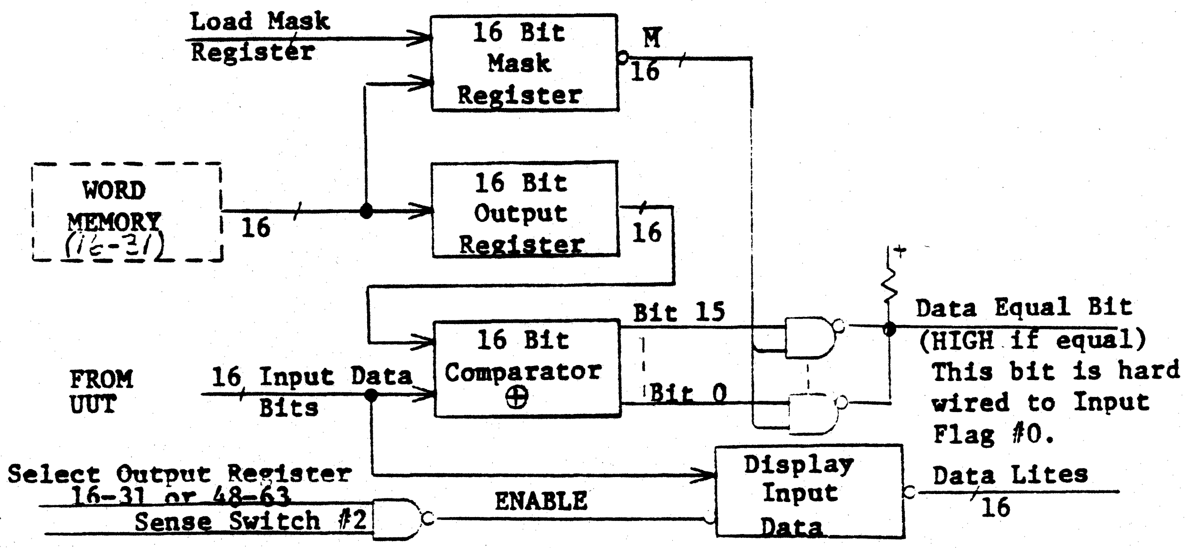


Figure 20. INPUT COMPARATOR BLOCK DIAGRAM

Thus, in conjuction with the word generator, the program generates correct or expected data patterns in the output register. When the program determines the input data should equal the output register data, input flag 0 is tested for the value "1" with a JIF instruction.

Each 16 bit of the comparator may be enabled or disabled via the mask register. The user loads this register with either the FMW$_1$ or FMW$_2$ instruction. However, bit 10 of the instruction must be a "1", the user must dedicate at least one location in the word memory to store the mask data. The instruction and word formats are detailed below.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FMW$_1$	1	0	1	1	0	1	MASK DATA ADDRESS									
***FMW$_2$**	1	1	0	0	0	1	X	X	X	X	X	X	X	X	X	X

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Word Memory location which stores mask data	M	M	M	———									M	M	M	M

M = 0 Enable Bit
M = 1 Disable Bit

* The FMWZ instruction has been changed for the RSM option.

When the FMW$_1$ or FMW$_2$ instruciton is executed with Bit 10 high, only the mask register is loaded (i.e., all output registers are unaffected).

Note however, that mask register must be loaded before setting up the first Word Memory address register with a normal FMW instruction because the FMW instruction dedicated to the mask function will also load the first memory address register.

For machines which require 32 input compare bits, two input compare cards are required. Refer to the block diagram titled "RS-432 with Options", Figure 16 on Page 42.

The input data bits are wired to a RS-432 back panel connector in a similar fashion as the other inputs and outputs. Detailed pin assignments are included in the Appendix.

To display input data on data lights, select output register (as defined on Table 7, Page 44) and place Sense Switch #2 HIGH. Sense Switch #2 LOW will display the respective output registers.

4.3.4 ALU Output Register Card

The ALU (Arithmetic/Logic Unit) option enables the user to perform 16-bit arithmetic and logical operations in each 16 bit output register incorporating an ALU. A typical example of this function would be the use of the RS-432 as a memory tester. One ALU output register would be used as a memory address counter for the memory under test while the other ALU register performs a data pattern algorithm generating test data for the memory under test. The ALU option incorporates a 74181 type ALU circuit, which performs various arithmetic and logic functions. Also incorporated in this ALU option are four general purpose registers that can be used to store constants, intermediate values or data for subsequent ALU operations.

It should be noted that with the ALU option installed, the word memory is limited to 1K memory words.

Figure 21 is a block diagram of the ALU output register, while Table 8 lists all functions that can be performed on the output register. It should be pointed out that the ALU option contains an output register labeled AØR. This register contains the same data that the normal Output Register (ØR) contains. AØR is in effect invisible to the user.

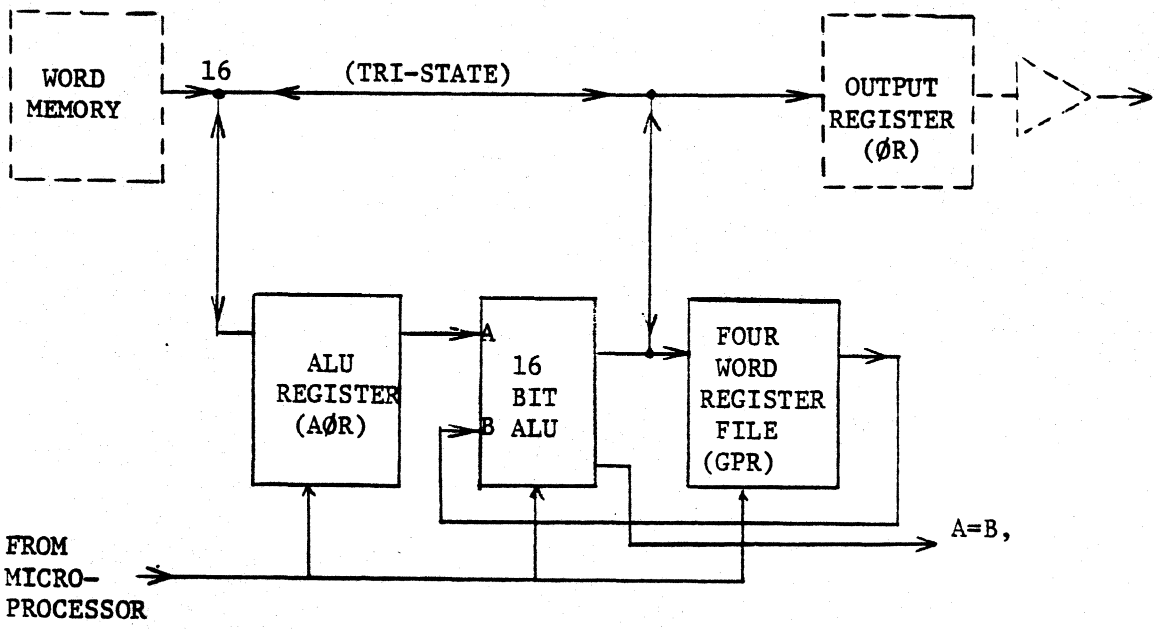


Figure 21. **ALU OPTION BLOCK DIAGRAM**

The ALU instruction is detailed below:

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALU INSTRUCTION	1	1	1	1	SA		SM				Cn	M	S_3	S_2	S_1	S_0

Select ALU Function (See Table 8)

SA = Select ALU
SM = Select mode

11	10	
0	0	Not Used
0	1	ALU in bit position 0-15 is selected
1	0	ALU in bit position 32-47 is selected
1	1	ALU in bit position 0-15 and 32-47 is selected

9	8	7	6	
0	0	0	0	Illegal codes for ALU instruction
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	ØR(AØR) to GPR1
0	1	0	1	" " GPR2
0	1	1	0	" " GPR3
0	1	1	1	" " GPR 4

9	8	7	6	
1	0	0	0	GPR1 ALU AØR results to ØR(AØR)
1	0	0	1	GPR2 " " " " "
1	0	1	0	GPR3 " " " " "
1	0	1	1	GPR4 " " " " "
1	1	0	0	GPR1 ALU AØR
1	1	0	1	GPR2 " "
1	1	1	0	GPR3 " "
1	1	1	1	GPR4 " "

Select mode (SM) codes 1100, 1101, 1110 and 1111 are used when the results of an ALU operation are to be tested only rather than stored. For example, when testing whether A=B, it may be desirable to maintain the value B. An ALU instruction would be performed with SM=1100. This would be followed by a jump on input pulse instruction to test the A=B ALU output. The value in the output register remains unchanged.

Since the ALU option was designed to operate under microprocessor program control, if an STL instruction is executed, the ALU option will be bypassed and other data will be loaded from the word memory to the output register in the normal word generator fashion.

The following notes explain how the ALU option affects the RS-432. It does not affect any instruction except as specified in these notes.

Note 1: Table 8 defines how the ALU instruction affects the output data bits. When an ALU instruction is decoded, the output register is loaded with the data coming from the ALU chip outputs. (Note the definition for ØR and DATA in Table 8.) ALU options operate as 16 bit processors unless otherwise specified.

Note 2: The four addressable registers labled GPR1, GPR2, GPR3, and GPR4 can only be loaded with data from memory via the ØR register. Thus, an appropriate ALU instruction must be executed to transfer data from the ØR(AØR) to the GPR's. These four general purpose registers can be used for storage of intermediate values, constants, or multiple values upon which the ALU will operate.

Note 3: The A=B flag is wired to input pulse 0. When testing this flag, it should be treated exactly as if it were of the four input flags, the latching function of the input pulse lines is not used. Use the JIP instruction as if it were a JIF instruction. It should also be noted that because these flags are not latched, a JIP instruction must immediately follow the ALU instruction used for comparisons to assure proper sampling of these flags.

When testing the A=B flag, the ALU function must be in the logic function (M=1) with the exclusive ØR (⊕) function; i.e.:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	SA		1	1	SM		0	1	1	0	0	1
Op Code															

Note 4: The CSR instruction can also control the ALU option. Refer to Page 35, the CSR Instruction for the appropriate Word Formats. When operating in this mode of control, each CSR instruction executed will load simultaneously, all output registers and increment to the next memory address. The output register with the ALU option will contain the ALU data as specified by the CSR instruction, all other output registers shall contain data from the respective Word Memory that they are tied to.

S_3	S_2	S_1	S_0	M = H LOGIC FUNCTION*	M = L ARITHMETIC OPERATIONS	
					C_n = L (no carry)	C_n = H (with carry)
L	L	L	L	DATA = $\overline{\text{ØR}}$	DATA = ØR minus 1	DATA = ØR
L	L	L	H	DATA = $\overline{\text{ØR}\cdot\text{GPR}}$	DATA = ØR·GPR minus 1	DATA = ØR·GPR
L	L	H	L	DATA = $\overline{\text{ØR}}$ + GPR	DATA = ØR·$\overline{\text{GPR}}$ minus 1	DATA = ØR·$\overline{\text{GPR}}$
L	L	H	H	DATA = 1	DATA = minus 1 (2's COMP)	DATA = ZERO
L	H	L	L	DATA = $\overline{\text{ØR + GPR}}$	DATA = ØR plus (ØR + $\overline{\text{GPR}}$)	DATA = ØR plus (ØR + $\overline{\text{GPR}}$) plus 1
L	H	L	H	DATA = $\overline{\text{GPR}}$	DATA = ØR·GPR plus (ØR + $\overline{\text{GPR}}$)	DATA = ØR·GPR plus (ØR + $\overline{\text{GPR}}$) plus 1
L	H	H	L	DATA = $\overline{\text{ØR} \oplus \text{GPR}}$	DATA = ØR minus GPR minus 1	DATA = ØR minus GPR
L	H	H	H	DATA = ØR + $\overline{\text{GPR}}$	DATA = ØR + $\overline{\text{GPR}}$	DATA = (ØR + $\overline{\text{GPR}}$) plus 1
H	L	L	L	DATA = $\overline{\text{ØR}}$·GPR	DATA = ØR plus (ØR + GPR)	DATA = ØR plus (ØR + $\overline{\text{GPR}}$) plus 1
H	L	L	H	DATA = ØR $\oplus$ GPR	DATA = ØR plus GPR	DATA = ØR plus GPR plus 1
H	L	H	L	DATA = GPR	DATA = ØR·$\overline{\text{GPR}}$ plus (ØR + GPR)	DATA = ØR·$\overline{\text{GPR}}$ plus (ØR + GPR) plus 1
H	L	H	H	DATA = ØR + GPR	DATA = ØR + GPR	DATA = (ØR + GPR) plus 1
H	H	L	L	DATA = 0	DATA = ØR plus ØR**	DATA = ØR plus ØR plus 1
H	H	L	H	DATA = ØR·$\overline{\text{GPR}}$	DATA = ØR.GPR plus ØR	DATA = ØR·GPR plus ØR plus 1
H	H	H	L	DATA = ØR·GPR	DATA = ØR·$\overline{\text{GPR}}$ plus ØR	DATA = ØR·$\overline{\text{GPR}}$ plus ØR plus 1
H	H	H	H	DATA = ØR	DATA = ØR	DATA = ØR plus 1

*ØR = Output register data before the ALU is loaded into the output register.
DATA = Output register data after the ALU data is loaded into the output register.
GPR = General purpose register 1, 2, 3 or 4 as selected by the ALU instruction.

** Each bit is shifted to the next more significant position.

4.4 PROGRAM EXAMPLES

In this section, each Example describes a problem to be solved, presents timing diagrams (if applicable),and shows the interconnection of signals between the RS-432 and the UUT. The description also includes a flow chart of program steps and a detailed listing of program instructions that can be entered into the RS-432 and executed. Do to some instruction changes for the RSM option these programs may not operate as stated on an RSM-432.

The signal mnemonics appearing in the block diagram that show signal interconnection between the RS-432 and the UUT correspond to the signals appearing in the Input/Output Connector Pin Assignments found in the Appendix. Prior to programming, a decision must be made concerning the assignment of these RS-432 I/O signals to corresponding I/O signals for the UUT.

Symbology used in the flow charts in straightforward:

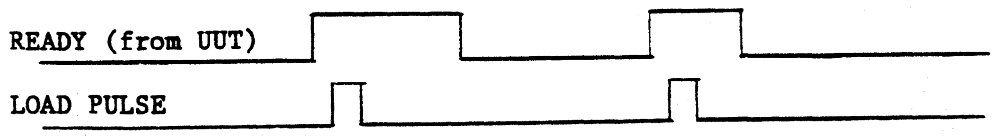
1. Rectangular boxes represent executable instructions requiring no decision-making.
2. The address at which this instruction resides in program memory is denoted by the octal number in the upper left-hand corner of the box.
3. The instruction's mnemonic abbreviation appears in the lower left-hand corner.
4. Decision-making instructions are represented by the appendaged rectangular box. This symbol normally has two exit points--one if the test condition is true, the other if it is false. Address location and mnemonic abbreviation appear in the same relative positions in the box as described above.

The program coding format is an abbreviated version of the program coding sheets presented in the Appendix. These sheets provide a convenient method of documenting the individual ones and zeroes that represent the instruction to be executed.

PROGRAM EXAMPLE 1: ONE PULSE

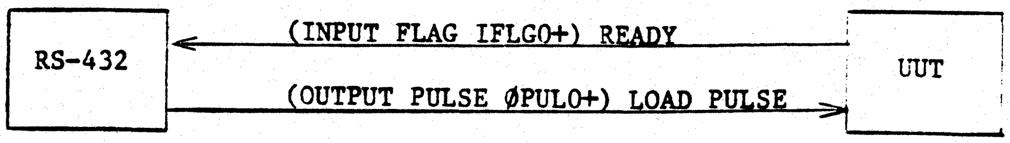
When a ready signal from the UUT goes high, generate one load pulse.

Timing Diagram

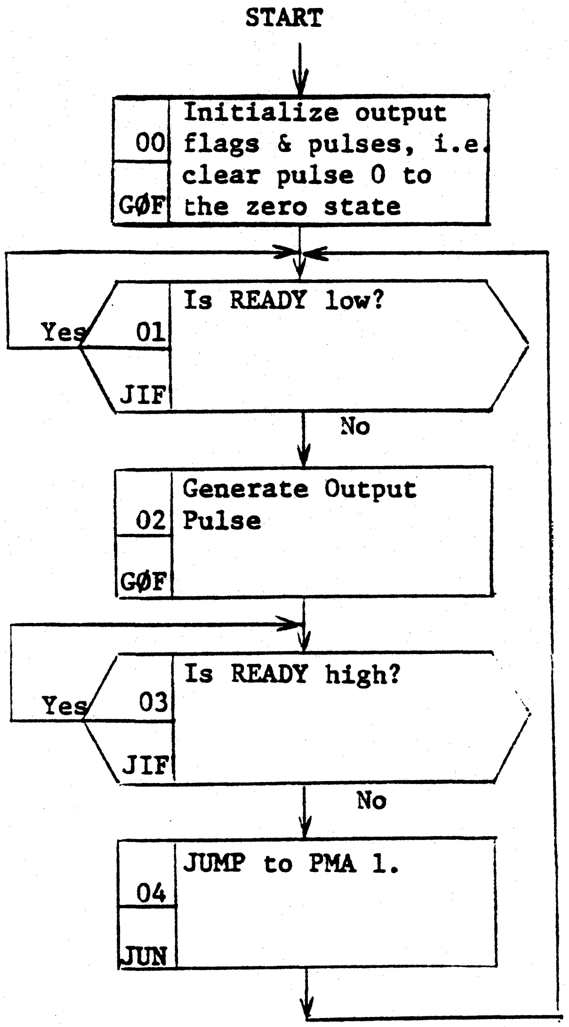


PROGRAM EXAMPLE 1: ONE PULSE (Cont'd)

Block Diagram



Flow Chart



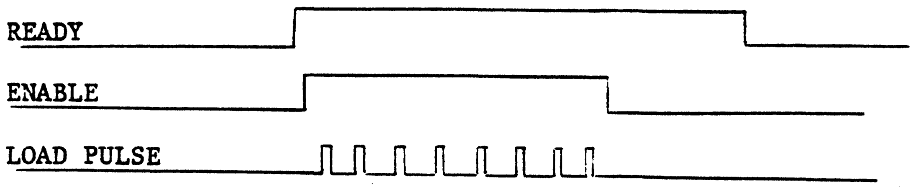
Program Coding

ADDRESS OCTAL	MNEMONIC	INSTRUCTION															
		OP CODE				DATA FIELD											
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
000	GØF	0	1	1	0	X	X	0	0	X	X	X	X	X	X	X	X
001	JIF	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
002	GØF	0	1	1	0	X	X	0	1	X	X	X	X	X	X	X	X
003	JIF	0	0	0	1	0	0	1	0	0	0	0	0	0	0	1	1
004	JUN	0	1	0	0	X	X	X	0	0	0	0	0	0	0	0	1

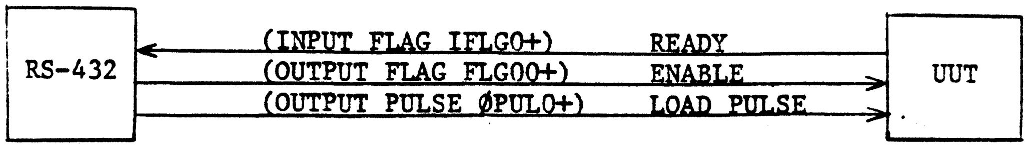
PROGRAM EXAMPLE 2: EIGHT PULSES

When a ready signal from the UUT goes high, generate an enable signal, followed by eight pulses. Each load pulse is 100 nanoseconds wide and spaced 600 nanoseconds apart.

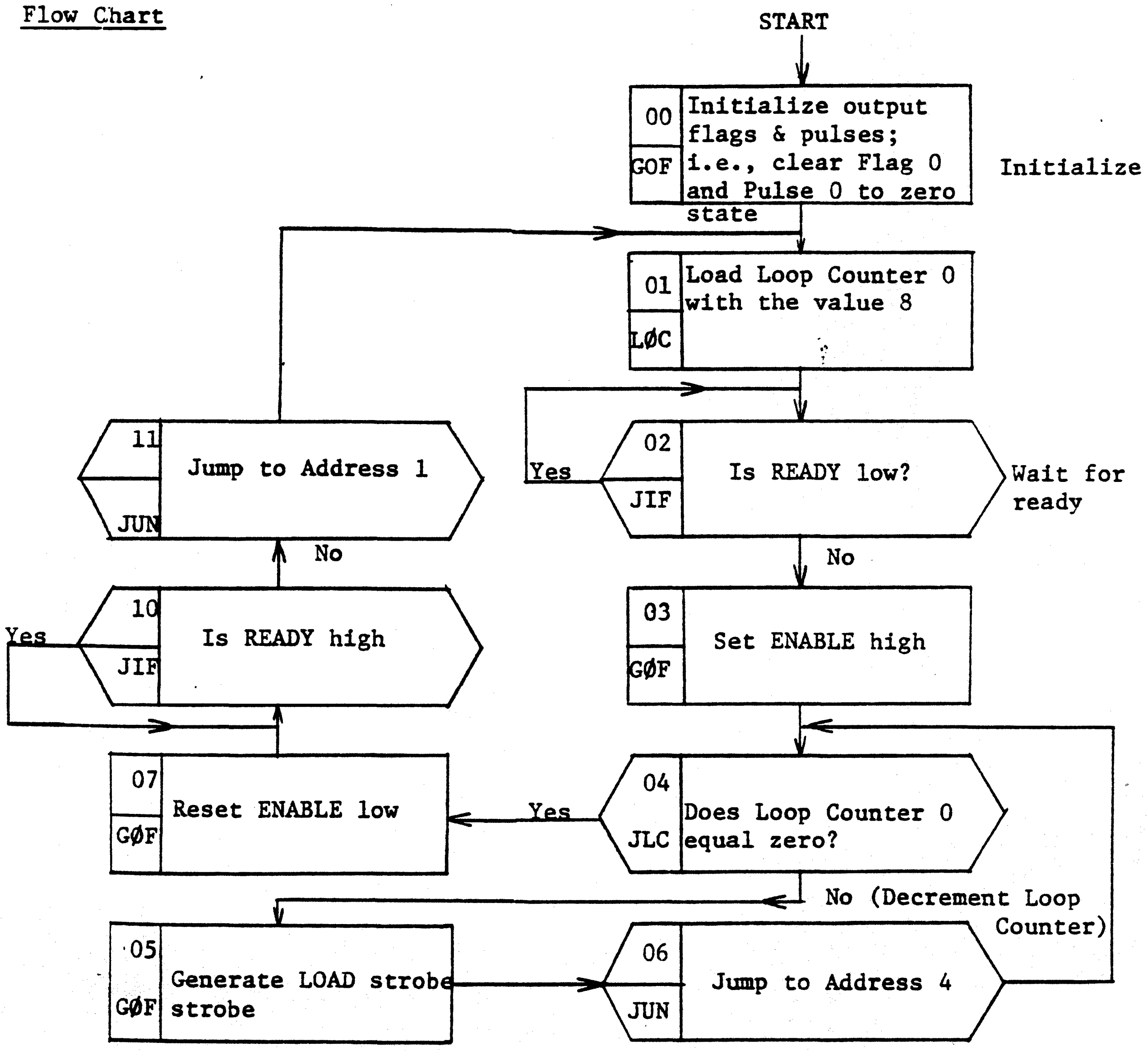
Timing Diagram



Block Diagram



Flow Chart


START
00
GOF
Initialize output flags & pulses; i.e., clear Flag 0 and Pulse 0 to zero state
Initialize
01
LØC
Load Loop Counter 0 with the value 8
02
JIF
Is READY low?
Yes
No
Wait for ready
03
GØF
Set ENABLE high
04
JLC
Does Loop Counter 0 equal zero?
Yes
No (Decrement Loop Counter)
05
GØF
Generate LOAD strobe strobe
06
JUN
Jump to Address 4
07
GØF
Reset ENABLE low
10
JIF
Is READY high
Yes
No
11
JUN
Jump to Address 1

PROGRAM EXAMPLE 2: EIGHT PULSES (Cont'd)

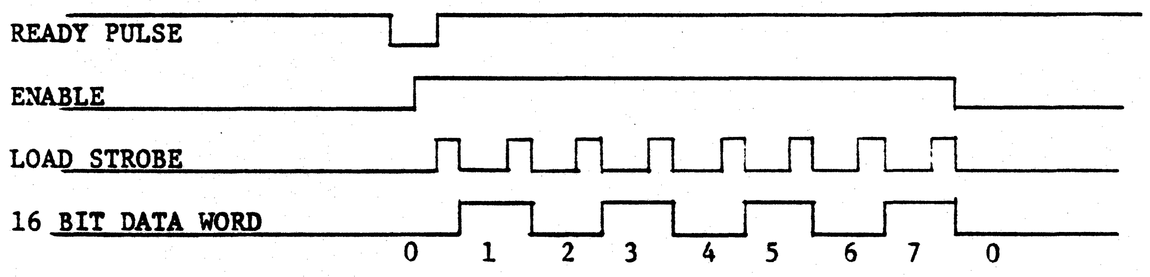
Program Coding

ADDRESS OCTAL	MNEMONIC	INSTRUCTION																
		OP CODE				DATA FIELD												
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
000	GØF	0	1	1	0	X	X	0	0	X	X	X	X	X	X	X	0	INITIALIZ
001	LØC	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	
002	JIF	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	WAIT
003	GØF	0	1	1	0	X	X	0	0	X	X	X	X	X	X	X	1	GENERATE TIMING
004	JLC	0	0	1	1	0	0	1	0	0	0	0	0	0	1	1	1	
005	GØF	0	1	1	0	X	X	0	1	X	X	X	X	X	X	X	1	
006	JUN	0	1	0	0	X	X	X	0	0	0	0	0	0	1	0	0	
007	GØF	0	1	1	0	X	X	0	0	X	X	X	X	X	X	X	0	
010	JIF	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	
011	JUN	0	1	0	0	X	X	X	0	0	0	0	0	0	0	0	1	DONE

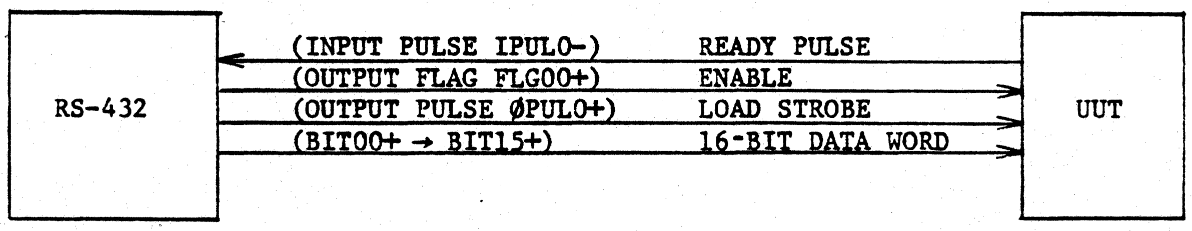
PROGRAM EXAMPLE 3: EIGHT PULSES WITH DATA

When a ready pulse from the UUT occurs, generate an enable signal followed by eight load pulses. With each load pulse, also generate a unique 16-bit parallel word. (In this example, the microprocessor will control the word generator to generate the required signals.)

Timing Diagram

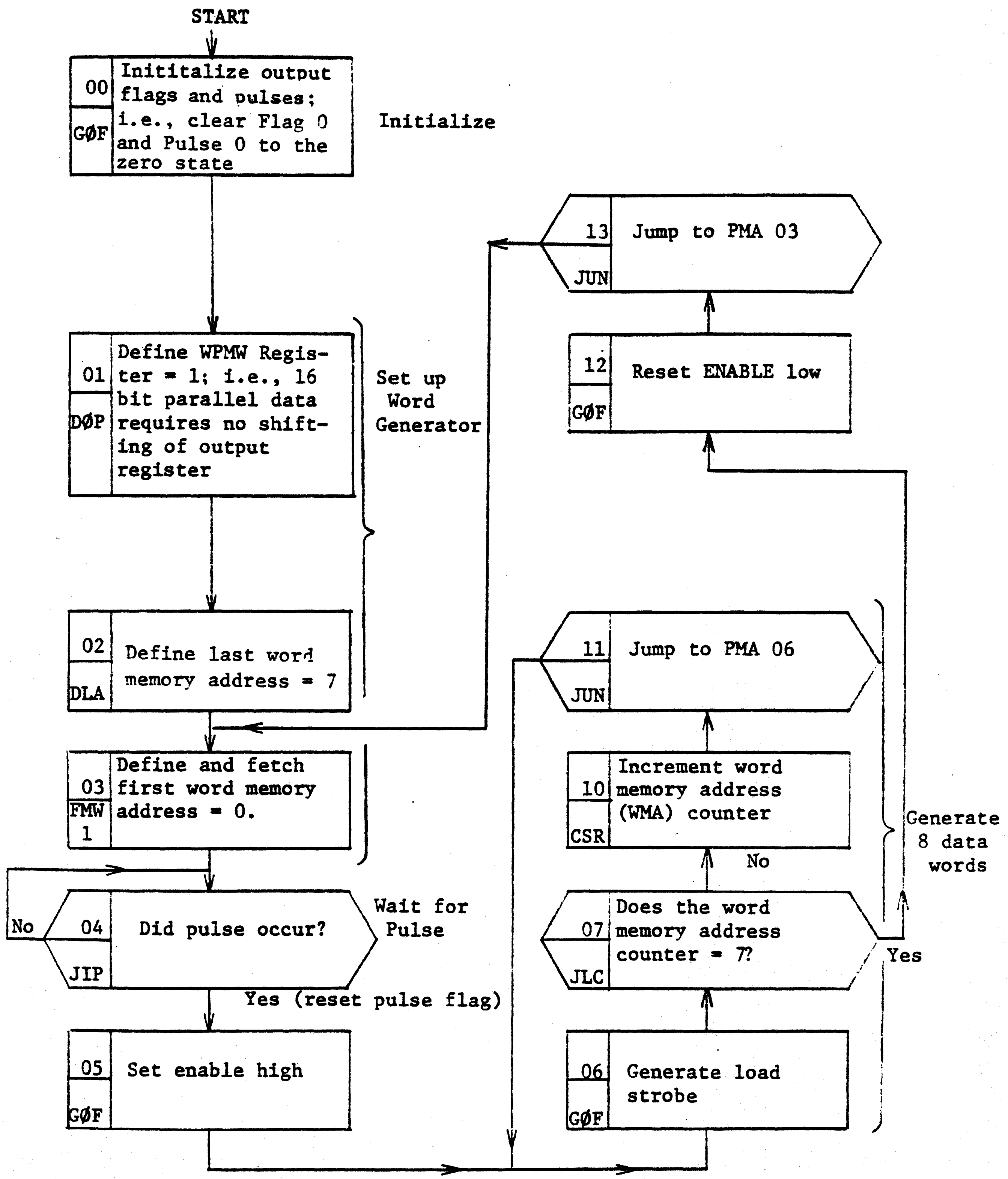


Block Diagram



PROGRAM EXAMPLE 3: EIGHT PULSES WITH DATA (Cont'd)

Flow Chart


START
00 GØF Inititalize output flags and pulses; i.e., clear Flag 0 and Pulse 0 to the zero state
Initialize
01 DØP Define WPMW Register = 1; i.e., 16 bit parallel data requires no shifting of output register
02 DLA Define last word memory address = 7
Set up Word Generator
03 FMW 1 Define and fetch first word memory address = 0.
04 JIP Did pulse occur?
No
Wait for Pulse
Yes (reset pulse flag)
05 GØF Set enable high
06 GØF Generate load strobe
07 JLC Does the word memory address counter = 7?
No
Yes
10 CSR Increment word memory address (WMA) counter
11 JUN Jump to PMA 06
Generate 8 data words
12 GØF Reset ENABLE low
13 JUN Jump to PMA 03

PROGRAM EXAMPLE 3: EIGHT PULSES WITH DATA (Cont'd)

Program Coding

ADDRESS	MNEMONIC	INSTRUCTION																
		OP CODE				DATA FIELD												
OCTAL		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
000	GØF	0	1	1	0	X	X	0	0	X	X	X	X	X	X	X	0	INITIALIZE
001	DØP	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	SET UP WORD GENERATOR
002	DLA	1	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	
003	FMW1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	
004	JIP	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	WAIT
005	GØF	0	1	1	0	X	X	0	0	X	X	X	X	X	X	X	1	GENERATE 8 DATA WORD & TIMING
006	GØF	0	1	1	0	X	X	0	1	X	X	X	X	X	X	X	1	
007	JLC	0	0	1	1	1	1	1	0	0	0	0	0	1	0	1	0	
010	CSR	1	1	0	1	X	X	X	X	X	X	X	X	X	X	X	X	
011	JUN	0	1	0	0	X	X	X	0	0	0	0	0	0	1	1	0	
012	GØF	0	1	1	0	X	X	0	0	X	X	X	X	X	X	X	0	
013	JUN	0	1	0	0	X	X	X	0	0	0	0	0	0	0	1	1	DONE

Word Memory Coding

	16 BIT DATA WORD
W.M.A.	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
0000	TEST WORD 0
0001	TEST WORD 1
0002	TEST WORD 2
0003	TEST WORD 3
0004	TEST WORD 4
0005	TEST WORD 5
0006	TEST WORD 6
0007	TEST WORD 7
0010	DON'T CARE
'	DON'T CARE
'	'
'	'

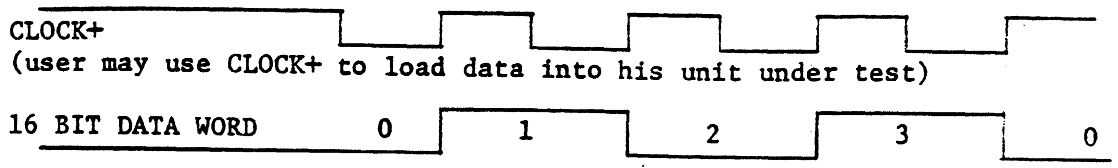
PROGRAM EXAMPLE 4: WORD GENERATOR WITH STL INSTRUCTION

Using an STL instruction, program the RS-432 to transfer four 16-bit data words from the word memory at a 10-usec transfer rate. The program will transfer the four words and halt.

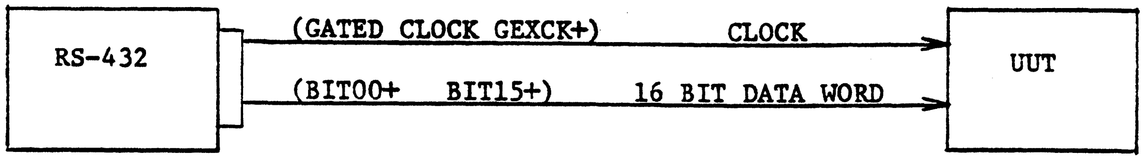
This example will generate the required data using the Start Word Generator Loop (STL) instruction. Program Example 5 generates the same data pattern, using a Clock Shift Register (CSR) program loop routine.

PROGRAM EXAMPLE 4: WORD GENERATOR WITH STL INSTRUCTION (Cont'd)

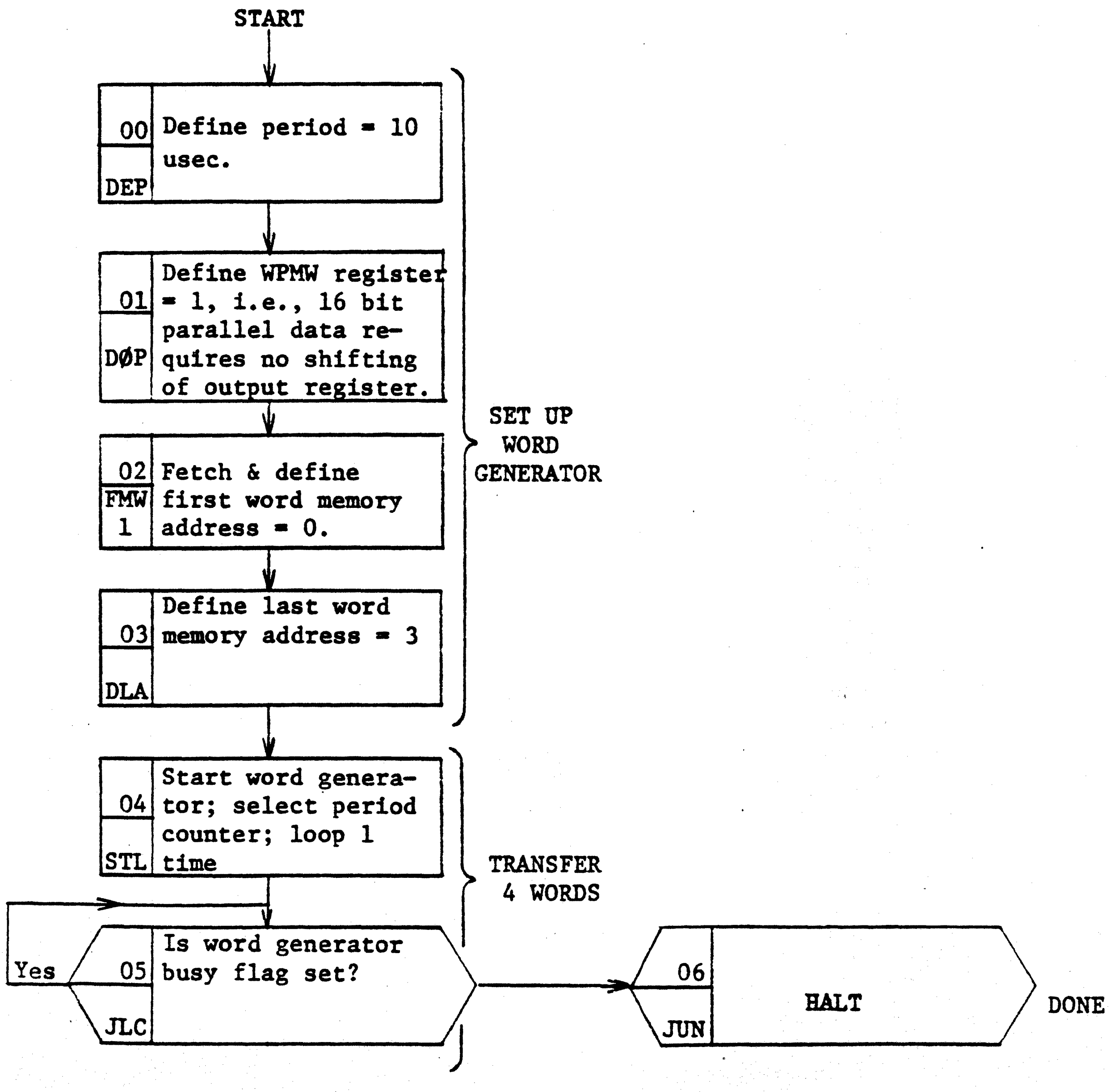
Timing Diagram



Block Diagram



Flow Chart


START
00
DEP
Define period = 10 usec.
01
DØP
Define WPMW register = 1, i.e., 16 bit parallel data requires no shifting of output register.
02
FMW 1
Fetch & define first word memory address = 0.
03
DLA
Define last word memory address = 3
SET UP WORD GENERATOR
04
STL
Start word generator; select period counter; loop 1 time
TRANSFER 4 WORDS
Yes
05
JLC
Is word generator busy flag set?
06
JUN
HALT
DONE

PROGRAM EXAMPLE 4: WORD GENERATOR WITH STL INSTRUCTION (Cont'd)

Program Coding

ADDRESS OCTAL	MNEMONIC	INSTRUCTION																
		OP CODE				DATA FIELD												
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
00	DEP	1	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0	SET UP
01	DØP	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	WORD
02	FMW_1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	GENERATOR
03	DLA	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	
04	STL	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	TRANSFER
05	JLC	0	0	1	1	1	0	1	0	0	0	0	0	0	1	0	1	4 WORDS
06	JUN	0	1	0	0	X	X	X	0	0	0	0	0	0	1	1	0	DONE
07																		

Word Memory Coding

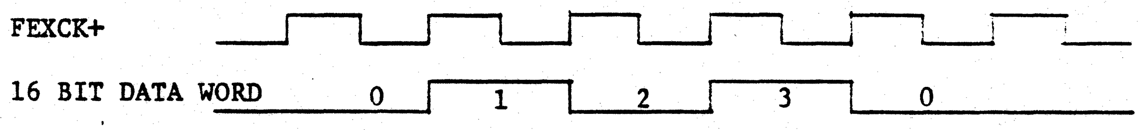
W.M.A.	16 BIT DATA WORD 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
0000	TEST WORD 0
0001	TEST WORD 1
0002	TEST WORD 2
0003	TEST WORD 3
0004	DON'T CARE
0005	DON'T CARE

PROGRAM EXAMPLE 5: WORD GENERATOR WITH CSR PROGRAM LOOP

Program the RS-432 to transfer four 16-bit data words from the word memory at a 10-usec transfer rate. The program will transfer the four data words and halt.

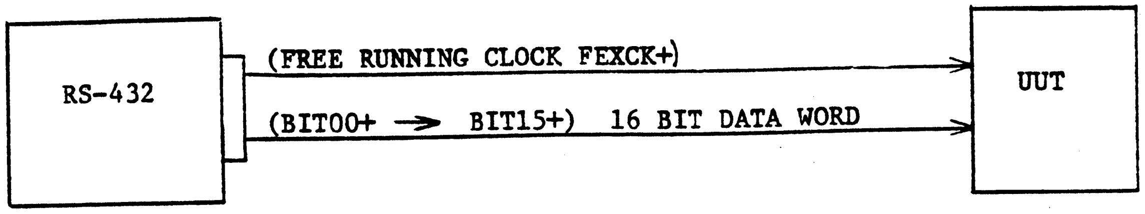
The example will generate the required data using a Clock Shift Register (CSR) program loop. Program Example 4 generates the same data patterns using the STL instruction. Note that this program approach will not generate the GEXCK+ clock signals.

Timing Diagram

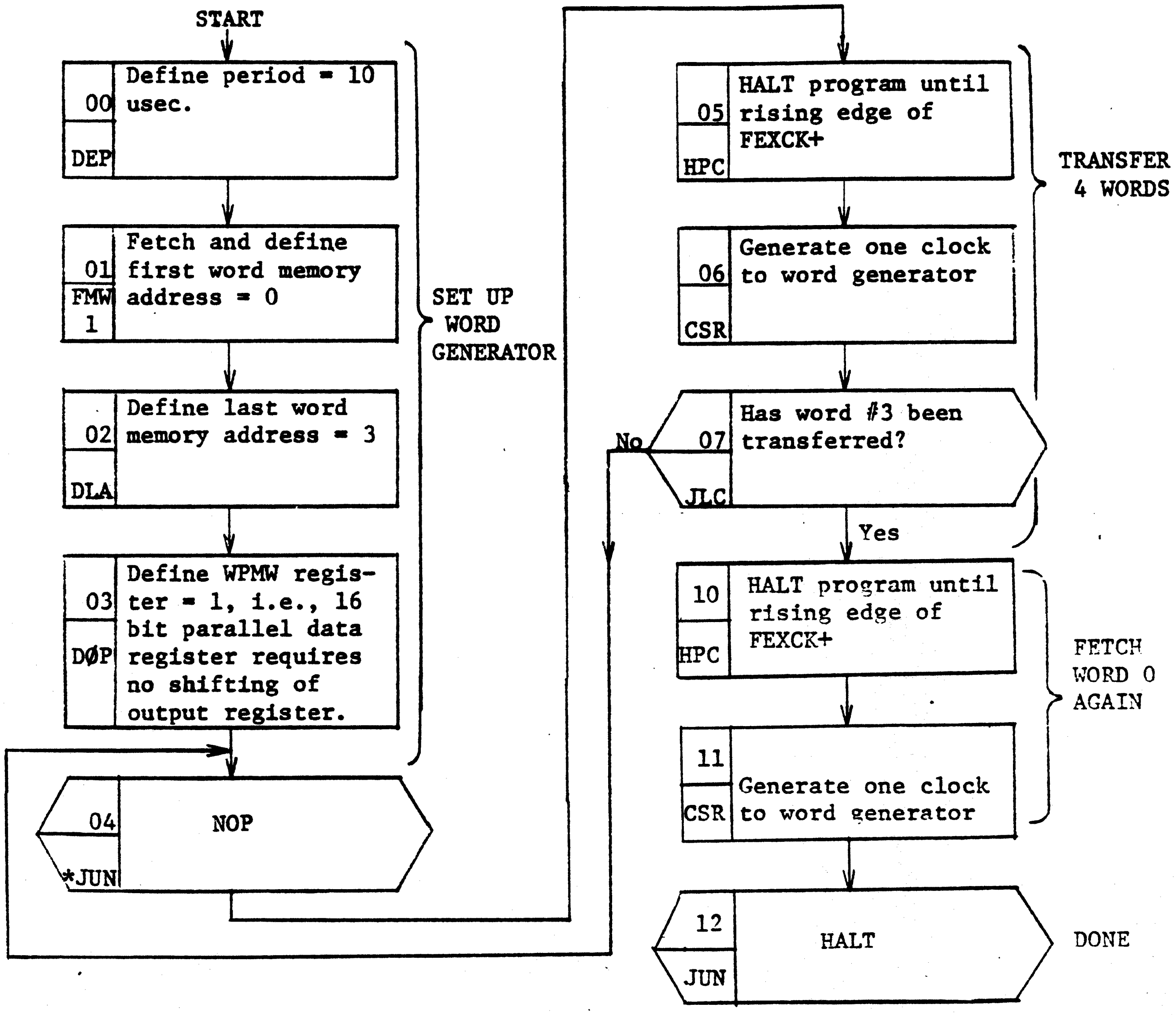


PROGRAM EXAMPLE 5: WORD GENERATOR WITH CSR PROGRAM LOOP (Cont'd)

Block Diagram


RS-432
(FREE RUNNING CLOCK FEXCK+)
(BIT00+ → BIT15+) 16 BIT DATA WORD
UUT

Flow Chart


START
00 DEP Define period = 10 usec.
01 FMW 1 Fetch and define first word memory address = 0
02 DLA Define last word memory address = 3
03 DØP Define WPMW register = 1, i.e., 16 bit parallel data register requires no shifting of output register.
SET UP WORD GENERATOR
04 *JUN NOP
05 HPC HALT program until rising edge of FEXCK+
06 CSR Generate one clock to word generator
TRANSFER 4 WORDS
07 JLC Has word #3 been transferred?
No
Yes
10 HPC HALT program until rising edge of FEXCK+
11 CSR Generate one clock to word generator
FETCH WORD 0 AGAIN
12 JUN HALT
DONE

*This instruction does nothing. However, if an output timing routine is required to transfer the data word, this location could store a jump instruction to that routine. Note that the program has slightly less than 10 usec to generate the output timing.

PROGRAM EXAMPLE 5: WORD GENERATOR WITH CSR PROGRAM LOOP (Cont'd)

Program Coding

ADDRESS OCTAL	MNEMONIC	INSTRUCTION																
		OP CODE				DATA FIELD												
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
00	DEP	1	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0	SET UP
01	FMW_1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	WORD
02	DLA	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	GENERATOR
03	DØP	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	
04	JUN	0	1	0	0	X	X	X	0	0	0	0	0	0	1	0	1	
05	HPC	0	1	1	1	X	X	X	X	X	X	X	X	X	X	X	X	TRANSFER
06	CSR	1	1	0	1	X	X	X	X	X	X	X	X	X	X	X	X	4 WORDS
07	JLC	0	0	1	1	1	1	0	0	0	0	0	0	0	1	0	0	
10	HPC	0	1	1	1	X	X	X	X	X	X	X	X	X	X	X	X	FETCH WORD
11	CSR	1	1	0	1	X	X	X	X	X	X	X	X	X	X	X	X	0 AGAIN
12	JUN	0	1	0	0	X	X	X	0	0	0	0	0	1	0	1	0	DONE

PROGRAM EXAMPLE 6: PROGRAM TIMING CONSIDERATIONS

This example presents certain timing considerations that the user should be aware of. Assume an output flag is tied directly into an input flag, and that a program is written which first sets the output flag high and the very next instruction tests to see whether the input flag is high.

It would seem that the jump instruction would cause a jump to occur since the previous instruction set the output flag high; however, this is not the case, the reason being as follows: The logic in the microprocessor decodes each instruction in 200 nsec. For every instruction except the jump instructions, execution of an instruction is performed exactly at the end of the 200 nsec decode time (i.e., at the beginning of the decode time of the next instruction). However, the jump instructions sample the condition to be treated at the beginning of their respective instruction decode time and store the sampled condition throughout the decode time. Thus, as shown in Figure 25, the JIF would sample the flag when it was still a zero and the jump would not occur.

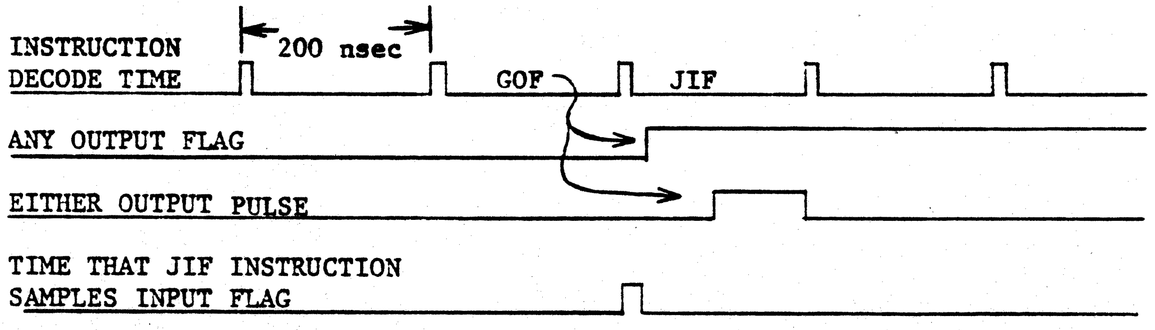


Figure 22. PROGRAM TIMING

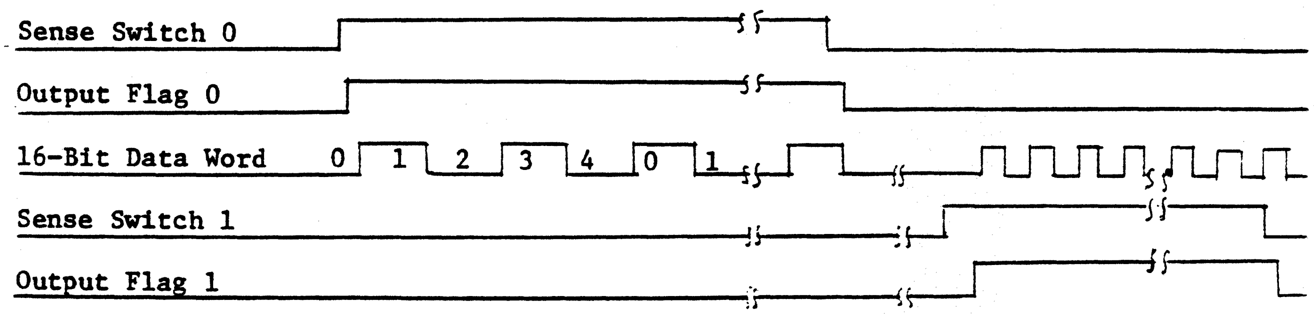
This timing consideration should be kept in mind when an output flag or pulse causes a condition to occur in the user's UUT which is then tested by the RS-432.

The sequence of instructions for testing the response to an output pulse or flag should be given careful attention. For example, insertion of a NOP or a series of NOP's might be required to allow enough time for a response to be returned. In this case, the program would essentially delay, allowing sufficient time for the response. Then it would check the response and branch to one location if it had occurred and branch to another if it had not. If, on the other hand, the program should not proceed until the proper response is returned, the jump instruction can be coded to test continuously for the proper condition. This is accomplished by making the jump address of the jump instruction the same as the address of the instruction. In this case, the jump instruction would continually be executed until the test condition allowed it to progress to the next instruction.

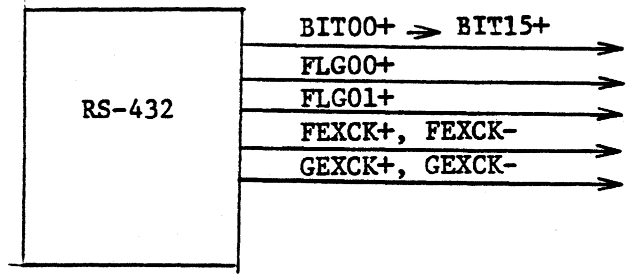
PROGRAM EXAMPLE 7: PARALLEL DATA TRANSFER

This example is a follow-up to the RS-432 example on Page 20. It will generate five 16-bit words in parallel at two different rates dependent upon sense switch settings. Sense Switch 0 will cause 300 millisecond data rates and Sense Switch 1 will cause 1 microsecond data rates.

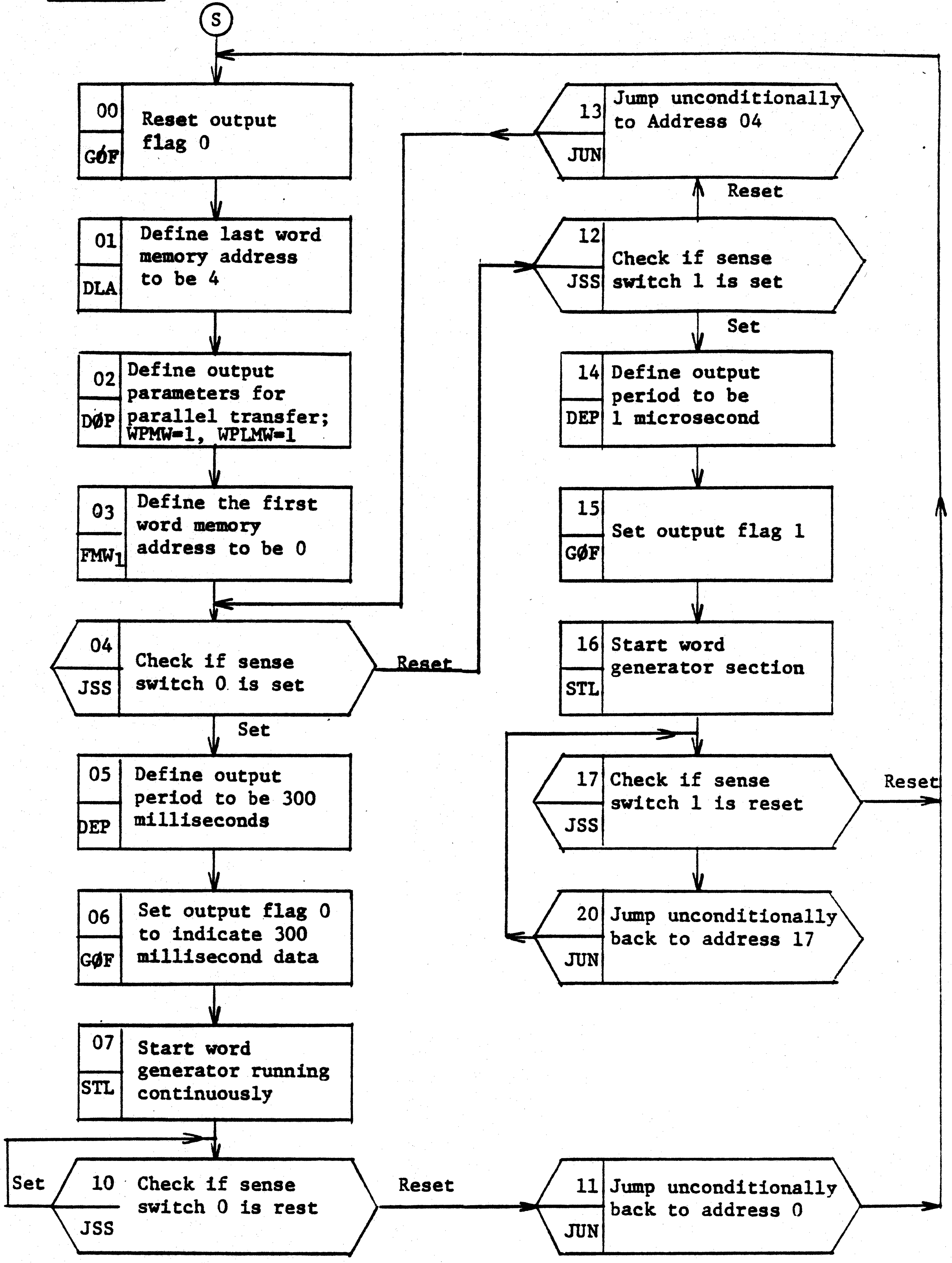
Timing Diagram



Block Diagram



Flow Chart


S
00 GØF Reset output flag 0
01 DLA Define last word memory address to be 4
02 DØP Define output parameters for parallel transfer; WPMW=1, WPLMW=1
03 FMW1 Define the first word memory address to be 0
04 JSS Check if sense switch 0 is set
Reset
Set
05 DEP Define output period to be 300 milliseconds
06 GØF Set output flag 0 to indicate 300 millisecond data
07 STL Start word generator running continuously
Set
10 JSS Check if sense switch 0 is rest
Reset
11 JUN Jump unconditionally back to address 0
13 JUN Jump unconditionally to Address 04
Reset
12 JSS Check if sense switch 1 is set
Set
14 DEP Define output period to be 1 microsecond
15 GØF Set output flag 1
16 STL Start word generator section
17 JSS Check if sense switch 1 is reset
Reset
20 JUN Jump unconditionally back to address 17

PROGRAM EXAMPLE 7: PARALLEL DATA TRANSFER (Cont'd)

Program Coding

RS-432 PROGRAMMING SHEET (CODING)

PROGRAM NAME: RS-432 USER's GUIDE EXAMPLE 7.

ADDRESS OCTAL		INSTRUCTION																
		MNEMONIC	OP CODE				DATA FIELD											
			15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	GØF	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	DLA	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0
0	2	DØP	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1
0	3	FMWI	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	4	JSS	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
0	5	DEP	1	0	0	0	0	0	0	1	0	0	1	0	1	1	0	0
0	6	GØF	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1
0	7	STL	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0
1	0	JSS	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0
1	1	JUN	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	2	JSS	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0
1	3	JUN	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	4	DEP	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
1	5	GØF	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0
1	6	STL	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0
1	7	JSS	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
2	0	JUN	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1
2	1																	
2	2																	
2	3																	
2	4																	
2	5																	
2	6																	
2	7																	

Word Memory Coding

RS-432 WORD MEMORY PROGRAMMING SHEET

PROGRAM NAME: RS-432 USER's GUIDE EXAMPLE 7.

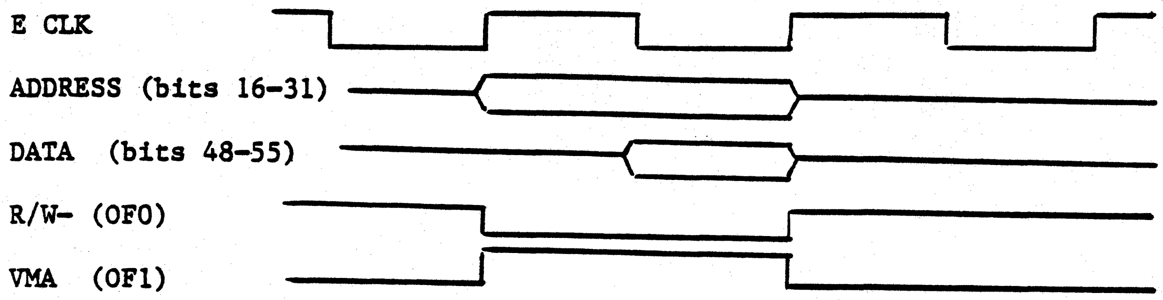
ADDRESS (OCTAL)		DATA															
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1
0	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0
0	2	1	0	1	1	0	0	0	1	1	0	0	0	1	1	0	1
0	3	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0
0	4	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	0
0	5																
0	6																
0	7																

Data Patte

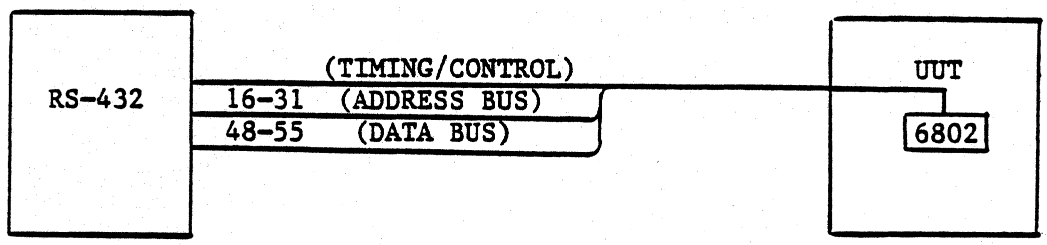
PROGRAM EXAMPLE 8: TIMING CYCLE

This example shows how to simulate a microprocessor bus cycle. This particular example depicts the timing for a single write cycle of a Motorola 6802 microprocessor. Instructions show only the bus cycle timing. Additional instructions are required to fetch and increment the data memories (example 5). (Note instruction 107 (JIF) tests the address and data bus for shorts to ground, five volts or another pin, reference Input Compare 4.3.3.)

Timing Diagram



Block Diagram



Program Coding

ADDRESS OCTAL	MNEMONIC	OP CODE 15	14	13	12	DATA FIELD 11	10	9	8	7	6	5	4	3	2	1	0
100	DEP	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
101	GOF	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1
102	HPC	0	1	1	1	X	X	X	X	X	X	X	X	X	X	X	X
103	CSR	1	1	0	1	X	X	X	X	X	X	X	X	X	X	X	X
104	HPC	0	1	1	1	X	X	X	X	X	X	X	X	X	X	X	X
105	GOF	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0
106	HPC	0	1	1	1	X	X	X	X	X	X	X	X	X	X	X	X
107	JIF	0	0	0	1	0	0	0	0	(ERROR ADDR)							
110	GOF	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1
111	CSR	1	1	0	1	X	X	X	X	X	X	X	X	X	X	X	X

* APPENDIX A. CONDENSED INSTRUCTION SET

MICROPROCESSOR JUMP INSTRUCTIONS

INSTRUCTION (MNEMONIC)	OP CODE 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	DESCRIPTION

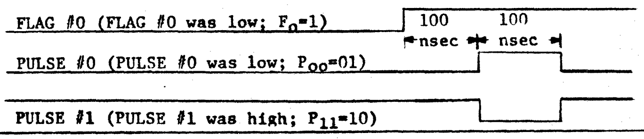
Jump on sense switch
JSS
	0 0 0 0 SS T J JA	Jump to JUMP ADDRESS if switch SS is in position defined by T.
		T = 0; test for lower pos.
		T = 1; test for upper pos.
		J = 0; JUMP ADDRESS = JA (binary).
		J = 1; JUMP ADDRESS = 8 LSB's of input register.
		SS=00; sense switch 0
		SS=01; sense switch 1
		SS=10; sense switch 2
		SS=11; sense switch 3

Jump on input flag
JIF
	0 0 0 1 IF T J JA	Jump to JUMP ADDRESS if input flag IF is in state defined by T.
		T = 0; test for 0 (0 volts).
		T = 1; test for 1 (≥2.4 volts).
		J = 0;1 (JUMP ADDRESS same as JSS instruction.)
		IF=00; input flag 0
		IF=01; input flag 1
		IF=10; input flag 2
		IF=11; input flag 3

Jump on input pulse
JIP
	0 0 1 0 IP T J JA	Jump to JUMP ADDRESS if input pulse IP is in state defined by T.
		T = 0; pulse did not occur.
		T = 1; pulse did occur.
		J = 0;1 (JUMP ADDRESS same as JSS instruction.)
		IP=00; input pulse 0
		IP=01; input pulse 1
		IP=10; input pulse 2
		IP=11; input pulse 3

Jump on loop count JLC	0 0 1 1 LC T J JA	Jump to JUMP ADDRESS if the conditions of LC and T are as defined below. LC=00 < T=0; jump if loop counter #0 ≠ 0 (not done). T=1; jump if loop counter #0 = 0 (done). LC=01 < T=0; jump if loop counter #1 ≠ 0 (not done). T=1; jump if loop counter #1 = 0 (done). (This instruction, with LC=00 or 01, automatically decrements the respective loop counter after execution of instruction.) LC=10 < T=0; jump if word generator busy flag not set. T=1; jump if word generator busy flag set. (Reference STL instruction.) LC=11 < T=0; jump if LMA • SCF flag not set. T=1; jump if LMA • SCF flag set. (LMA = last (word) memory address.) (SCF = WPMW shifts complete flag.) J = 0;1 (JUMP ADDRESS same as JSS instruction.)
Unconditional jump JUN	0 1 0 0 J JA	Unconditionally jump to JUMP ADDRESS. Note; Used as HALT instruction if JUMP ADDRESS = instruction address. Used as NOP instruction if JUMP ADDRESS = instruction address + 1. J = 0;1 (JUMP ADDRESS same as JSS instruction.)

MICROPROCESSOR CONTROL INSTRUCTIONS

Load loop counter LOC	0 1 0 1 L LOOP CNT	Load loop counter #0 (L=0) or loop counter #1 (L=1) with data defined by LOOP CNT. (LOOP CNT data is binary.)
Generate output flags GOF	0 1 1 0 P_1P_1 P_0P_0 F_7 F_6 F_5 F_4 F_3 F_2 F_1 F_0	Set output flag F_n high (≥2.4 volts) if F_n=1. Reset output flag F_n low (0 volts) if F_n=0. (n=0,1,2,.....7) P_0P_0=00; reset output pulse #0 to 0 (0 volts). P_0P_0=01; pulse output pulse #0 high for 100 nsec. P_0P_0=10; pulse output pulse #0 low for 100 nsec. P_0P_0=11; set output pulse #0 to 1 (≥2.4 volts). P_1P_1 (output pulse #1 is the same as output pulse #0.) **FLAG TIMING**  FLAG #0 (FLAG #0 was low; F_0=1) 100 nsec 100 nsec PULSE #0 (PULSE #0 was low; P_{00}=01) PULSE #1 (PULSE #1 was high; P_{11}=10)
Halt until period complete HPC	0 1 1 1	Halts program execution until the rising edge of the free running period counter clock. This instruction should be followed by a load or shift of output data if data and period clock are to be in sync.

* Some instructions have been changed for RSM option; ref: Appendix I

APPENDIX B. INPUT/OUTPUT CONNECTOR

SIGNAL DESCRIPTION

OUTPUT CONNECTOR** PIN #	SIGNAL MNEMONIC AND DESCRIPTION		OUTPUT CON PIN # (TWISTED GROUND RE
1	BIT00+	OUTPUT DATA BIT 00	21
2	BIT01+	" " " 01	22
3	BIT02+	" " " 02	23
4	BIT03+	" " " 03	24
5	BIT04+	" " " 04	25
6	BIT05+	" " " 05	26
7	BIT06+	" " " 06	27
8	BIT07+	" " " 07	28
9	BIT08+	" " " 08	29
10	BIT09+	" " " 09	30
11	BIT10+	" " " 10	31
12	BIT11+	" " " 11	32
13	BIT12+	" " " 12	33
14	BIT13+	" " " 13	34
15	BIT14+	" " " 14	35
16	BIT15+	" " " 15	36
17	FEXCK+	FREE RUNNING CLOCK: PERIOD DEFINED BY DEP INSTRUCTION	37
18	FEXCK-	INVERTED FREE RUNNING CLOCK	38
19	GEXCK+	GATED CLOCK: ONE CLOCK FOR EACH WORD MEMORY WORD	39
20	GEXCK-	INVERTED GATED CLOCK	59
40	FLG00+	OUTPUT FLAG #0; SET HIGH OR LOW BY PROGRAM	60
41	FLG01+	" " #1; " " " " " "	61
42	FLG02+	" " #2; " " " " " "	62
43	FLG03+	" " #3; " " " " " "	63
44	FLG04+	" " #4; " " " " " "	64
45	FLG05+	" " #5; " " " " " "	65
46	FLG06+	" " #6; " " " " " "	66
47	FLG07+	" " #7; " " " " " "	67
48	ØPUL0+	OUTPUT PULSE #0; PULSED HIGH OR LOW BY PROGRAM	68
49	ØPUL1+	" " #9; " " " " " "	69
*50	IFLG0+	INPUT FLAG #0; PROGRAM MAY SENSE THIS FLAG	70
*51	IFLG1+	" " #1; " " " " "	71
*52	IFLG2+	" " #2; " " " " "	72
*53	IFLG3+	" " #3; " " " " "	73
*54	IPUL0-	INPUT PULSE #0; PROGRAM MAY TEST OCCURANCE OF PULSE	74
*55	IPUL1-	" " #1; " " " " " "	75
*56	IPUL2-	" " #2; " " " " " "	76
*57	IPUL3-	" " #3; " " " " " "	77
58	GINCK-	GATED INCOMING CLOCK: EXTERNAL DEVICE MAY CLOCK WORD GENERATOR WITH THIS CLOCK.	78

* Some input flags and pulses may be designated for particular options

** 78 Pin Male Connector (mounted on back panel)
AMP series HD-22 part #204509-1

78 Pin Female Connector (attaches to UUT cable)
AMP series HD-22 part #204508-1

APPENDIX C. RS-432 OUTPUT CONNECTOR

OUTPUT DATA BITS 16-31

OUTPUT CONNECTOR * PIN #	SIGNAL MNEMONIC AND DESCRIPTION		OUTPUT CONNECTOR * PIN # (TWISTED PAIR GROUND RETURN)
1	BIT16+	OUTPUT DATA BIT 16	20
2	BIT17+	" " " 17	21
3	BIT18+	" " " 18	22
4	BIT19+	" " " 19	23
5	BIT20+	" " " 20	24
6	BIT21+	" " " 21	25
7	BIT22+	" " " 22	26
8	BIT23+	" " " 23	27
9	BIT24+	" " " 24	28
10	BIT25+	" " " 25	29
11	BIT26+	" " " 26	30
12	BIT27+	" " " 27	31
13	BIT28+	" " " 28	32
14	BIT29+	" " " 29	33
15	BIT30+	" " " 30	34
16	BIT31+	" " " 31	35

* 37 Pin Male Connector (mounted on back panel)
AMP series 20 part #205210-1

37 Pin Female Connector (attaches to UUT cable)
AMP series 20 part #205209-1

APPENDIX D. RS-432 OUTPUT CONNECTOR

OUTPUT DATA BITS 32-63

OUTPUT CONNECTOR* PIN #	SIGNAL MNEMONIC AND DESCRIPTION		OUTPUT CONNECTOR* PIN # (TWISTED PAIR GROUND RETURN)
1	BIT32+	OUTPUT DATA BIT 32	21
2	BIT33+	" " " 33	22
3	BIT34+	" " " 34	23
4	BIT35+	" " " 35	24
5	BIT36+	" " " 36	25
6	BIT37+	" " " 37	26
7	BIT38+	" " " 38	27
8	BIT39+	" " " 39	28
9	BIT40+	" " " 40	29
10	BIT41+	" " " 41	30
11	BIT42+	" " " 42	31
12	BIT43+	" " " 43	32
13	BIT44+	" " " 44	33
14	BIT45+	" " " 45	34
15	BIT46+	" " " 46	35
16	BIT47+	" " " 47	36
40	BIT48+	" " " 48	60
41	BIT49+	" " " 49	61
42	BIT50+	" " " 50	62
43	BIT51+	" " " 51	63
44	BIT52+	" " " 52	64
45	BIT53+	" " " 53	65
46	BIT54+	" " " 54	66
47	BIT55+	" " " 55	67
48	BIT56+	" " " 56	68
49	BIT57+	" " " 57	69
50	BIT58+	" " " 58	70
51	BIT59+	" " " 59	71
52	BIT60+	" " " 60	72
53	BIT61+	" " " 61	73
54	BIT62+	" " " 62	74
55	BIT63+	" " " 63	75

* 78 Pin Male Connector (mounted on back panel)
AMP series HD-22 part #204509-1

78 Pin Female Connector (attaches to UUT cable)
AMP series HD-22 part #204508-1

DATE:________________

SHEET OF __________

APPENDIX E. RS-432 PROGRAMMING SHEET (CODING)

PROGRAM NAME:																							
ADDRESS OCTAL						INSTRUCTION																	
						MNEMONIC	OP CODE				DATA FIELD												COMMENTS
							15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
				0																			
				1																			
				2																			
				3																			
				4																			
				5																			
				6																			
				7																			
				0																			
				1																			
				2																			
				3																			
				4																			
				5																			
				6																			
				7																			
				0																			
				1																			
				2																			
				3																			
				4																			
				5																			
				6																			
				7																			
				0																			
				1																			
				2																			
				3																			
				4																			
				5																			
				6																			
				7																			
				0																			
				1																			
				2																			
				3																			
				4																			
				5																			
				6																			
				7																			
				0																			
				1																			
				2																			
				3																			
				4																			
				5																			
				6																			
				7																			

interface TECHNOLOGY

Date:____________________

Sheet_______ of ________

APPENDIX F. RS-432 WORD MEMORY PROGRAMMING SHEET

PROGRAM NAME:

ADDRESS (OCTAL)						DATA																COMMENTS
						15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		
				0																		
				1																		
				2																		
				3																		
				4																		
				5																		
				6																		
				7																		

interface TECHNOLOGY

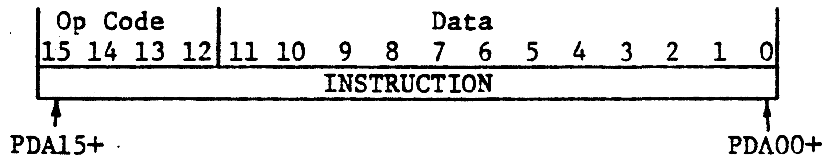
APPENDIX G. RS-432 PROM CARD

PROGRAMMING PROCEDURE

The RS-432 PROM card may be used in the program memory and/or the word memory to replace a RAM memory card. The PROM card is capable of being built in eight configurations dependent on the word size requirements. The 32, 64, and 256 word configurations are capable of being used in the program memory while the word memory may use the 32, 64, 256, 512 and 1024 configurations. The user should reference the configuration notes on the PROM card assembly/LBD drawing to determine the specific type and location of each PROM IC for the necessary memory configuration.

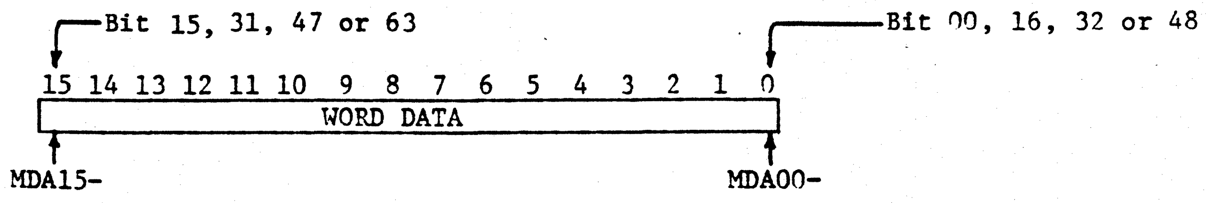
Programming of Program Memory PROMs

Program memory PROM programming is straight-forward. A logic one as observed on the front panel is programmed as a one (i.e., greater than 2.4 volts) in the program memory PROM. Conversely, a logic zero is programmed as a zero (i.e., less than 0.4 volts) in the program memory PROM. The correspondence between the program memory instruction format and the PROM card signal mnemonics is illustrated below:



Programming of Word Memory PROMs

Word memory PROM programming is opposite to that of the program memory. A logic one as observed on the front panel is programmed as a zero (i.e., less than 0.4 volts) in the word memory PROM. A logic zero as observed on the front panel is programmed as a one (i.e., greater than 2.4 volts) in the word memory PROM. The correspondence between the word memory data and the PROM card signal mnemonics is illustrated below:



APPENDIX H. RS-432 INPUT COMPARE

INPUT DATA BITS 16-31 & 48-63

INPUT CONNECTOR* PIN #	SIGNAL MNEMONIC AND DESCRIPTION		INPUT CONNECTOR* PIN # (TWISTED PAIR GROUND RETURN)
1	IND16+	INPUT DATA BIT 16	21
2	IND17+	" " " 17	22
3	IND18+	" " " 18	23
4	IND19+	" " " 19	24
5	IND20+	" " " 20	25
6	IND21+	" " " 21	26
7	IND22+	" " " 22	27
8	IND23+	" " " 23	28
9	IND24+	" " " 24	29
10	IND25+	" " " 25	30
11	IND26+	" " " 26	31
12	IND27+	" " " 27	32
13	IND28+	" " " 28	33
14	IND29+	" " " 29	34
15	IND30+	" " " 30	35
16	IND31+	" " " 31	36
40	IND48+	" " " 48	60
41	IND49+	" " " 49	61
42	IND50+	" " " 50	62
43	IND51+	" " " 51	63
44	IND52+	" " " 52	64
45	IND53+	" " " 53	65
46	IND54+	" " " 54	66
47	IND55+	" " " 55	67
48	IND56+	" " " 56	68
49	IND57+	" " " 57	69
50	IND58+	" " " 58	70
51	IND59+	" " " 59	71
52	IND60+	" " " 60	72
53	IND61+	" " " 61	73
54	IND62+	" " " 62	74
55	IND63+	" " " 63	75

* 78 Pin Male Connector (mounted on back panel)
AMP series HD-22 Part #204509-1

78 Pin Female Connector (attaches to UUT cable)
AMP series HD-22 Part #204-508-1

APPENDIX I. RSM SUPPLEMENTAL USER'S GUIDE

The Model RSM-432 is an RS-432 which has been adapted to transmit a series of multiple loop tables without adding discontinuities between the tables. This adaptation has been accomplished through both hardware and software means.

The following paragraphs specify the modifications to the standard RS-432 instructions and data formats. The reader is assumed to be familiar with the RS-432 User's Guide.

STL

To initially start the word generator, the start loop instruction (STL) is processed normally; i.e., the instruction is immediately executed and causes the word generator loop counter to be loaded. Once the word generator is running and an STL instruction is executed by the microprocessor, the microprocessor's address counter is prevented from advancing until the word generator completes processing the last word of the last loop of the previous table. The word generator then loads its loop counter with the contents of the STL instruction.

Note that this process requires that the microprocessor execute an STL instruction prior to the word generator completing the previous table. If this condition is not satisfied, the word generator will load the current contents of the microprocessor instruction register into the loop counter. This process also places a restriction of a 250 nanosecond minimum period on the first word of a table.

Once the word generator has been started, it may only be stopped by either depressing the CLEAR switch on the front panel or by executing the last table's STL instruction with bit 10 low. Bit 10 of the STL instruction has been redefined as follows:

0 -	stop word generator when current table/loops complete
1 -	continue to next table when current table/loops complete

DFMA (FMW2)

The fetch memory word 2 instruction (FMW2) has been redefined as define first memory address (DFMA). The DFMA instruction causes the first memory address to be defined (i.e., saved in a register) but does not cause it to be selected into the word memory address network as the standard FMW2 instruction accomplishes. The DFMA is used to define the first address of the next table while the word generator is completing the last loop of the current table. Since the word generator automatically accesses the stored first address upon completion of table, the DFMA permits defining the first address of the next table before completion of the current table thereby permitting the next table to be initiated without timing discontinuities.

DLA

The define last address (DLA) instruction no longer will halt the word generator. Note that the DLA instruction must be executed immediately following the STL instruction once the word generator has been started. This is due to the fact that the word generator does not halt between tables, automatically accesses the first word of the next table (DFMA) and continues accessing data. This also requires that all tables be greater than one word in length.

INPUT FLAGS/PULSES

Input flag 3 (IFLG3) is connected to the word generator's last loop flag and is therefore not available for use by external systems. Input flag 0 is dedicated to the error flag of the comparator option if installed; if not, IFLGO is available for use for external control. Input pulse 0 is dedicated to the A=B flag of the ALU option if installed; if not, IPULO is available for use for external control.

TIMING SIMULATOR DATA FORMAT

The timing simulator option is connected to word memory 32-47. Front panel access of the contents of word memory 32-47 is achieved by raising sense switch three as described in Table 7 of the RS-432 User's Guide. The format of the timing simulator data stored in word memory 32-47 is as follows:

```
15|14 13 12|11 10 9|8 7 6|5 4 3|2 1 0
---------------------------------------
| M  M |             PERIOD           |
---------------------------------------
```

PERIOD: binary, LSB bit 0

MM: Multiplier

00 =	500 usec	(500 usec to 8.1915 sec)
01 =	50 usec	(50 usec to 819.15 msec)
10 =	500 nsec	(500 nsec to 8.1915 msec)
11 =	50 nsec	(100 nsec to 819.15 usec)

PROGRAMMING EXAMPLE

The following is a simplified program which shall be used to describe the programming procedures. The example program transmits table A ten times, table B five times, table C twice and then halts.

PMA	PM	COMMENTS
00	DOP 1,1	Define output word format as 1 output word per memory word and 1 output word for the last memory word.
01	FMW1 = A start	Define and fetch first word of table A
02	STL E,C,10	Start word generator using external clock (i.e., timing simulator), continue to next table when complete, and repeat for 10 loops.
03	DLA = A end	Define last address of table A
04	JIF3 = 0, 04	Wait at address 04 until word generator has started last loop of table A.
05	DFMA = B start	Define first address of table B.
06	STL E,C,5	Continue word generator operation for table B under timing simulator control, continuing to next table when complete, and repeat for 5 loops. Note the microprocessor will wait at address 06 until the word generator completes the last word of the last loop of table A.
07	DLA = Bend	Define last address of table B.
10	JIF3 = 0, 10	Wait until word generator has started last loop of table B.
11	DFMA = C start	Define first address of table C.
12	STL E, S, 2	Continue with table C, stopping word generator when complete, and repeating for 2 loops.
13	DLA = C end	Define last address of table C
14	JLC2 = 1, 14	Wait at address 14 until word generator is complete.
15	JUN 15	Halt

In addition to the constraints discussed in previous paragraphs, it should be noted that the overhead programming of the microprocessor in setting up for the next table imposes a minimum on the total period of the current table (including loops). Once the microprocessor starts the word generator processing of one table, it must execute a specific series of instructions in order to set up for the next table; i.e., STL, DLA, JIF3, DFMA, and STL. Therefore, the microprocessor requires at least 1 usec (5 x 200 nsec) before the word generator completes processing the current table. If the user inserts other instructions (e.g., LOC, JLC, GØF, etc.), the total period of the previous table must account for the additional microprocessor overhead.

APPENDIX J. 16 BIT PARALLEL INTERFACE USER'S GUIDE

INTRODUCTION

The RS-432 16 bit parallel interface enables an external device to update all program and word memory locations as well as start and stop the RS-432 program execution. The external device (computer, tape reader, card reader, etc.) gains control of the RS-432 by either requesting control (see REXTC- signal description in Table 1) or by the operator pressing the EXT CONTROL switch. Tables 1, 2 and the Timing Diagram provide a detailed description of the interface.

INTERFACE - MECHANICAL AND ELECTRICAL

All interface signals are provided on a 50 pin Cannon connector. The male connector (Cannon part #DD50P) is housed on the back panel of the RS-432. The female connector (Cannon part #DD50S) is connected (via a twisted pair cable) to the external device.

Interface signals are TTL compatible signals. All Input signals are 1 TTL unit load. The Output signal (READY-) can drive 10 TTL unit loads.

MNEMONIC DESCRIPTION

All signal mnemonics infer "positive logic". A "+" sign at the end of a signal mnemonic indicates the signal is active when in the "1" (_2.4 volts) state. A "-" sign indicates the signal is active when in the "0" (0 volts) state.

The exception to this rule applies to the twisted pair ground return lines. Their signals are always at 0 volts, however, the signal mnemonic is the opposite of the respective twisted pair signal.

TABLE 1

OUTPUT CONNECTOR PIN ASSIGNMENTS AND SIGNAL DESCRIPTION

PIN #	SIGNAL NAME AND DESCRIPTION		
1	DUS00+	INPUT DATA BIT 0 (Also defines Bit 16)	
2	DUS01+	" " " 1 " " " 17	
3	DUS02+	" " " 2 " " " 18	
4	DUS03+	" " " 3 " " " 19	
5	DUS04+	" " " 4 " " " 20	
6	DUS05+	" " " 5 " " " 21	
18	DUS06+	" " " 6 " " " 22	
19	DUS07+	" " " 7 " " " 23	
20	DUS08+	" " " 8 " " " 24	
21	DUS09+	" " " 9 " " " 25	
22	DUS10+	" " " 10 " " " 26	
34	DUS11+	" " " 11 " " " 27	
35	DUS12+	" " " 12 " " " 28	
36	DUS13+	" " " 13 " " " 29	
37	DUS14+	" " " 14 " " " 30	
38	DUS15+	" " " 15 " " " 31	
42	FCT01+	Function Code Bit 1 (The four function code bits define the operation which will occur after an execute pulse is generat See Table 2).	
26	FCT01-	Twisted pair ground return for FCT01+.	
43	FCT02+	Function Code Bit 2.	
27	FCT02-	Twisted pair ground return for FCT02+.	
44	FCT03+	Function Code Bit 3.	
28	FCT03-	Twisted pair ground return for FCT03+.	
45	FCT04+	Function Code Bit 4.	
29	FCT04-	Twisted pair ground return for FCT04+.	
7	ADD01+	Device Address Bit 1.	The 8 devide address bits enabl
8	ADD02+	" " " 2.	external control. These
9	ADD03+	" " " 3.	8 bits must equal the 8 device
23	ADD04+	" " " 4.	address switches (dual in line
24	ADD05+	" " " 5.	switch) on the front panel prin
25	ADD06+	" " " 6.	circuit card (See page 6). Thi
39	ADD07+	" " " 7.	equality must exist through ent
40	ADD08+	" " " 8.	external control process.
10	REXTC-	Request External Control Pulse (same as front panel EXT CONT Switch). Device address bits must equal device address swit for the REXTC- pulse to gain control. Note, the <u>only</u> way th front panel can regain control of the machine is activation the front panel STOP/CLEAR switch. REXTC- pulse may be retu high after 50 nanoseconds or may remain low until user requi front panel control.	
11	REXTC+	Twisted pair ground return for REXTC-.	

TABLE 1: CONTINUED

PIN #	SIGNAL NAME AND DESCRIPTION	
48	READY-	The READY- signal is the only signal returned to the user. READY- goes low (0 volts) when the device address bits equal the device address switches <u>and</u> when the request external control pulse (REXTC-) has been received (or when the front panel EXT CONTROL switch has been activated). Note in the timing diagram that after each Execute Pulse (XCUTE-) is received, READY- will go not ready (high) for approximately 600 nanoseconds while data is entered into the RS-432. Note also that READY- will go not ready if an invalid function code is received (see note on Page 4).
32	READY+	Twisted pair ground return for READY-.
49	XCUTE-	The Execute Pulse performs the operation defined by the 4 function code bits. The operation begins execution after receiving a negative transition (high level to low level ‾↓_⌐‾) on the XCUTE- line. The RS-432 then requires approximately 600 nanoseconds after the negative transition to completely execute the function. XCUTE- may be returned high after 50 nanoseconds but obviously must be returned high before another XCUTE- is generated.
33	XCUTE+	Twisted pair ground return for XCUTE-.
46	RESET-	The RESET- pulse clears the RS-432 by stopping the execution of the program and stopping the word generator. RESET- performs all the functions that the front panel STOP/CLEAR switch performs except it does return control to the front panel. The RESET- function occurs when the RESET- pulse goes low (0 volts). RESET- pulse may be returned high after 50 nanoseconds but must be returned high before a START- or XCUTE- pulse is generated.
30	RESET+	Twisted pair ground return for RESET-.
12	START-	The START- pulse starts execution of the program. The START function occurs when the START- pulse goes low (0 volts). The START- pulse may be returned high after 50 nanoseconds but must be returned high before a RESET- pulse is generated. The START- pulse performs the same function as the START PROGRAM switch on the front panel.
13	START+	Twisted pair ground return for START-.
41	SELCT+	When transferring data into the RS-432 word memory (defined by function code 0001 and 0011) the SELCT+ signal selects the low order data bits (bits 0-31) when low (0 volts) and the high order data bits (bits 32-63) when high ($\geq$ 2.4 volts). This signal <u>only</u> applies when more than 32 bits are incorporated in the word memory (i.e., for 16 and 32 bit machines, SELCT+ may be left open). SELCT+ has the same timing relationship as data bits DUSXX+.
50	LOGIC GROUND - 0 volts. Also ground return for SELCT+ if applicable.	

TABLE 2

FUNCTION CODE ASSIGNMENTS

FUNCTION CODE	DESCRIPTION
0000	Load data (DUS00+ → DUS15+) into program memory.
0001	Load data (DUS00+ → DUS15+) into word memory bits 0-15.
0010	Invalid code* (See note below).
0011	Load data (DUS00+ → DUS15+) into word memory bits 16-31.
0100	Load data (DUS00+ → DUS07+) into program memory address counter.
0101	Load data (DUS00+ → DUS11+) into word memory address counter.
0110	Invalid code*.
0111	Invalid code*.
1000	Same as function code 1100.
1001	Same as function code 1101.
1010	Invalid code*.
1011	SAme as function code 1101.
1100	Increment program memory address counter.
1101	Increment word memory address counter.
1110	Invalid code*.
1111	Invalid code*.

FCT01+ (0= selects program memory; 1 = selects word memory)

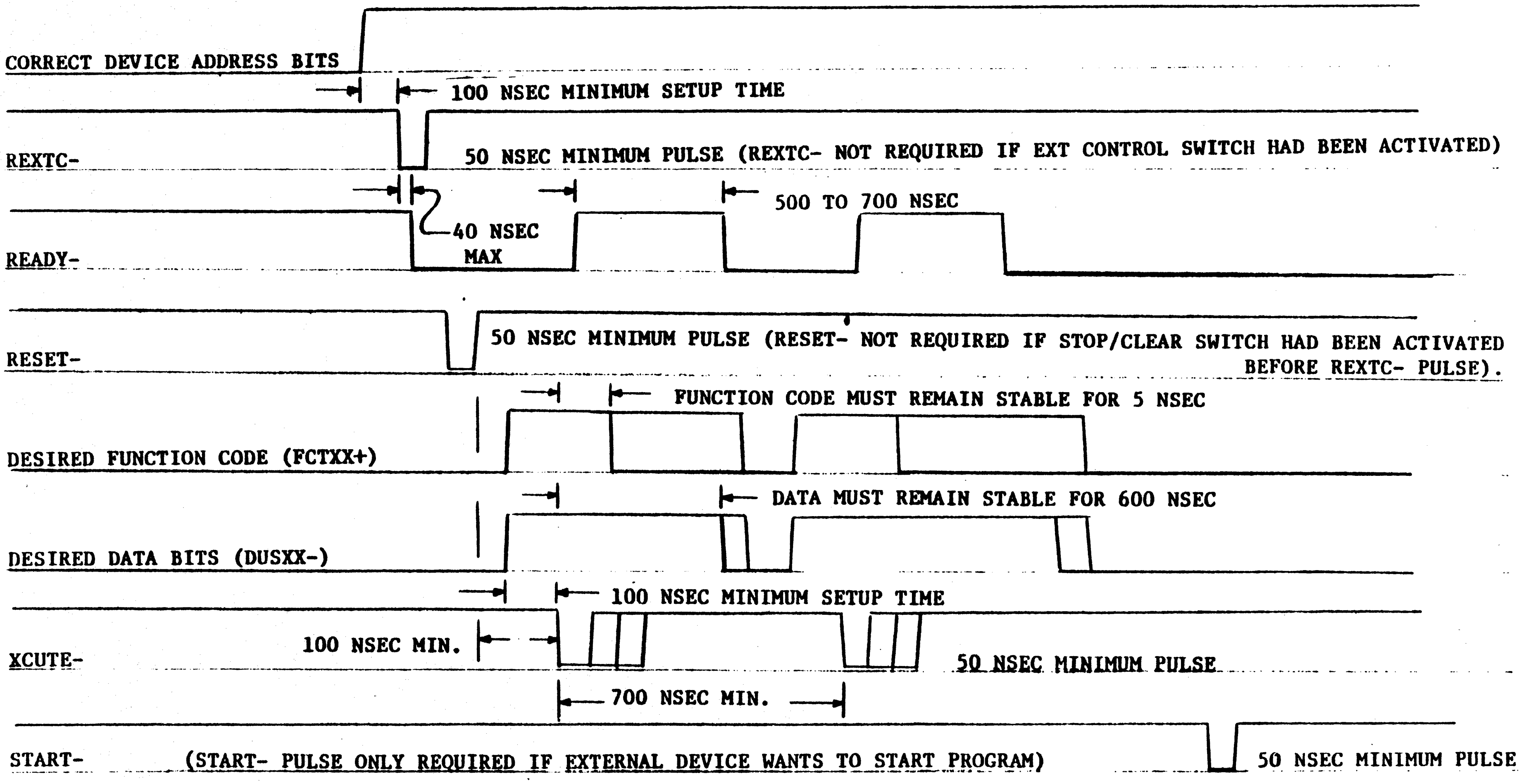
FCT02+ (0 = selects word memory data bits 0-15; 1 = selects word memory data bits 16-31.)

FCT03+ (0 = selects operation on memory; 1 = selects operation on addre counter).

FCT04+ (0 = selects load function; 1 = selects increment function)

*If an execute pulse is received with an invalid function code, no data will be enter into the RS-432, however, the READY signal will go not ready. It will remain in the not ready condition until a valid function code is received with an execute pulse or when a RESET signal is received.

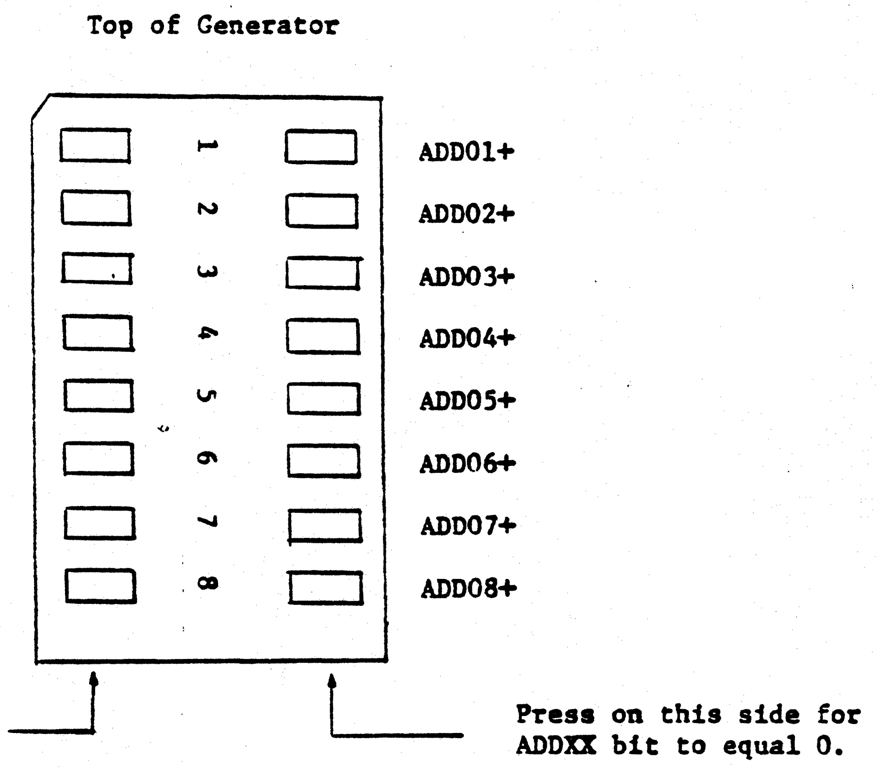
TIMING DIAGRAM AND TYPICAL SEQUENCE OF OPERATION



CORRECT DEVICE ADDRESS BITS
100 NSEC MINIMUM SETUP TIME
REXTC-
50 NSEC MINIMUM PULSE (REXTC- NOT REQUIRED IF EXT CONTROL SWITCH HAD BEEN ACTIVATED)
READY-
40 NSEC MAX
500 TO 700 NSEC
RESET-
50 NSEC MINIMUM PULSE (RESET- NOT REQUIRED IF STOP/CLEAR SWITCH HAD BEEN ACTIVATED BEFORE REXTC- PULSE).
DESIRED FUNCTION CODE (FCTXX+)
FUNCTION CODE MUST REMAIN STABLE FOR 5 NSEC
DESIRED DATA BITS (DUSXX-)
DATA MUST REMAIN STABLE FOR 600 NSEC
XCUTE-
100 NSEC MINIMUM SETUP TIME
100 NSEC MIN.
50 NSEC MINIMUM PULSE
700 NSEC MIN.
START-
(START- PULSE ONLY REQUIRED IF EXTERNAL DEVICE WANTS TO START PROGRAM)
50 NSEC MINIMUM PULSE

SPECIAL NOTES

DEVICE ADDRESS SWITCHES: The device address switches are located on the lower right hand side (when viewed from inside of RS-432) of the front panel printed circuit card. The following diagram depicts which switch corresponds to the respective device address bit.



Top of Generator
1
2
3
4
5
6
7
8
ADD01+
ADD02+
ADD03+
ADD04+
ADD05+
ADD06+
ADD07+
ADD08+
Press on this side for ADDXX bit to equal 1.
Press on this side for ADDXX bit to equal 0.

INTERFACE CABLE LENGTH: For proper operation it is recommended that interface cable length between the RS-432 and the control device be limited to 10 feet. Note that control signals and the function bus are twisted pair lines while the address and data buses are single conductor.

1.0 INTRODUCTION

The RS-432 IEEE interface has been designed to operate in accordance with IEEE Std 488-1975. The following functions have been implemented: Source Handshake (SH), Acceptor Handshake (AH), Listener (L), Service Request (SR), Remote Local (RL), Device Clear (DC), and Device Trigger (DT). These functions permit the user to load the RS-432's memories, to start and stop the program under remote control, and to send status information. The talk function (T) control logic is also included in the interface design; however, it is not available for use in the basic RS-432 IEEE interface since the data transmitted by the talk function is dependent upon the option configuration of the particular RS-432.

Figure 1 depicts the IEEE interface as implemented in the RS-432.

2.0 DATA FORMATS

2.1 Memory Load Data

The following illustrates the format of the data bytes (DAB) to be received by the RS-432 IEEE interface listen function in order to load the program and word memories. It should be noted that the format specifies both each byte's content and the sequence of the bytes when transmitted to the RS-432.

Byte No.	Name	Format
1	Memory Select	

```
8 7 6 5 4 3 2 1 <---- IEEE Bus Bit Number
|X X X X X| M |W|
```

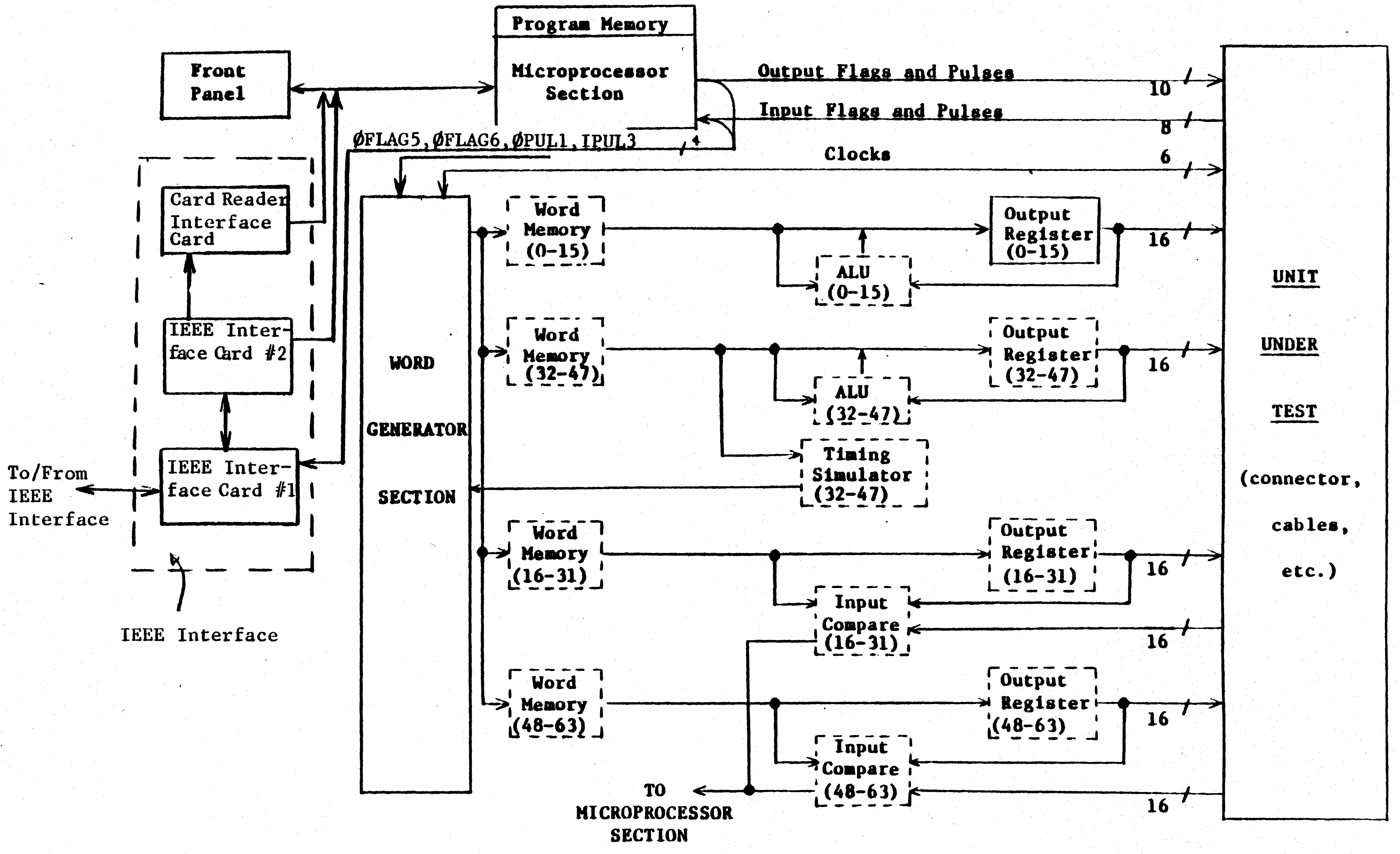
W
0 = Load program memory
1 = Load word memory

M (interpreted only if W = 1)
0 0 = Load Word Memory 00-15
0 1 = Load Word Memory 16-31
1 0 = Load Word Memory 32-47
1 1 = Load Word Memory 48-63

Byte No.	Name	Format
2	Memory Address MSBs	

```
8 7 6 5 4   3   2  1
|X X X X|A11 A10 A9 A8|
         M          L
```

$A_{11}\ A_{10}\ A_9\ A_8$: Four MSBs of memory address where the first data word is to be loaded. Byte 2 is paired with Byte No. 3 to form a 12 bit address.



8/15/76

FIGURE 1. RS-432 BLOCK DIAGRAM WITH IEEE INTERFACE

interface

Byte No.	Name	Format
3	Memory Address LSBs	8 7 6 5 4 3 2 1 A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0 M L A_7 - A_0: Eight LSBs of memory address where the first data word is to be loaded.
4	Data Word LSBs	8 7 6 5 4 3 2 1 D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0 M L D_7 - D_0: Eight LSBs of data word. Byte No. 4 is paired with Byte No. 5 to form a 16 bit data word.
5	Data Word MSBs	8 7 6 5 4 3 2 1 D_{15} D_{14} D_{13} D_{12} D_{11} D_{10} D_9 D_8 M L D_{15} - D_8: Eight MSBs of data word.

Repeat transmission of byte numbers 4 and 5 until the desired number of memory words has been loaded.

To indicate completion of a particular memory load, the controller must either send the END message simultaneously with the last byte or clear the LISTEN condition (i.e., UNL or MTA).

2.2 Status Byte

The status byte (STB) illustrated below is transmitted by the RS-432 in response to a bus serial poll request.

8	7	6	5	4	3	2	1
X	R	X	X	OF7	L	OF6	OF5

R
0 = service not requested by RS-432
1 = service requested by RS-432; i.e., RS-432 set SRQ on IEEE bus

L
0 = RS-432 is under remote control; i.e., on-line
1 = RS-432 is under local control; i.e., off-line

OF7, OF6, OF5 = Output flags 7, 6, and 5 (user may set the condition of these signals via RS-432 GØF instruction)

XXXX = spare bits; will be interpreted as 1 level by controller

3.0 BUS PROTOCOL

3.1 RS-432 Program Interaction with the Bus

The RS-432 program is capable of interacting with the bus in several ways. As described in paragraph 2.2, the program may set RS-432 output flags 5, 6 and/ 7 to pass status information to the system controller. This is accomplished by using the RS-432 GØF instruction.

When the program desires to request bus service, it uses output pulse 1 (ØPUL1). ØPUL1 must be initialized by the RS-432 program to the 1 level; i.e., the GØF instruction format is 0 110 11X XXX XXX XXX. To generate service request (SRQ), the program is required to pulse output pulse 1 to the low level; i.e., a GØF instruction of the following format: 0 110 10X XXX XXX XXX.

The IEEE bus is capable of remotely starting the RS-432 program by transmitting the GET message. If the program is already started and a GET message is transmitted, the latch associated with input pulse 3 (IPUL3) is set. This permits the program to wait for bus commands by looping on input pulse 3; i.e., if input pulse 3 did not occur, jump back to the address of the JIP 3 instructio When IPUL3 (i.e., GET) is received, the program will advance to the next instruction. Typically, the JIP instruction, which checks input pulse 3 is used following the request for service (ØPUL1 = SRQ).

3.2 Bus Protocol Example

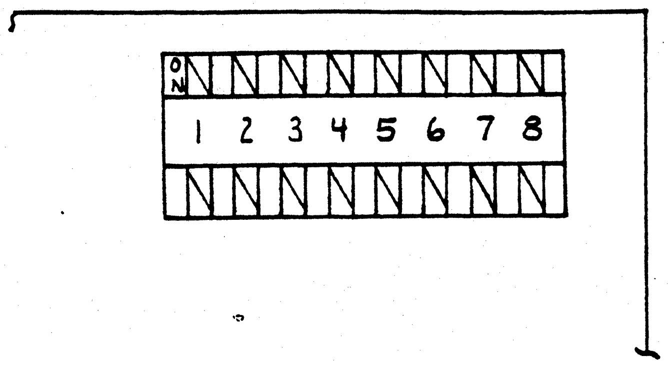
The following chart illustrates a typical IEEE interface message sequence and the associated RS-432 action or response. An interface message which is underlined (e.g., <u>STB</u>) indicates it is transmitted by the RS-432. Conversely, a message which is not underlined indicates one which is received by the RS-432.

Step No.	Interface Message	Action/Response
1	IFC	Clears interface networks.
2	MLA∧ATN∧REN	Sets up RS-432 as listener under remote control.
3	DCL∧ATN	Stops RS-432 program.
4	DAB	Loads selected RS-432 memory. Reference paragraph 2.1.
	⋮	
5	DAB	Repeat steps 4 - 6 until all memories are loaded as required. Steps 6 and 7 mav be omitted if the controller is capable of sending the END message simultaneously with the 1 byte.
6	UNL∧ATN	
7	MLA∧ATN	
8	DAB	Data bytes 1, 2, & 3 illustrated in paragraph 2.1.
9	DAB	Presets RS-432 program memory address counter to starting location.
10	DAB∧END	

Step No.	Interface Message	Action/Response
11	GET∧ATN	Start RS-432 program. Assume RS-432 is a listener. Note that the GET message will start the program if the program is stopped. If the program is already started, the GET message will strobe input pulse 3 (IPUL3). A typical application of this would be for the program to loop waiting for IPUL3.
12	UNL∧ATN	Eliminates RS-432 as interface listener.
13	SRQ	RS-432 program transmits service request message by generating output pulse 1 (ØPUL1). Service request may be due to normal program completion or due to detection of an error. The program would normally loop waiting for IPUL3 generated by the next GET message.
14	SPE∧ATN	Controller sets up all devices for serial poll.
15	MTA∧ATN	Sets up RS-432 to talk in order to transmit status byte.
16	STB	RS-432 transmits status byte. Reference paragraph 2.2. Note that END message is not transmitted.
17	OTA∧ATN	Controller continues serial poll, disabling RS-432 as talker.
18	SPD∧ATN	Controller completes serial poll.
		If the RS-432 program is done, the controller may repeat the test by branching to Step11 or may change the memory test data by branching to Step 2.

4.0 HARDWARE CONFIGURATION

In order to use the RS-432 IEEE interface, the user must connect his bus cable to the connector located on the rear of the unit and then manually set the RS-432 talk/listen address dipswitch to his desired address. To set the dipswitch, the RS-432's top cover must be removed. The dipswitch may be easily located on one of the vertically mounted logic cards and is illustrated below:



Dipswitches 4 thru 8 are used to select the talk/listen address. Switch 4 corresponds to bus address bit 5 while switch 8 is associated with address bit 1. The unit is normally shipped with an address of 00 100 selected. Depress the lower section of each switch to make the address bit a logic "1". Depress the upper section of each switch to make the address bit a logic "0".

APPENDIX L. RS-232-C/TTY INTERFACE

Introduction

The RS-232-C interface provides the capability to remotely load the program and word memories of the RS-432 and to remotely reset and start the RS-432.

The receive baud rate of the RS-232-C interface may be set to one of the following rates:

(a)	50	(h)	600
(b)	75	(i)	1200
(c)	110	(j)	1800
(d)	134.5	(k)	2400
(e)	150	(l)	4800
(f)	200	(m)	9600
(g)	300		

The serial format consists of one start bit, 7 data bits, one parity bit (ignored) and one stop bit.

The interface operates in the receive-only mode thereby requiring the transmission mode to be half-duplex.

The RS-432 may be connected directly to a terminal (i e , Data Terminal Equipment, DTE) in which case the RS-432 functions as the Data Communication Equipment (DCE). Alternately, the RS-432 may be connected as a terminal to a DCE.

The valid character repertoire consists of the following:

(a)	#	:	request remote control
(b)	@	:	data entry complete
(c)	Ø 1 2 3 4 5 6 7	:	numeric data (octal)
(d)	,	:	load preceding data
(e)	R	:	reset RS-432
(f)	S	:	start RS-432

All other characters are ignored. This feature permits inserting characters such as carriage return (CR), line feed (LF), space, etc., for ease of page printout formatting.

Preparation for Use

To connect the RS-432 to a terminal, simply connect the terminals integral cable to the designated RS-432 rear panel connector. Set the terminal's mode to half-duplex.

To connect the RS-432 as a terminal, connect P1 of the supplied terminal cable to the RS-432 rear panel connector and connect P2 to the DCE's connector. Set the DCE's mode to half-duplex.

To set the system baud rate, disconnect the AC power source to the RS-432 and remove the top cover .

CAUTION

IT IS RECOMMENDED THAT THE AC POWER SOURCE ALWAYS BE DISCONNECTED PRIOR TO CONFIGURING THE UNIT OR PERFORMING MAINTENANCE.

IF MAINTENANCE PROCEDURES REQUIRE CONNECTION TO THE PRIMARY AC SOURCE, EXTREME CAUTION SHOULD BE EXERCISED. ALL POTENTIALLY ACCESSIBLE PRIMARY VOLTAGE TERMINALS ARE REASONABLY PROTECTED TO PRECLUDE ACCIDENTAL CONTACT.

Locate the RS-232-C interface card by referencing the Model RS-432E Top Assembly drawing. The card is located in slot J6 of the rear logic assembly A4. The four rocker switches for setting the baud rate are located on the top edge of the card near the rear. Each switch has a number imprinted on the top half of its rocker . Switch number one is the rear-most switch. To turn a switch on, depress the top half of the switch; conversely, to turn it off, depress the bottom half. Set the baud rate according to the following:

Baud Rate	Switch No.	1	2	3	4
50		ON	ON	OFF	ON
75		ON	ON	OFF	OFF
110		OFF	OFF	OFF	OFF
134.5		ON	OFF	ON	ON
150		OFF	OFF	OFF	ON
200		ON	OFF	ON	OFF
300		OFF	OFF	ON	OFF
600		ON	OFF	OFF	ON
1200		OFF	ON	OFF	OFF
1800		OFF	ON	OFF	ON
2400		ON	OFF	OFF	OFF
2400		OFF	OFF	ON	ON
4800		OFF	ON	ON	OFF
9600		OFF	ON	ON	ON

<u>Operation</u>

The general protocol to load a memory is as follows:

```
#           request remote control
nn,         select memory
nnnn,       specify starting address (octal)
nnnnnn,
   .
   .        memory data (octal)
   .
   ..
nnnnnn,
```

@	data entry complete
#	request remote control
nn,	select next memory
.	etc.
.	
.	
.	

The request remote control character (#) must be followed by the two memory select characters. The memory select characters are as follows:

ØØ	program memory
Ø1	word memory ØØ-15
Ø3	word memory 16-31
Ø5	word memory 32-17
Ø7	word memory 48-63

The memory select characters must be followed by the four starting address characters which specify the 12 bit address.

Following the starting address characters and execute comma, the data character fields may be transmitted. Leading zeroes may be omitted; however, one execute comma may not immediately follow another. If a word is all zeroes, one zero and a comma must be transmitted. Data words are entered MSB first. The execute comma causes the preceding data to be loaded and the memory address to be incremented in expectation of the next data.

Following transmission of a memory's data, the data entry complete character (@) must be transmitted. To load the next memory, the sequence must be repeated.

The request remote control character (#) places the RS-432 in EXTERNAL CONTROL. To return to local front panel control, the user must manually depress the front panel STOP/CLEAR switch. Unless the RS-432 is in EXTERNAL CONTROL, all received characters will be ignored.

The reset character (R) may come at any time while in EXTERNAL CONTROL and stops the RS-432. Likewise, the start character (S) starts the unit while in EXTERNAL CONTROL. Note that the program memory address must be preset to the desired starting address prior to transmitting the run character. Also, note that the RS-432 cannot be loaded if it is running, ie., first reset the unit prior to loading.

<u>Examples</u>

The following pages provide a sample program illustrating the preceding discussion.

RS-232-C Sample Program Entry

```
# . . . . . . . . . . . . . . . . .    places RS-432 in remote control
R . . . . . . . . . . . . . . . . .    resets RS-432
00, . . . . . . . . . . . . . . . .    selects program memory
0000, . . . . . . . . . . . . . . .    selects pgm memory address 0000
110 101,  . . . . . . . . . . . . .    loads DOP 1,1 into PMA 0000
130 000,  . . . . . . . . . . . . .    loads FMW1 = 000 into PMA 0001
120 017,  . . . . . . . . . . . . .    loads DLA = 017 into PMA 0002
037 006,  . . . . . . . . . . . . .    loads JLC LMA into PMA 0003
150 000,  . . . . . . . . . . . . .    loads CSR into PMA 0004
040 003,  . . . . . . . . . . . . .    loads JUN to PMA 0003 into PMA 0005
040 001,  . . . . . . . . . . . . .    loads JUN to PMA 0001 into PMA 0006
@ . . . . . . . . . . . . . . . . .    end of transfer
#
00, . . . . . . . . . . . . . . . . . }  presets PMA = 0000 to start
0000,
@
#
01, . . . . . . . . . . . . . . . .    selects WM00 - 15
0000, . . . . . . . . . . . . . . .    selects WMA = 0000
1,  . . . . . . . . . . . . . . . .    loads 000 0001 into WMA 0000
2,  . . . . . . . . . . . . . . . .    loads 000 002 into WMA 0001
4,
10,
20,
40,
100,
200,
400,
1 000,
2 000,
4 000,
10 000,
20 000,
40 000,
100 000,
@
S . . . . . . . . . . . . . . . . .    starts RS-432
```

Resultant Program Memory Contents

00

00	DOP 1,1	110 101
01	FMW1 00	130 000
02	DLA 17	120 017
03	JLC LMA	037 006
04	CSR	150 000
05	JUN * -2	040 003
06	JUN * -5	040 001

Resultant Word Memory Contents

00	000 001
01	000 002
02	000 004
03	000 010
04	000 020
05	000 040
06	000 100
07	000 200
10	000 400
11	001 000
12	002 000
13	004 000
14	010 000
15	020 000
16	040 000
17	100 000

APPENDIX M. 8 BIT PARALLEL INTERFACE USER'S GUIDE

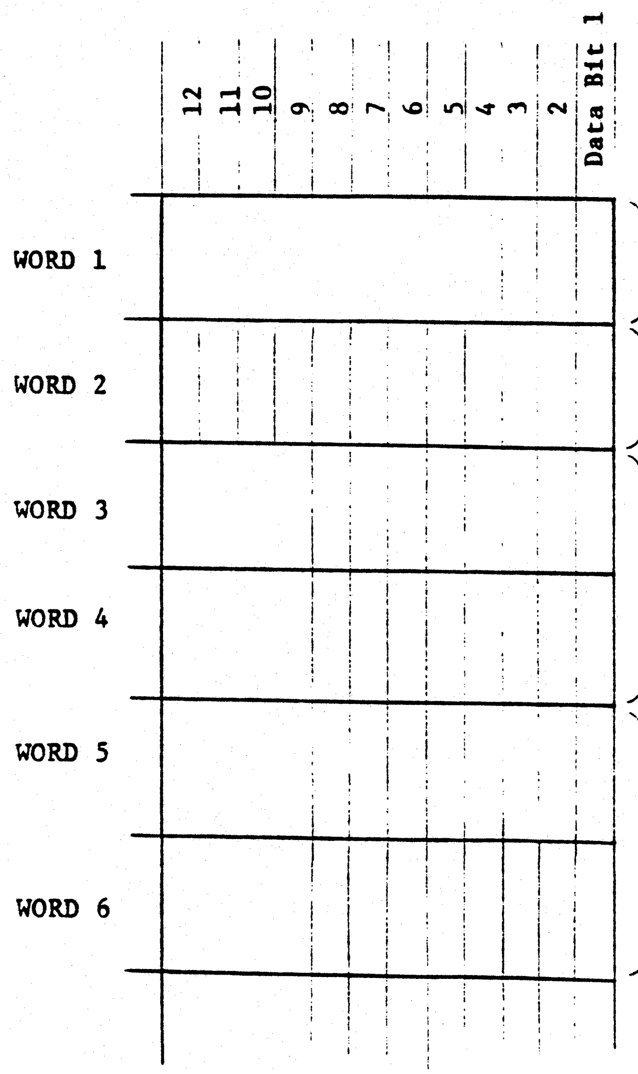
GENERAL DESCRIPTION

The 8 bit paralle interface enables the user to load 8 bit parallel data bytes into the RS-432 or RSM-432 program and word memories. The interface accepts data blocks from the loading device, assembles these blocks into 16 bit data words and generates all load, increment and addressing signals required for proper loading into the RS-432.

It must be noted that the interface actually incorporates 12 data lines of which 8 are used for the data bits. The additional 4 lines are required for addressing when the RS-432 has more than 256 words of word memory. Table 1 depicts the data block format; Figure 1 describes the timing; and Table 2 shows the connector pin assignment.

All signals are TTL compatible each signal representing 2 TTL loads. Refer to the Card Reader Interface Card (Dwg #10011002) for an assembly and schematic drawing of this interface.

From an operational point of view the user shall go through one timing sequence as shown in Figure 1, Timing Diagram, for the Program Memory a second timing sequence for the Word Memory bits 00-15 and other timing sequences as required to load the other Word Memories. The user can then perform a truncated timing sequence where the CPØMD signal is brought low, two data bytes (P/W and FMA) are transferred and CPØMD is brought high to terminate transmission. This truncated sequence presets the Program Memory Address Counter to a known starting address, (the FMA value transmitted with the truncated timing signal).



Define which memory is to be loaded

Bit 1=0= Program Memory
Bit 1=1= Word Memory

Bit 1=1 then Bit 2	Bit 3	
0	0	Data 0-15
0	1	Data 16-3
1	0	Data 32-4
1	1	Data 48-6

Define first memory address (FMA)

Bit 1=LSB
Bit 12=MSB

Data for FMA

8LSB's	Bit 1 = Bit 0
	Bit 8 = Bit 7
8MSB's	Bit 1 = Bit 8
	Bit 8 = Bit 15

Data for FMA+1

Continue Data for any number of Data Words.

TABLE 1 - Data Block

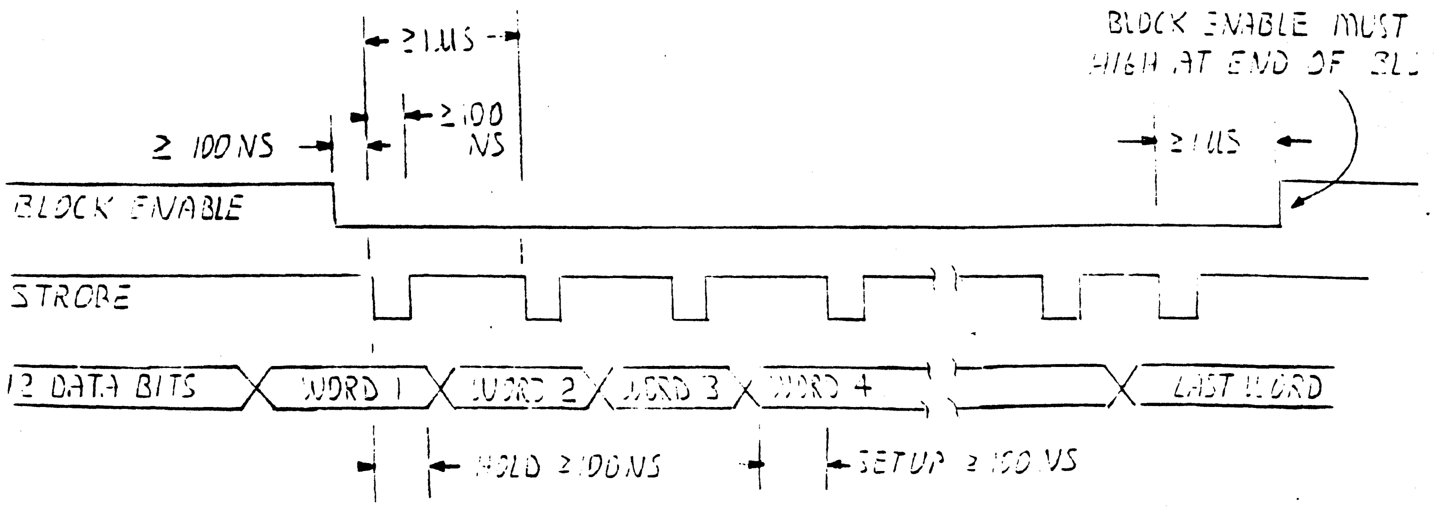


FIG. 1 TIMING DIAGRAM

Signal Name	Signal Mnemonic	Connector Pin No. 1
Data Bit 1 (LSB)	DAT00+	1
Data Bit 2	DAT01+	2
Data Bit 3	DAT02+	3
Data Bit 4	DAT03+	4
Data Bit 5	DAT04+	5
Data Bit 6	DAT05+	6
Data Bit 7	DAT06+	18
Data Bit 8	DAT07+	19
Data Bit 9	DAT08+	20
Data Bit 10	DAT09+	21
Data Bit 11	DAT10+	22
Data Bit 12 (MSB)	DAT11+	34
Strobe	STØMI-	49
Strobe tw. pr.		
Block Enable	CPØMD-	10
Block Enable tw. pr.		
Signal Ground		15
+5VDC		14 2
		10

Notes:

1 Connector designated 16 BIT PARALLEL INTERFACE on rear panel of RS-432
2 User should not have any connections to this pin

TABLE 2
8 BIT PARALLEL INTERFACE CONNECTOR PIN ASSIGNMENTS

APPENDIX N. CARD READERS USER'S GUIDE

The RS-432 Card Reader Option enables the user to mark (with soft pencil) or punch cards and load the data from these cards into the program and word generator memories.

The marked cards are hand fed into the card reader (card reader is a Hewlett Packard part $9870A type card reader). The data format for each card is depicted in Figure 3, Card Data Formats.

For all data, a "one" ($\geq$2.4v) is inserted by marking the bit positions with a soft lead pencil. A "zero" (0v) is achieved by making no mark in the respective bit positions.

The first row of data bits (Row 1) defines the memory in the RS-432 which will receive the cards data. If Bit 0 = 0, the data is to be loaded into the program memory. Bit positions 1 and 2 become "don't care" conditions. If Bit 0 = 1, the data will be loaded into the word memory. Bit positions 1 and 2 then specify which word memory will receive the card data as defined in Figure 1.

Bit 2	Bit 1	
0	0	Word memory bits 0 - 15
0	1	Word memory bits 16 - 31
1	0	Word memory bits 32 - 47
1	1	Word memory bits 48 - 63

Figure 1

The second row of data bits (Row 2) defines the first memory address (program or word memory address depending on Bit 0 in Row 1) which will receive the data from the data block.

The data block of 16 memory words is defined by the following 32 rows od data. Rows 3 and 4 define the data for the first word (FW), rows 5 and 6 define the data for the second word (FW+1), and finally row 33 and 34 define the data for the sixteenth word (FW+15). Each two rows define one sixteen bit data word shown in Figure 2.

	7	6	5	4	3	2	1	0	
	Bit							Bit	
3	7	6	5	4	3	2	1	0	Data for FW
4	15	14	13	12	11	10	9	8	
	Bit							Bit	
5	7	6	5	4	3	2	1	0	Data for FW+1
6	15	14	13	12	11	10	9	8	

FIGURE 2.

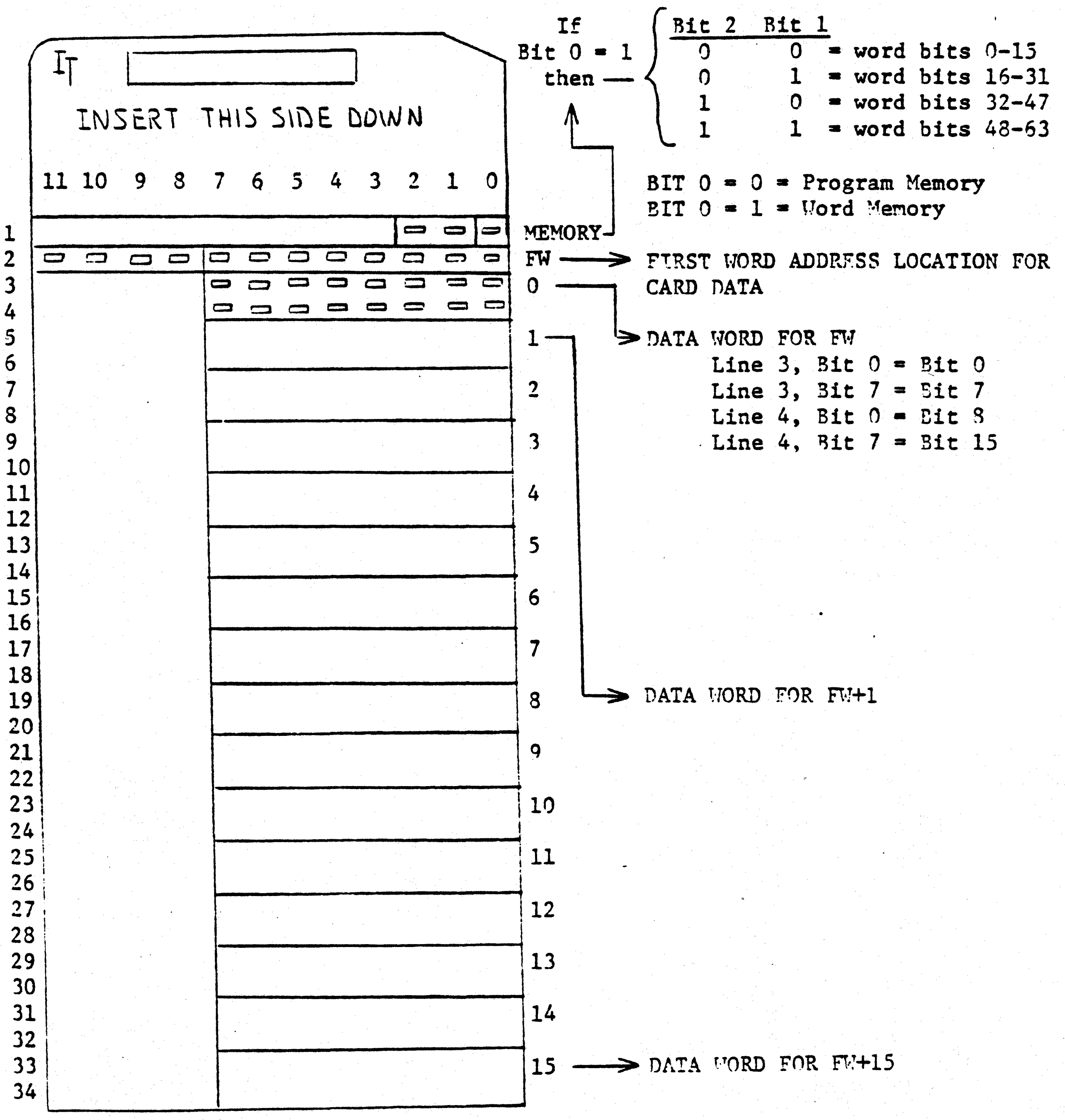


FIGURE 3. CARD DATA FORMAT

With this method of memory definition a standard RS-432 program memory of 64 x 16 bits can be recorded on four cards. A 256 x 16 bit program or word memory can be recorded in 16 cards. The individual cards, because of the first memory address and memory destination definition can be loaded in any sequence, not requiring an ordered deck of cards.

NOTES FOR CORRECT LOADING SEQUENCE

Enter each card into the card reader with the marked side down. Release card as soon as motor begins pulling card from hand. DO NOT PUSH card into reader after motor grasps card.

If a card is pushed into the reader, erroneous data will result. However, it is often difficult to determine if the card was pushed or not. The following observation of the RS-432 front panel address lights can aid in detecting a bad loading sequence.

Each card will always load sixteen words regardless if all data was marked on the card or not. The memory address for a correct loading sequence will always end up at the cards First Memory Address plus 16. In other words, after the card has been loaded the 4 least significant bits (LSB's) of the address counter (address lights on RS-432 front panel) will equal the 4 LSB's of the FIRST MEMORY ADDRESS on the card. If they do not, a bad loading sequence occurred and the card should be read over.

APPENDIX O. DUAL 8 BIT PARALLEL/16 BIT PARALLEL INTERFACE

The Dual Card Reader/16 Bit Parallel Interface permits the user to enter data into the RS-432 via either the protocol of the 8 Bit Parallel Interface or the 16 Bit Parallel Interface. Reference Appendix J for discussion on on the subject interface.

The I/O signals of both interfaces are connected to the back panel connector marked 16 BIT PARALLEL INTERFACE. The following paragraphs describe the means of converting from one interface to the other. Reference the attached interface schematic/assembly drawing.

8 Bit Parallel to 16 Bit Parallel

1. Move front panel cable (A1J5) from interface card location 0-2 to 0-1.

2. Move front panel cable (A1J4) from interface card location 4-6 to 4-5.

3. Install two N8242 ICs into front panel (A1) locations C1 and C2.

4. Install dipswitch IC into front panel location W1. Set dipswitches to the desired address.

16 Bit Parallel to 8 Bit Parallel

1. Remove dipswitch and N8242 ICs in front panel locations C1, C2 and W1.

2. Move front panel cable (A1J5) from interface card location 0-1 to 0-2.

3. Move front panel cable (A1J4) from interface card location 4-5 to 4-6.

Note this is not a standard catalog item; however, it is available on some RS-432 models. Consult the factory if there are any questions.