





SY24-3581-4

OUTLINE				314	5 Proce	ssing Un	it Theo	ory-Maint	enance					• • • • •
	VOL 30			VOL 31		VOL	32		VOL 33			VOL	. 34	
	INDEX PLAN INTR CPU			MIC CFA CPK		PW STC	R DR		CHNL IFA FEAT			CN RE DI RE	ISL IC AG IF	
X		And and a second second			No.	1		X	n da se	and the second second		Ć		
PLAN	INTR	CPU	MIC	CFA	СРК	PWR	STOR	CHNL	IFA	FEAT	CNSL	REC	DIAG	REF
Preface Documentation Plan	Model 145 General Description	SPTL SDBO Pre Asm, Asm	Microprogram Principles	Introduction Data Flow	Introduction 3210 Console Printer-	General Information Safety	General Information Functional	Model 145 1/O Channels Byte-Multiplever	Abbreviation List Disgram List	IBM 1401/1440/ 1460 and 1410/7010	Fixed Indicators Roller Indicators	Introduction Recording Modes	Testing Philosophy Basic Discussio	Voltage Distribution
Abbreviations	CPU Configuration	SDBO, Pre-Assembler	a da ser da se la seconda de la seconda En la seconda de la seconda En la seconda de la seconda	Clock Shift Register	Keyboards 3210 PK-KB	Motor Generator	Functional	Channel	Introduction Functional	Direct Control	Rotary Control Switches	Error Checking and Correction	Troubleshooting Extended Tests	Symbology 3145 Support
Index	Features 3145 Data Flow	Local Storage		Sector Ready	Integrated Attachment	Component Locations	Operations Servicing	Channels Block-	Units Theory of	Channel-to- Channel Adapter	A-H	Machine Checks and Error	Troubleshooting CPU Diagnostic	Documents Functions of
	Microprograms Error Handling	Expanded Local Storage (EXPLS)		CF Commands CF Error Checks	3215 PK-KB Integrated Attachment	Power Distribution	Information	Multiplexer Feature	Operation Maintenance		Keys	Retry Summary of	Hardware Selector Channel	Diagnose Instruction
-	Compatability with System/360	External Facilities		Initial Microprogram Load (IMPL)	Alter/Display Operations	CPU Power Conversion and Distribution			Features		Use meters	Machine Check Handling Feature	GA Diagnostic Functions	CPU Maintenance Aids
	3145 Channels General Description	A-Register B-Register and ALU		CF Removal and Replacement	Microprogram Operation	Power Frame, Power			Supplies and Interface			Stop After Log	Functions	Channel Maintenance
	Description Integrated	Z-Register and D-Register			Programming Information	Conversion and Distribution						Recovery	Functions Diagnostic Test	Aids Console-Printer
		C-Register (Control Word			Andreas Andreas Antreas	Controls and Indicators							Listings	Keyboard Maintenance Aids
		Decode) CPU Clock				Power Sequencing								IFA Maintenance
		M, N, and MB-Registers				Protection and Checking							영감에 전신하는 것. 2017년 - 1997년 - 1997년 - 1997년 - 1997년	Aids 370
		Secondary Control				Checks								Microprogram Load
		Assembler Control Registers				Removal and Replacement		• •						Major Functional Uni Card
	8 -	Standard Features		al dan sa Tanggan Tanggan sa	Aliman dan seria	AC/DC Module Removal and Replacement								Assignments Plug Card
						Blowers and Air Filters				3 6.	19 - A.			Information MST Wiring
					an a	Motor Generator Removal and								Change Procedure
						Replacement Preventive							La statisti Statistica	370 Microprogram Index
						Maintenance Motor Generator								
						Troubleshooting Power Check								
				a latin a suite anna anna anna anna anna anna anna an		Troubleshooting		I	L					

INTRODUCTION

CONTENTS

SYSTEM/370 MODEL 145 -	
GENERAL DESCRIPTION.	INTR 2
Features	INTR 3
3145 Models GE, GFD, H, HG, and I	INTR 3
3145 Models H2, HG2, 12, 1H2, J2, J12, and K2	INTR 4
OVERALL DATA FLOW	INTR 6
3145 DATA FLOW	INTR 7
Storage	INTR 8
Main Storage	INTR 8
Control Storage (CS)	INTR 8
Storage Data Bus-Out Assembler.	INTR 9
C-Register	INTR 9
M-Register	INTR 9
N-Register	INTR 9
MB-Register	INTR 9
Storage Protection	INTR 9
A- and B-Registers	INTR 10
Arithmetic and Logic Units (ALUs)	INTR 10
Z-Register	INTR 10
D-Register	INTR 10
SPTL-Special External Word	INTR 10
H-Register	INTR 10
Backup and Retry External Register	INTR 10
Flush-Through Check	INTR 10
Local Storage	INTR 11
External Facilities	INTR 11
Expanded Local Storage	INTR 11
I-Cycle Controls	INTR 11
1-Buffers	INTR 11
Address Adjust and Dynamic Address Translate	INTR 11
A- and B-Local Store Compare	INTR 11

MICROPROGRAMS		INTI	R 12
Control Word Readout		INTI	R 12
Instruction/Data Readout		INTI	R 12
Control Word Functions		INTI	R 13
Branch and Module Switch		INTI	R 13
Branch Word		INTI	R 13
Branch and Link or Return		INTI	R 13
Word Move		INTI	R 13
Storage Word		INTE	R 13
Arithmetic Word		INTI	R 13
Word Type-Definition		INTI	R 13
ERROR HANDLING		INTE	R 14
Microprogram Instruction Retry		. INTE	R 14
Error Checking and Correction (ECC)		INTE	R 14
Channel Retry		INTE	3 14
Command Retry		INTE	14
COMPATIBILITY of MODEL 145	with OTH	ER	
SYSTEM/370 MODELS and SYST	EM/360	INTE	R 14
Control Registers		INTE	14
Program Status Word Changes		INTE	14
STANDARD INTERFACE		INTE	ł 15
Data-In	· • • ·	INT	₹ 15
Disconnect-In		INTE	15
Mark-0-in	• • • ·	INTE	15
Data-Out		INTE	15

3145 CHANNELS – GENERAL DESCRIPTIC	ΟN	l			•	•	•	•	۰.		INTR 16
Standard Features											INTR 16
Optional Features											INTR 16
Byte-Multiplexer Channel .											INTR 16
Data Rates							2				INTR 16
Selector Channels	÷			÷	÷					Ŀ	INTR 16
Data Transfer							Č.		Ţ.		INTR 16
Data Rates								Ċ		•	INTR 16
Block-Multiplexing Feature			•			•	•	•	•	•	INTR 17
Block Multiplexing	•	•	•	•	•	•	•	•	•		INTR 17
Block-Multiplexer Operation	'n	•	:	:	:	:	:	:	:	:	INTR 17

INTEGRATED FILE ADAPTER. INTR 17

3145 TM INTR 1

GENERAL DESCRIPTION INTR 2

SYSTEM/370 MODEL 145 - GENERAL DESCRIPTION

The System/370 Model 145 is a high-availability, general-purpose data processing system that provides the reliability, availability, performance, and convenience demanded by both business and scientific users. This is achieved by:

- Using monolithic system technology (MST) circuitry. All system storage-local, control, and main-is implemented using monolithic technology.
- Providing logout information. Hard copy is available under console switch control, Programming determines whether the logout information is to be written on some I/O device (disk, tape, etc).
- Providing microprogram retry, Detected CPU hardware errors can be retried automatically by CPU retry hardware, CPU retry is accomplished by additional microprogram routines and hardware,
- Providing error checking and correction (ECC). Correction circuitry for both main and control storage automatically corrects single-bit errors. Double-bit storage errors are detected, the error location is indicated in fixed storage, and a machine-check interrupt occurs.
- Providing expanded machine-check interrupt facilities to facilitate better error recording and recovery procedures,

3145 PROCESSING UNIT (CPU) contains from 112k to 2048k bytes of main storage, plus 32k for control storage. Monolithic control storage is reloadable and is used to contain the microprogram necessary for system operation. Control storage is not accessible to the user.

The CPU contains all hardware controlled by the microprogram, necessary to decode and execute the System/370 instruction set and, optionally, those in the hardware compatibility features required by the 1401/1440/1460 and 1410/7010 emulator programs. All CPU and channel operations are controlled by the microprogram contained in control storage loaded from the console file. The CPU clock is basically one main oscillator. Its pulses are distributed to each board. Each board clock then develops the basic timing signals to time the CPU circuitry. These clocks have four basic cycle times: 202.5, 247.5, 292.5, and 315 nanoseconds.





SYSTEM CONTROL PANEL contains the operator panels, the lights, and the switches used during check-out and maintenance of the system. This console contains roll charts that can be checked to determine the status of different conditions using the same lights. For a detailed description of the system control panel, refer to "System Console (CNSL)."

Either Console Printer-Keyboard can be used. For information on integrated attachment for these Console Printer-Keyboards, refer to "Console Printer-Keyboard (CPK)."



3215 CONSOLE PRINTER-KEYBOARD is a wire-matrix printer that prints 85 characters per second.



3210 CONSOLE PRINTER-KEYBOARD MODEL 1 uses the SELECTRIC [®] I/O II printing unit. The print element contains 88 characters arranged in an optimized pattern to provide fast response. The printing speed is 15.5 characters per second.



CONSOLE FILE through an integrated attachment is used to load control storage with either the System/370 microprogram for customer operations or with the microdiagnostic used to check out the CPU. For details about this integrated attachment, refer to "Console-File Adapter (CFA)."

FEATURES

3145 Models GE, GFD, H, HG, and I

Standard Features

3145 Processing Unit (with main power frame) 160k to 512k Bytes of Main Storage 32k Bytes of Control Storage

Model	Main Storage	Control Storage
3145FED*	114,688 bytes (112k)	
3145GE	163,840 bytes (160k)	
3145GFD	212,992 bytes (208k)	See Note 1
3145H	262,144 bytes (256k)	
3145HG**	393,216 bytes (384k)	
31451**	524,288 bytes (512k)	
and the second		

*Withdrawn from the product line.

**See Note 2.

Audible Alarm Byte-Multiplexer Channel **Byte-Oriented Operand Channel Indirect Data Address** Channel Retry Information Console File Control Registers **Dynamic Address Translation** Error Checking and Correction (ECC) Extended Control Interval Timer Machine-Check Handling Microprogram Instruction Retry OS/DOS Compatibility Program-Event Recording Selector Channel 1 (without IFA) or Selector Channel 2 (with IFA) Storage Protection (Store and Fetch) System 370 Commercial Instruction Set Time-of-Day Clock

Note 1: The standard control-storage is 32,768 bytes (32k). The system is equipped with a movable controlstorage boundary that allows up to 64k bytes of control storage, depending upon the mix of features installed. These additional control-storage requirements are at the expense of main storage. The storage boundary is set automatically as a function of the initial microprogram load (IMPL) function. For additional information, refer to "Control Storage Requirements." Note 2: Main storage above 256k bytes is contained in the 3345 Storage and Control Frame Model 1 or 4 (128k additional bytes for 384k bytes total) or the 3345 Storage and Control Frame Model 2 or 5 (256k additional bytes for 512k total). When any of these units is included, it contains the low-order storage addresses. The 3046 Power Unit is required for these models.

Both main and control-storage are equipped with error checking and correction (ECC).

Optional Features

Additional Byte-Multiplexer-Channel Subchannels (Note 3) Block-Multiplexer Channels (up to four) (Note 4) Channel-to-Channel Adapter Clock Comparator and CP!/ Timer Conditional Swapping Feature Direct Control (with external interrupt) System/370 Floating-Point Instruction Set and Floating-Point (includes extended precision) Integrated File Adapter (IFA) (Note 5) Integrated Storage Control (contained in 3345 Models 3, 4, and 5) Selector Channels 2, 3, and 4 (without IFA) or Selector Channel 3 (with IFA) (Note 5)

Virtual Machine Assist Feature

Word Buffer (Note 6) 1401/1460, 1440 Emulator 1401/1460, 1440 Emulator 3210 Console Printer- Keyboard Model 1 (Note 7) 3210 Console Printer-Keyboard Model 2 (Note 7) 3215 Console Printer-Keyboard Model 1 (Note 7) 3345 Storage and Control Frame Model 1 or 4 (for 384K System)(Note 8) 3345 Storage and Control Frame Model 2 or 5 (for 512K System)(Note 8)

Note 3: The byte-multiplexer channel has 16 subchanneis that address up to 136 I/O devices (eight shared UCWs can address up to 16 devices each; eight unshared UCWs can address one device each). Up to eight control units can be attached. Configurations with 32, 64, 128, or 256 subchannels are available.

Note 4: A block-multiplexer feature can be ordered in place of all installed selector channels. Block-multiplexer channels cannot replace selector channels 1 and 4 when integrated file adapter is installed.

Note 5: The integrated file adapter and selector channels 1 and 4 are mutually exclusive. Each selector channel addresses up to 256 I/O devices. Up to eight control units can be attached. Note 6: The word buffer feature is installed on all selector or block-multiplexer channels or none. The word buffer feature is not available for the integrated file adapter feature.

Note 7: The 3210 Model 1 and the 3215 are mutually exclusive; one is required. The 3210 Model 2 can be used as an auxiliary printer-keyboard (except for alter/ display functions).

Note 8: A 3046 Power Unit Model 1 is required for all configurations having a 3345 Storage and Control Frame Model 1, 2, 4, or 5.

3145 Models H2, HG2, I2, IH2, J2, JI2, and K2

Standard Features

3145 Processing Unit (with main power frame) (Note 1) 256k to 2048k Bytes of Main Storage 32k Bytes of Control Storage

Model	Main Storage	Control Storage
3145 H2	262,144 (256k)	
3145 HG2	393,216 (284k)	Note 2
3145 12	524,288 (512k)	
3145 IH2	786,432 (768k)	
3145 J2	1,048,576 (1024k)	
3145 JI2	1,572,864 (1536k)	
3145 K2	2,097,152 (2048k)	

Audible Alarm Byte-Multiplexer Channel Byte Oriented Operand Channel Indirect Data Address Channel Retry Information Console File **Control Registers Dynamic Address Translation** Error Checking and Correction (ECC) Extended Control Interval Timer Machine-Check Handling Microprogram Instruction Retry OS/DOS Compatibility Program-Event Recording Selector Channel 1 Storage Protection (Store and Fetch) System/370 Commercial Instruction Set Time-of-Day Clock

Note 1: The 3145 Models H2, HG2, I2, IH2, J2, J12, and K2 require a 3047 Power Unit Model 1.

Note 2: The standard control storage is 32,768 bytes (32k). The system is equipped with a movable control-storage boundary that allows up to 64k bytes of control storage, depending upon the mix of features installed. These additional control-storage requirements are at the expense of main storage. The storage boundary is set automatically as a function of the initial microprogram load (IMPL) function. For additional information, refer to "Control Storage Requirements."

Both main and control storage are equipped with error checking and correction (ECC).

Optional Features

Additional Byte-Multiplexer-Channel Subchannels (Note 3) Advanced Control Program Support Feature (Note 4) Block-Multiplexer Channels (up to four) (Note 5) Channel-to-Channel Adapter Clock Comparator and CPU Timer Conditional Swapping Feature Direct Control (with External Interrupt) System/370 Floating-Point Instruction set and Floating Point (includes extended precision) Integrated Storage Control Selector Channels 2, 3, and 4 Virtual Machine Assist Feature Word Buffer (Note 6) 1401/1460, 1440 Emulator 1401/1460, 1440 and 1410/7010 Emulator 3210 Console Printer-Keyboard Model 1 (Note 7) 3210 Console Printer Keyboard Model 2 (Note 7)

3215 Console Printer Keyboard Model 1 (Note 7) Note 3: The byte-multiplexer channel has 16 subchannels that address up to 136 I/O devices (eight shared UCWs can

address up to 16 devices each; eight unshared UCWs can address one device each). Up to eight control units can be attached. Configurations with 32, 64, 128, or 256 subchannels are available.

Note 4: Available only on 3145 Models IH2, J2, JI2, and K2.

Note 5: A block-multiplexer feature can be ordered in place of all installed selector channels.

Note 6: The word buffer feature is installed on all selector or block-multiplexer channels or none.

Note 7: The 3210 Model 1 and the 3215 are mutually exclusive; one is required. The 3210 Model 2 can be used as an auxiliary printer-keyboard (except for alter/display functions).



This is a high-level data flow of the 3145 CPU. The general layout is such that you can easily reference from any area of this high-level data flow to the same area of the overall data flow. The facing page illustrates the overall data flow for the 3145 CPU.

The number of bits entering or leaving a function or register is identified either by the wight (thickness) of the line (see the legend block at the lower center of the overall data flow) or by placing the number of bits in the flow line.

This section gives a brief description of the 3145 functional units illustrated in the high-level data flow. Some basic facts to be introduced for overall understanding of the system follow. When power is first applied to the system, an automatic

- Initial Microprogram Load (IMPL) occurs. Data is read from the console-file disk into the external assembler A. From the assembler the data moves to the A-Register I, to the A-Byte Assembler, out on Storage Data Bus-In SDBI and then the data is loaded into control storage.
- The 3145 is a word length processor. Each time storage is accessed, a doubleword is read out. Through address bit format, the even or odd address word is gated and used for that specific operation.
- Normal addressing of storage is through the B-Register D to the M-Register.
 The M-Register sets up the address of main or control storage to be accessed. Control storage is a reserved area and is unavailable to the user. All addresses are validity checked.
- The microprogram resides in control storage and is read out into the C-Register.
 This control word information provides clock cycle length and sets up hardware gating controls for the handling of data. The clock is a variable cycle clock designed to accommodate operations requiring longer cycle times.
- Local Storage (LS) Consists of two monolithic stacks of 64 words each (A-LS and B-LS). Destined data is written into both A-and B-LS so that both stacks contain the same information at any corresponding address. This permits comparison checking of LS data.
- •The External facilities are composed of registers, buses, status lines, and other circuitry that form the communication line between the microprogram and:

Channels (CHNL) Console-File Adapter (CFA) Console Printer-Keyboard (CPK) Checking Facilities Retry Circuits

Integrated File Adapter (IFA) Features. External facilities have restrictions associated with them because of the manner in which they are used. For example, certain externals cannot be addressed as destinations for data. Others cannot be sources for data. • The arithmetic and logic unit (ALU) II performs the logic manipulations and adding operations in the CPU. Two ALU units are provided to allow halfword binary and word-move operations in one pass. Each ALU consists of A and B entry gates, logic and arithmetic circuits, and output gating to position the output byte in the Z-Register. • The SPTL 10, special external registers, are used for direct and indirect addressing, byte selection, and status indications. The SPTL registers are destined in the same machine cycle.



3145 DATA FLOW INTR 8

STORAGE

For details about storage, refer to "Storage (STOR)." Storage for the Model 145 is implemented by monolithic technology. It is based on bipolar, semiconductor storage cells with nondestructive read capabilities. Unlike magnetic core storage, the content of storage is lost when power is turned off.

- The main advantages of semiconductor storage are: • price/performance
- reliability and serviceability (a storage card can easily be replaced).

Storage Structure

All 145 storage (local, control, and main) is implemented using monolithic technology.

Main Storage

The 3145 processor has the following models and storage size options:

Model	Storage Si
FED	112k
GE	160k
GFD	208k
H or H2	256k
HG when used with a 3345 Model 1 or 4, or HG	2 384k
I when used with a 3345 Model 2 or 5, or 12	512k
IH2	768k
J2	1024k
JI2	1536k
K2	2048k

*112k withdrawn from product line.

Data-Transfer Times

Storage data path width is eight bytes. The CPU fetches a doubleword in 540 nanoseconds. It stores one word in 607.5 nanoseconds. (This involves fetching eight bytes, updating four of them, resetting the ECC, and then storing back.)

Control Storage (CS)

The amount of control storage required depends upon features in the system.

Basic system microcode contains:

- Standard instructions Standard features
- Patch area and routine
- and requires 24,600 bytes of CS.

Additional storage is required for:

Byte and block MPX UCWs Block MPX feature Console support Integrated file adapter (IFA) 14XX/7010/DOS emulators Floating-point



Direct control

Clock comparator

Advanced control program Sample requirements

Gampie requirementa

CS is assigned in the high-order range of available storage.

For example: 160k Model 145

	Amount	Address Range
Main Storage	160k	0-160k
Control Storage	32k	160k-192k

Instructions accessing control storage cause address checks; attempting to access a control word from main storage causes a machine check. This is checked by comparing the address with the setting of the address check boundary register (ACB). The ACB register is part of the external facility.

Reloadable Control Storage (RCS)-Advantages

flow) allows greater serviceability.

- Amount of control storage needed is minimized by recording a console-file cartridge for each customer according to the features the customer orders.
- · Engineering changes to the microprogram can be easily effected.
- One storage system (single addressing design, circuits, data
- Functions implemented through RCS can be easily extended.

Speed

Fetch cycle: 540 ns, 8 bytes.

RCS SIZE

32k bytes is the minimum. RCS may be expanded in increments of 2k bytes up to 64k at the expense of main storage.

The location of the RCS/main storage boundary is established by microcode at IMPL time.

For details of items on this pace to "CPU Hardware (CPU)," For storage protection, refer to "corage (STOR)."

STORAGE DATA BUS-OUT ASSEMBLER

• The Storage Data Bus-Out (SDBO) preassembler receives a doubleword of data from main storage.

4 *

- The output of the SDBO preassembler is gated to provide word, halfword, or byte selection.
- The SDBO assembler receives input data from the SDBO preassembler, the storage-protect stack and the D-register. It provides an output that is used as data for external bus-in (EBI) and local storage.

C-REGISTER

The C-register decodes the control word and provides control and gating of CPU functions. When read out of control storage and gated to the C-register, the control word is decoded to determine:

- Word type
- CPU function
- CPU clock cycle and length

The C-register is set through the secondary control assembler during certain operations; for example, manually setting a control word from the switches on the console.

M-REGISTER

- Composed of M1, M2, and M3, which provides a 21-bit (plus three parity bits) storage address.
- Addresses main and control storage. Storage is read out on a doubleword boundary and stored on a word boundary.



N-REGISTER

- Composed of N2 and N3.
- Backup register for control-storage addressing.
- N2 is set with the same information as M2 and is changed only when the control word being executed performs a module-switch function.
- N3 is set with the same information as M3.
- N is not changed when a trap occurs.
- When a trap occurs, the M-register is set to the trap address. The trap routine stores the contents of N (the N-Reg contains the next address that would have been used had the trap not occurred). At the end of the trap routine, M and N are restored to their original value so that the controlword sequence may continue as if there had not been any trap.

MB-REGISTER

- Composed of MB2, MB3.
- Set with the control-word address in M2, and M3 from M2 BFR and M3 BFR.
- When the CPU clock is stopped, MB contains the address of the last word executed.
- MB output is available to the retry and backup circuits as well as the external assembler.

STORAGE PROTECTION

The storage-protect unit has a 64 x 8 monolithic storage protection stack that applies to main storage locations zero through 131,072 (in sequential blocks of 2,048 bytes). Additional stacks are provided in the CPU when mainstorage capacity extends from 131,072 bytes to 524,288 bytes. Above 524,288 bytes, storage protect is located in the Power Frame (03) and is a mix of 64 x 8 and 64 x 18 monolithic storage cards. Storage-protect circuits prevent accessing protected areas during either store or fetch operations. To protect specific areas of storage, key bits are first stored in the array of the storage-protect circuit by a write-key operation. During a subsequent store or fetch operation, one of the prestored keys is accessed and compared with the key provided by the user. If the keys match, access to data storage is granted; if not, access is denied.

3145 DATA FLOW INTR 10

For details of items on this page, refer to "CPU Hardware (CPU)."

A- AND B-REGISTERS

The A- and B-registers, each with a fullword capacity, provide the primary data inputs to the ALUs.

- The B-register also feeds the M-register during address setup:
- In the first cycle of a storage word, or
- During a return function in which the return address is taken from local storage or an external facility.

ARITHMETIC AND LOGIC UNITS (ALUs)

Two one-byte ALUs (ALU2 and ALU3) perform the following operations in one CPU cycle:

- Binary addition, true or complement, of up to two fullword operands. Two halfword additions are performed to achieve the fullword add. Binary halfword addition is achieved by gating the two low-order operand bytes of each halfword into ALU3, and two high-order operand bytes of each halfword into ALU2.
- Logical operation on two 1-byte operands. The operation can be AND, OR, or Exclusive OR.
- One-byte, packed-decimal addition (true or complement).
- Operations and microprogram symbols are:

Symbol	Operation
.A,	AND
,OR,	OR
,OE,	Exclusive OR
+	True ADD
-	Complement ADD
,D+-,	Decimal ADD
+.	Binary ADD
,A-,	Complement AND

Z-REGISTER

ALU results are set into the four-byte Z-register. The ALU result data can then be routed from Z to:

- The D-register (normal gating)
- The S, P, T, or L-registers
- The A- or B-registers.

Also, the Z-register data (that is, ALU result) is tested, if so specified in the control word being executed, to set/reset S-register bits.

D-REGISTER

The D-register is used as an interim destination for data to be routed to external facilities or local storage. The data leaves the D-register on the following control-word cycle.



SPTL--SPECIAL EXTERNAL WORD

- Addressed directly by control-word bits.
- Has special data path to A- and B-register inputs.
- Only external that can be used as a B-source.
- Only facility that is destined in the same cycle (except H-Reg).
- Composed of four one-byte registers: S, P, T, and L.

S-Register: holds the status of arithmetic and logical results; controls some arithmetic functions.

P-Register: base address register for local storage and external addressing.

T-Register: used in conjunction with special branch functions, shifting, storing, and indirect-byte addressing.

L-Register: used in conjunction with P-high bits to form indirect local-storage addresses.

H-REGISTER

The setting of the latches in the H-register is used to determine the priority of traps. The bits are set during trap-2 cycle and prevent traps of lower priority from occurring.

BACKUP AND RETRY EXTERNAL REGISTER

To allow recovery from certain kinds of errors, hardware registers (externals) are provided. The backup registers are set to the current cycle setting of the prime register; the retry registers are set to the cycle prior to the one currently being executed.

FLUSH-THROUGH CHECK

Data routed to local storage, as the result of some control-word operation, other than a storage word read, is gated from the D-register through the SDBO assembler **1** to local storage. The data that is stored in A-local storage is set into the Flush-Through Check (FTC) latches and is matched to the data routed from the D-register. If the match is unequal, an error condition is set. The same check is made on information routed to the external facilities from the D-register.

For details about items on this refer to "CPU Hardware (CPU)."

LOCAL STORAGE

- A and B local storage are identical monolithic stacks of 64 words each.
- Both stacks contain the same information at the corresponding address. This enables checking to ensure that data being operated on is correct.
- The microprogram uses the local-storage area as a buffer between main storage and the CPU hardware.
- Addresses are formed with combinations of bits from the control word, P-register, L-register, T-register, console-file command register, and forced by the selector channel.
- Access time is 24 nanoseconds.

EXTERNAL FACILITIES

 External facilities are composed of registers, buses, status lines, and other circuitry that form the communications line between the microprogram and:

Channels Console file

Documentary console Checking facilities Retry circuits Integrated file adapter Features

- Features
 - Addresses are formed from control words, console-file data, selector-channel circuits, console switches, retry information, and local-storage address data.
 - Data from the externals enters the data flow through the external assembler to the A-Reg.
 - Data to the externals is gated through the SDBO assembler on the external bus-in (EBI).

EXPANDED LOCAL STORAGE

The expanded local storage (EXPLS) registers are hardware registers addressed as though they are local-storage (LS) registers. EXPLS operates similar to the external facilities; however, the output is routed to the A- and B-registers similar to LS.

I-CYCLE CONTROLS

This hardware improves the CPU performance for System/360 and System/370 instructions by reducing the time during I-Phase of instruction processing.

I-BUFFERS

The I-buffers consist of three 1-word registers and are used to hold the present instruction plus, in most cases, the next doubleword of the instruction stream. ADDRESS ADJUST AND DYNAMIC ADDRESS I

This logic is used by the OS/DOS compatibility and Dynamic Address Translation features. OS/DOS compatibility feature uses the logic for executing a DOS supervisor and DOS programs under control of the OS supervisor in any mainstorage location. Dynamic Address Translation uses the logic to make available by software and hardware up to 16M of virtual storage.

WIND B-LOCAL STORE COMPARE

Data stored in local storage is located at the corresponding address in both local-storage stacks. The data is read from both stacks and compared. If the data does not compare, an error condition is set. Note that the output of expanded local storage is routed along the same path but is not compared.



MICROPROGRAMS INTR 12

MICROPROGRAMS

For details about microprograms, refer to "Microprogram (MIC)."

- All functions performed by the 3145 are controlled by a microprogram.
- Before any processing may begin, the microprogram must be loaded into the control-storage area.
- The microprogram is loaded into control storage from a disk that is read by the console file.
- This loading process is called Initial Microprogram Program Load (IMPL).
- The microprogram is composed of microroutines of varying sizes, each having a specific task to perform.
- The microprogram handles the processing of the instructions and data that are read into the main-storage area.
- Channel operations and the operations of the integrated devices are also handled by the microprogram.
- Each microroutine is composed of bit-significant control words that handle particular functions. These functions control execution of the specified task of the microroutine.

CONTROL WORD READOUT

Before a control word can perform any of its functions, it must be set into the four-byte control register (C-Reg). The outputs of the C-Reg activate circuitry that causes the execution of specified data-flow functions.

Control words are normally read from control storage and set into the C-Reg. However, control words can be set into the C-Reg directly from the console file, and certain control-word bit combinations may be forced into the C-Reg by circuitry. Assume that control and main storage have been loaded and that processing has begun:

- 1. Control words are read out of control storage and gated out on the Storage Data Bus-Out (SDBO).
- 2. Portions of the control words are gated from the SDBO to either the local storage control assembler D or the external control assembler D. This is done to set up source addresses for these facilities early in the cycle.
- 3. The control words are gated into the control register (C-Reg) I, where they are decoded. Decoding the control words brings up control and addressing lines that access and execute the programs located in main storage.



INSTRUCTION/DATA READOUT

When a control word performs a read operation on main storage, either instructions or data is accessed. All read operations, for control or main storage, result in a doubleword's being accessed from storage.

- Assume that a control word is performing a read operation on main storage:
- 1. The doubleword from main storage is gated out on the SDBO I to the SDBO pre-assembly latches I.
- 2. The odd or even address word, of the doubleword, is selected and gated to the SDBO assembler.
- 3. If the word selected is a data word, it is gated to local storage G or some external facility II.
- 4. If the word is an instruction, it is gated to the I-buffers **II**, expanded local storage **II**, and in some cases to the address adjustment circuits **II**.

CONTROL WORD FU IONS

For details about control words, refer to "Microprogram (MIC)". The control words and their high-level functions are:

Branch and Module Switch

- Functions: Branch
- Module switch
- Destine data to the S, T, or L-registers

Branch Word

- Functions:
- Branch
- Module switch (special function).
- Set/reset bits in local storage or external registers

Branch and Link or Return

Branch and Link Functions:

- Store S, P, N2, N3 into a link register
- · Set P with a value, or module switch
- Branch
- **Return Functions:**
- Restore S, P, N2, N3
- · Reset H-register bits
- Alter the link address in some cases

Word Move

- Move a fullword or selected bytes from one local storage/external location to another.
- Branch

Storage Word

Functions:

- Read data from or store data into-Local storage Main storage
- Control storage
- External registers
- Storage protect stack
- Branch



- Type 10 Functions:
- Perform a variety of arithmetic and logical operations
- Operate on fullwords for arithmetic or shifting operations
- Type 11 Functions:
- Operates on bytes only
- · Performs OR, or true ADD only
- Provides A-register input crossing



2

3145 TM INTR 13

ERROR HANDLING

If application-program errors occur (such as illogical action requests), the operating system attempts to handle the exception and provide any necessary operator messages.

If a failure occurs within the CPU or an I/O unit, provisions are made to retry the failing operation. Error-logout facilities to record any such failures are incorporated into the system. This is in addition to any provisions made by the operating system for error retry and error logging.)

Microprogram instruction retry, limited and extended channel logout, storage validation (Error Checking and Correction-ECC) the main and control storage, and other error-detection and error-handling provisions are standard.

MICROPROGRAM INSTRUCTION RETRY

The ability to recover from most intermittent failures is provided by retry techniques. CPU retry is done by microprogram routines that save the source data before it is altered by the operation. When an error is detected, a microprogram routine returns the CPU to the beginning of the operation (or to a point during the operation that was executed correctly), and the operation is repeated. For a detailed description of microprogram instruction retry, refer to the "Recovery Features (REC)."

ERROR CHECKING AND CORRECTION (ECC)

Error checking and correction circuitry for main and control storage automatically corrects single-bit errors. Automatic detection of double-bit errors is also provided. For a detailed description of ECC circuits, refer to "Storage (STOR)." For a description of handling ECC errors, refer to "Recovery Features (REC)."

CHANNEL RETRY

This feature ensures that most failing channel operations can be retried by error-handling routines. Both a limited and an extended channel logout are implemented. When a channel error or a CPU error associated with a channel operation occurs, the channel status word (CSW) and an extended channel status word (ECSW) are stored in the fixed lower storage area during the I/O interrupt. The ECSW or limited channel log out data provides additional, more exacting status information about the channel failure. This data is formatted by the channel check handler (CCH) routine and passed to a device-dependent error recovery routine to be used in the retry of the failing I/O operation. The ECSW contains information as to:

Which unit detected the error Which unit caused the error Successful retries Channel retries Validity flags Retry code-how far has the instruction progressed in execution

COMMAND RETRY

Command retry is a control-unit-initiated procedure between the channel and the control unit. (Not all control units have this capability.) No I/O interruption is required. The number of retries is device-dependent. Within the storage capacity, internal and input/output channel processing rates, and type of input/output devices that can be attached, compatibility is maintained with other System/370 and System/360 models, with the following exceptions.

COMPATIBILITY of MODEL 145, with OTHER SYSTEM/370 MODELS

- 1. Programs using machine-dependent data (for example, machine logouts).
- 2. Programs using the ASCII bit (PSW bit ...).

and SYSTEM/360

- Programs that depend upon features or I/O devices that are not implemented on this system (such as special instructions for ths System/360 Model 44).
- 4. Programs that depend upon validity of data after the system power has been turned off and restored.

Programs written for other System/370 or System/360 models that contain the following conditions or requirements should be evaluated on an individual basis to ensure proper operation.

1. Time-dependent programs,

- 2, Programs written to cause deliberate program checks.
- 3. Programs that depend upon model-dependent features of other System/370 and System/360 models.
- Programs that use storage locations between address 128 (decimal) and 704 after a diagnostic logout into program storage. However, such programs may be executed if:
 Check-control switch is set to Stop After Log
- a. Check-control switch is set to stop After Log position. In this case, processing stops after the diagnostic logout into program storage takes place.
- b. Program-storage locations that are overlaid by the diagnostic logout are restored with the program requirements followed by an appropriate program restart procedure.

Any attempt to continue processing after a dianostic logout to program storage without restoring your program information to the logout area has unpredicatable results. The 705-byte extent (the permanently assigned programstorage locations) can be reduced to 512 bytes by moving the 192 bytes (between locations 512 and 704) into another program-storage area. The technique used to accomplish this relocation depends upon your application.

CONTROL REGISTERS

The control registers provide for loading and storing control information.

The structure provides for sixteen 32-bit registers for control purposes. These registers are not part of addressable storage. One or more specific bit positions in control registers are assigned to each function requiring register space. Some of these functions and registers are:

Making the timer interrupt and external interrupt in control register 0.

Masking machine-check subclasses by bits set in control register 14.

Pointing to an extended CPU logout area in control register 15.

For details of the control registers, refer to "CPU Hardware (CPU)."

PROGRAM STATUS WORD CHANGES

Bit 7 External Mask bit is now a summary bit with control register 1 conataining the individual mask bits.

Bit 12 is now reserved and must be zero. ASCII code is removed.

Bit 6 is the mask bit for channels 6 and over.

Bit 13 Hard Stop bit is now a summary bit. Control register 14 contains mask bits for subclasses of machine checks.

ERROR HANDLING-3145 COMPATIBILITY INTR 14

The standard interface for System/370 has all the lines used on the System/360 standard interface, plus several additional lines. The additional lines used by the 3145 are identified on this page. For details of these lines, refer to "Channel (CHNL)."

DATA-IN

During read and sense operations, Data-In rises when data is available on Bus In. During write and control operations, Data-In indicates that the control unit is ready to receive data.

Data-In indicates to the channel that data on Bus-Out was accepted by the control unit or that the control unit provided the requested data on Bus-In.

Data-In is effective with selector/block-multiplexer channels only and is used along with the Service-In Tag line to increase data rates,

DISCONNECT-IN

Disconnect-In provides control units with the ability to alert the system of malfunctions.

The channel responds to Disconnect-In by performing a selective reset.



Data-Out is the response to Data-In.

Data-Out indicates to a control unit that data on Bus-In was accepted by the channel or that the channel provided

channels only and is used along with the Service-In Tag line to increase data rates.

CHANNELS INTR 16

3145 CHANNELS--GENERAL DESCRIPTION

For details, refer to "Channel (CHNL),"

The Model 145 has two types of channels available:

- Byte Multiplexer
- Selector The selector channel optionally may have the blockmultiplexer feature attached.

Channels on the Model 145 are integrated in the CPU and share CPU cycles for I/O operations.

STANDARD FEATURES

Byte multiplexer channel

- Selector channel 1 (Channel 2 if the IFA is present)
- Channel retry

OPTIONAL FEATURES

- Selector channels 2-4 (only selector channel 3 if IFA is present).
- Block-multiplexer feature.
- Integrated File Adapter for 2319 DSF. (Displaces Channels 1) and 4.)
- Channel-to-Channel Adapter.

BYTE MULTIPLEXER CHANNEL

Functionally is equivalent to the System/360 multiplexer channel.

Data transfer is on a byte basis only,

UCWs (Unit Control Words) are provided for subchannels in control storage. Each UCW is contained in four words (16 bytes).

UCWs provide a place to store channel register data between data transfers, thus allowing multiplexing of data.

• A maximum of 256 subchannels is available on the Model 145.

• 16 UCWs are standard on the Model 145. A shared UCW can be shared by up to 16 devices, of which only one can operate at a time.

A non-shared UCW can be used by one device only.

Thus, with 16 UCWs, if 8 are shared and 8 non-shared, a total of 136 I/O devices can be attached.

Up to eight control units can be attached per channel.

 Configurations of 32, 64, 128, or 256 subchannels are available. The number of subchannels must be specified so that the proper amount of control storage may be allocated and written on the console file.



Data Rates

- Aggregate data rate in byte mode is 50 kb. Note that the selector channels and IFA can interfere with the byte multiplexer channel.
- Burst mode data rate is 180 kb.

SELECTOR CHANNELS

The 3145 has one selector channel as standard. Up to three more may be attached as an optional feature. DASD devices without command retry feature should not be attached to channel 4.

- Functionally is equivalent to the System/360 selector channel.
- If the IFA is installed, only channels 2 and 3 can be installed.

Data Transfer

Data is transferred one byte at a time unless the optional word buffer feature is installed on the channel,

- A four-byte buffer is provided for each channel if the feature is installed.
- The word buffer feature allows fewer accesses to main storage to be made while transferring data from the selector channels and increases channel data rates.
- A one-byte operation requires 585 nanoseconds; a onebyte fetch operation requires 517.5 nanoseconds. The word buffer feature allows four bytes to be transferred rather than one.
- The word buffer is required if the 2305 is attached. This buffer is recommended if the 3330 is attached.

Data Rates

- Single Channel without word buffer .82 megabytes with word buffer 1.85 megabytes
 - without word buffer 1.5 megabytes with word buffer 5.0 megabytes
- Aggregate data rate

BLOCK-MULTIPLEXER FEATURE

The block-multiplexer feature may be installed on the selector channels as an optional feature.

It is required if the 3330 and 2305 are attached. The selector channel operates as a block-multiplexer channel when the mode bit in the control register is set on.

A maximum of 512 UCWs is provided when the blockmultiplexer feature is installed. These UCWs provide a pool that may be assigned to devices. Each UCW is contained in two words.

- UCWs may be shared or non-shared
- UCWs are contained in control storage
- Shared UCWs must be determined and assigned device addresses,
- Non-shared UCWs are dynamically assigned in blocks of eight to devices at start I/O time. If no UCWs are available, a not-operational-condition code is returned.
- UCWs are provided in control storage in increments of 16 UCWs. Each increment contains two groups of 8 UCWs.

Block Multiplexing

Block multiplexing allows the channel to disconnect a device at channel-end time. During the interval between channel-end and device-end, another device on the channel could be started or could complete data transfer for a previously started operation. Thus, a block-multiplexer channel can multiplex blocks of data from different devices giving a much greater effective data rate than a selector channel.

Block multiplexing occurs only if a control unit presents channel-end and not device end during command chaining, and the channel is in block MPX mode.

The block-multiplexer channel can operate as a selector channel so that existing System/360 channel programs can run unchanged.

Block-Multiplexer Operation

Because the channel is busy only during the time when data is actually being transferred, several channel programs can be executed concurrently by sharing the channel hardware. This is called "Channel Multiprogramming." The sequence of events in channel multiprogramming is:

A channel program controlling a device is started by the channel and remains active until the device signals that it has no need for the channel path at that stage of its operation.

The channel disconnects the channel program and stores all information needed to restart the program in UCWs that are in control storage.

The channel can start another channel program at this point if one is ready.

Upon receipt of a signal from the previously disconnected device indicating that it is ready to use the channel data path again, the channel restarts the appropriate channel program.

The process is repeated for all active devices until each one is completely serviced.

If a channel is busy when a device reconnection is requested, the device must wait until the channel becomes available.

To facilitate channel scheduling, a new *channel available interrupt* has been defined for the block-multiplexer channel.

At disconnect time for a channel program, the channel is available for the resumption of an uncompleted channel program or the initiation of a new one. A channel available interrupt occurs at disconnect time if any I/O command was issued previously while the channel was busy.

INTEGRATED FILE ADAPTER

- The Integrated File Adapter (IFA) feature connects three to eight 2314-type disk drives to the System/370 Model 145.
- The IFA feature is assigned exclusive use of the channel-1 address and functions as both channel and control unit for the files.
- Data transfer takes place one byte at a time on a share-cycle basis the same selector channels.
- The initiation of operations and each step of the file sequence requires the CPU controls and microprogramming.

The primary control for the IFA is contained in the CPU, where it can make use of the CPU hardware and microprogram for operation. The 2319-A1 contains the read clocking circuits, the write oscillator, and the storage module switching for up to eight files. The disk storage drives operate the same as the 2314 system connected to a selector channel. The record format is identical, and the operation requires the same programming systems.

The IFA control-unit operation is initiated as a channel operation using the I/O instructions and channel commands. Primary control information for the file operation is stored in the CPU. Operating commands are processed by microprograms stored in the CPU. The microprogram starts the operation by developing the appropriate information for a portion of the sequence and issues a mini-op to the control-unit hardware. While the hardware is performing the mini-op, the microprogram stores a link address and returns to CPU operation, When the hardware finishes that portion of the sequence, it requests a trap to return to the microprogram link address to continue the operation. To complete a command, an operation may require several of these transfers between the microprogram and hardware.

Data movement during the hardware control period is performed by requesting a selector-channel share cycle for each byte. The CPU or other channel operations can use the CPU hardware and microprogram for other operations when time is not required by the IFA controls either for setup or data transfer. The file operation should never overrun during normal operation because of the assigned priorities.

For details on the IFA, refer to "Integrated File Adapter (IFA)."

CPU HARDWARE

CONTENTS COTI

SPTL	CPU 3	EXT
S-Register	CPU 4	
P-Register	CPU 6	Ex
T-Register	CPU 6	Χ.
L-Register	CPU 7	So
그는 것이 같이 많이 많이 많이 있는 것이 같이 많이 많이 했다.		Flu
		Ex
		Ex
		Ex
SDBO PRE-ASM, ASM	CPU 8	
SDBO PRE-ASSEMBLER UNIT DATA FLOW	CPU 9	
집에 가지 않는 것은 것을 알았다.		
LOCAL CTODACE	001110	
LUCAL STORAGE	CPU 10	
Local Storage Operation	CPU 10	
Data Checking	CPU 10	
Hush-Inrough Check (FIC)	CPU 10	
A and B Local Storage Compare	CPU 10	
Local Storage Liming	CPU 10	Pr
Local Storage Destination Addressing	CPU 11	
A Logal Station Llock Arread	CPU 11	
A Local Storage Address Assembly	CPU 12	
R Local Storage Address Assembly	CPU 13	
B Local Storage Addressing Assembly	CPU 14	
B-Local Storage Mon (270 Misroprogram in	CFU 15	
Control Storage Map (370 Microprogram in	CDU 16	
Soona Proceedings for Local Storage Addressing	CPU 10	
Scope Frocedure for Local-Storage Addressing	01017	I-CYC
EXPANDED LOCAL STORAGE (EXPLS)	CPU 19	I-P
Expanded Local Storage Map	CPU 21	
I-Begister (EXPLS 50)	CPU 21	
V-Register (EXPLS 51)	CPU 21	
W-Register (EXPLS 52)	CPU 21	
U Register (EXPLS 53)	CPU 21	
IBU-Register (EXPLS 54)	CPU 21	
TR-Register (EXPLS 55)	CPU 21	
ICS (I-Cycle Status) Register (EXPLS 56)	CPU 21	
G2DRL, G3DRL, G1DRL, and G4DRL (EXPLS		
60, 64, 68 and 6C)	CPU 22	
G2DBRL, G3DBRL, G1DBRL, and G4DBRL		
(EXPLS 61, 65, 69, and 6D)	. CPU 22	
SN-Register (EXPLS 78)	. CPU 22	
PN-Register (EXPLS 79)	CPU 22	
WK-Register (EXPLS 7A)	. CPU 22	
NP-Register (EXPLS 78)	. CPU 22	
DK-Register (EXPLS 7C)	. CPU 22	
Expanded Local Storage: Source Gating	. CPU 23	
Expanded Local Storage: Source Gating Examples .	. CPU 25	
Expanded Local Storage: Destining	. CPU 27	
Destination Control	. CPU 28	

EXTERNAL FACILITIES	. CPU 29
External Control Assembler	. CPU 29
X-Y Decodes	. CPU 29
Source Addressing	CPU 29
Flush-Through Check	. CPU 29
External Assembler Data Flow	. CPU 30
Expanded External Assembler Data Flow	. CPU 31
External Assignment and Index Map	. CPU 32
NOREG Word	. CPU 34
Diag Word Byte 2	. CPU 34
Diag Word Byte 3	. CPU 34
CPU Word	. CPU 34
SW Word (Console Switches)	. CPU 34
PSWCTL Word	CPU 34
Misc Word Bytes 2 and 3	. CPU 34
In Word (Interrupt Register)	. CPU 34
ACB (Address Check Boundary) Register	. CPU 35
Sys (System) Register	. CPU 38
Priority Operations-H-Register	. CPU 39
Priority Operations	. CPU 39
H-Register	. CPU 41
Priority and Trap Controls.	. CPU 43
M2 Gating (Traps)	. CPU 44
M3 Gating (Traps)	. CPU 45
-CYCLES	CPU 47
I-Phase Functions	CPU 48
Hardware Functions	CPU 49
Microcode-Hardware Functions	CPU 50
Microcode Hardware Relationship	CPU 51
I-Cycles Microcode Module Assignment	CPU 52
I-Cycles Microcode and Control Hardware	
Loading of I-Buffers	CPU 53
I-Cycles Microcode and Control Hardware	
Calculate Operand Address and Perform Prefetches	CPU 54
I-Cycle Hardware Locations	CPU 56
I-Cycle Hardware Description	CPU 57
I-Cycles Data Flow	CPU 58
Instruction Cycles Address Generation	CPU 63
I-Cycles Control Line Generation	CPU 64
I-Cycles Address Generation and Control	
Decode	CPU 65
Unique Conditions During I-Cycles	CPU 66
I-Cycle Error Conditions	CPU 66
Storage Correction Cycle	CPU 66
I-Cycle Timings	CPU 67
I-Cycle Operational Description	CPU 68
I Cycles Alignment Routine	CPU 73

Execute Phase (I-Cycles)	CPU 82
Example 2B Add (5A) Instruction (Double	•
Indexing with Alignment).	CPU 83
MVC (D2) Instruction Example	CPU 84
Execute 44 Instruction Example.	CPU 86
A.REGISTER R REGISTER and ALLIA	001100
Ancosten, Bacosten, allu ALOS.	CPU 88
A-Register and A-Byte Assembler	CPU 89
B-Register and B-Byte Assembler	CPU 90
ALU A-Entry Gating	CPU 91
ALU B-Entry Gating	CPU 91
Shift Gating	CPU 02
Al II K.Assembler	CPU 02
Arithmetia and Logic Unit (ALLI)	CFU 92
Holf Cum Charling	CPU 93
	CPU 93
Z-REGISTER and D-REGISTER	CPU 94
	5 C
2-Register Parity Checking	CPU 94
Z-Register	CPU 94
D-Register	CPU 95
D-Register and Flush-Through-Check (FTC)	CPU 96
Register	
and the second	
C-REGISTER (CONTROL WORD DECODE)	CPU 98
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 98
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 98
C-REGISTER (CONTROL WORD DECODE) C-Register M, N, and MB-REGISTERS	CPU 98 CPU 98 CPU 102
C-REGISTER (CONTROL WORD DECODE) C-Register M, N, and MB-REGISTERS	CPU 98 CPU 98 CPU 102 CPU 102
C-REGISTER (CONTROL WORD DECODE)	CPU 98 CPU 98 CPU 102 CPU 102 CPU 103
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 98 CPU 102 CPU 102 CPU 103
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 98 CPU 102 CPU 102 CPU 103 CPU 103
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 98 CPU 102 CPU 102 CPU 103 CPU 103 CPU 103
C-REGISTER (CONTROL WORD DECODE) C-Register M, N, and MB-REGISTERS M-Register Setting M-Register for Main Storage Addressing . Setting M-Register for Control Storage Addressing . N-Register MB-Register .	CPU 98 CPU 98 CPU 102 CPU 102 CPU 103 CPU 103 CPU 103 CPU 103
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 98 CPU 102 CPU 102 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 98 CPU 102 CPU 102 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 98 CPU 102 CPU 102 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 104
C-REGISTER (CONTROL WORD DECODE) C-Register M, N, and MB-REGISTERS M-Register Setting M-Register for Main Storage Addressing . Setting M-Register for Control Storage Addressing . N-Register MB-Register Buffer Registers M-, N-, and MB-Registers Branch Circuits .	CPU 98 CPU 98 CPU 102 CPU 102 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 104
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 98 CPU 102 CPU 102 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 104 CPU 99
C-REGISTER (CONTROL WORD DECODE) C-Register M, N, and MB-REGISTERS M-Register	CPU 98 CPU 98 CPU 102 CPU 102 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 104 CPU 99 CPU 99
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 102 CPU 102 CPU 102 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 104 CPU 99 CPU 99 CPU 99 CPU 99
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 98 CPU 102 CPU 102 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 104 CPU 99 CPU 99
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 102 CPU 102 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 104 CPU 99 CPU 99 CPU 99 CPU 99 CPU 99 CPU 99
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 102 CPU 102 CPU 102 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 104 CPU 99 CPU 99 CPU 99 CPU 99 CPU 99 CPU 99 CPU 100 CPU 100
C-REGISTER (CONTROL WORD DECODE) C-Register	CPU 98 CPU 102 CPU 102 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 103 CPU 104 CPU 99 CPU 100

.

STANDARD FEATURES CPU 109 Clock Security Switch (TOD CLK) CPU 109 Clock Validity Indicator (TOD CLK INVAL) . . CPU 109 TOD Circuit Card Locations and New Instructions for OS/DOS Emulator CPU 119 Feature Mask Store Then Mask Instructions CPU 138 Basic Operation of Dynamic Address Translation . CPU 140 Selector and Block Multiplexer and IFA Channels . CPU 170

The SPTL word is a special external register that has a word address of 04.

- Addressed directly by control-word bits.
- Has special data path to A- and B-register inputs.
- Only external that can be used as a B-source.
- Only facility (other than the H-Reg) that is destined in the same cycle.
- Composed of four byte registers: S, P, T, and L.
- Set by I-cycle controls during I-cycles.





Note: For details concerning how SPTL affects control-word operations and addressing, see the Microprogram (MIC) section.

S-REGISTER



S-register bits are used primarily to indicate the results of ALU operations or to specify how certain ALU operations are to be performed.

Branch fields in the control words can be set to specify branch testing of S-register bits. The results of the branch testing are used to determine a portion of the next control-word address. Therefore, a control-word sequence can be modified according to ALU results. For example, the following is frequently used in microprogram routines.

- 1. A control word specifying an ALU operation calls out set/ reset of specific S-register bits, depending upon the ALU result.
- 2. A subsequent control word specifies branching on the same S-register bits.

Note: If the S-register is set with a control word and this same control word is branching on S-bits, the branch test is made on a previous S-register setting and not on the result of the current control-word operation.

3. The control word branched to continues the microprogram sequence required by the ALU result.

The Sregister can also be used as a general-purpose data register. For example, a control word can cause the S-register to be loaded with a byte for use in operations with subsequent control-word operations such that none of the following descriptions apply. Such use of the S-register is determined by the microprogrammer. The following listed S-register bit functions are not automatically performed. Any function must be explicitly specified in the control word for which the function is desired.

The data from the S-register is used in the M-register for branching.

Note: The duplicate S-registers are designated S-register A and B, respectively. The data from S-register B is displayable, and is gated to the A-register and B-register.

SO

The setting of SO determines whether a true or complement add is to be performed in the ALU(s) when either a binary or decimal add is specified by the control word.

SO Value	Specifies	
0	True add	
1	Complement add	
The true/complement	nt circuitry affects only the B-input	(from

the B-register or the K-assembler) to the ALU(s). The A-register input is always presented to the ALU(s) in true form, regardless of the value of S0.

In arithmetic word (type 10) shift operations, the value of SO is shifted into the four high bits of the result word (shifted right) when the shift field of the arithmetic word specifies (SR4, SO). The bits (4-7) shifted out of the source-word byte 3 are set into T-register bits 0 through 3, respectively.

S1

 In decimal operations, S1 is set to 1 if an invalid decimal digit is detected in the A or B inputs to the ALU. S1 is not changed if the decimal digits are all valid. A decimal digit greater than 1001 (binary) is invalid. The test on the inputs is made before the original decimal data is binarily added in ALU3; no such test is made on decimal data when it is being sent to ALU3 on a ±6 correction cycle.

In order for S1 to function in this manner, the arithmetic control word (type 10) must specify both decimal addition (C, D+, C), and the S12 status set.

- 2. In binary operations, S1 is set to the value of the carry-out of:
 - a. Bit 1 in single-type ALU operations,

b. Bit 1 of byte 0 in fullword ALU operations.

The control word must specify the S12 status set, along with the appropriate binary ALU operation, in order for S1 to function in either of these two ways.

S2

- In byte operations, S2 is set to 1 if the Z-bus (ALU result byte 3) is not zero (ZO). If the Z-bus is zero S2 is not changed from its prior setting. The arithmetic control word calling for the byte operation must specify the S12 status set in order for the S2 bit to function in this manner.
- 2. In fullword binary operations, S2 is set to 1 if the entire 32bit result is nonzero. S2 is not changed if the 32-bit result is zero. A status set of S12 must be specified in the arithmetic control word (calling for the fullword operation) in order for S2 to function in this manner.
- 3. In fullword binary operation, S2 is set to 1 if: a. A Z24 status set is specified, and
- b. The low-order 24 bits (ALU result bytes 1, 2, and 3) are nonzero. If the low-order 24 bits are all zero, S2 is not changed.
- 4. When an S2 status set is specified in a storage word, S2 is set to 1 if the count field is not zero after the count is decremented; S2 is set to 0 if the count field is zero after decrementing. The decrement-count function is specified in the storage word to effect the following actions:
- a. The 24-bit address, in bytes 1, 2 and 3 of the even word (of an even/odd pair of local-storage words), is updated.
- b. The count field (low-order 16 bits of the odd word of the pair) is decremented.

c. S2 is set according to the low-order 16-bit result of step b. Note that the count value is updated by circuitry and that S2 is set/reset if the S2 status set is specified even if decrement count is not specified. In this case, however, the updated count value is not stored back into the count location.

S3

S3 is set to the value of the carry-out of bit-0 of the ALU operation. This function is used in both byte and word operations. The arithmetic operation field must contain a bit configuration designated by a statment that contains a C at the left, in order for S3 to function in this manner. For example, in the statement:

S0 = 0

C + 0

the C specifies that S3 is to be set/reset.

- In fullword arithmetic operations, S3 is set to the carry-out of: a. Bit 0 of byte 0 if an S12 or no status set is specified.
- b. Bit 0 of byte 1 if a Z24 status set is specified.

S3 is not set/reset in storage-word address and count updates.

S4 and S5

S4

S4

In arithmetic words (types 10 and 11) that specify a status set of S45, S4 and S5 are set/reset according to the bit values of the ALU result byte as follows:

Bit	Value	Indicates Result Byte Bits
S4	0	0-3 not equal to 0000
S4	11	0-3 = 0000
S5	0	4-7 not equal to 0000
S5	1	4-7 = 0000
lf a Ze	6 status set i	s specified:

Bit Value Indicates Result Byte Bits

- 0 0-5 not equal to 000000
- 1 0-5 = 000000
- S5 0 4-7 not equal to 0000
- S5 1 4-7 = 0000

The S45 and Z6 status sets can be specified in an arithmetic word, only if a single-byte ALU operation is called for; S45 and Z6 do not pertain to fullword arithmetic operations.

In storage-word operations, S4 and S5 functions are the same as in the arithmetic words and are specified in the same manner (S45 and Z6). S4 and S5 are set according to the value of the low-order byte of the count field after the count has been decremented. The count is in bytes 2 and 3 of the odd word of an even-odd pair of local-storage words. The address is in bytes 1, 2, and 3 of the even-word location.

In an arithmetic word (type 10) that specifies an AB CK byte operation: S4 is set to 1 if a parity-check error is detected on the A input to the ALU(s).

Note: If an I24 status set is specified in a type 10 arithmetic word, no S-register bit setting occurs, regardless of any other specified status set. For example, if the operation specified is: S0 = 0

C+0

and I24 is also specified, then S3 and S0 are not changed even though the operation field calls for such set/reset functions.

SPTL CPU 6

P-REGISTER



local-storage control.

The primary function of the P-register is to provide a base address during local-storage or external-register addressing. That is, the P-register is used to point to groups of external registers or areas of local storage; the remainder of the external/local-storage address is specified by the outputs of the C-register. In some cases, the L- and/or T-register contents are used to determine portions of the address.

T REGISTER



Used as a mask for a store-word function. Used to hold low-order hex digit shifted out during arithmetic word shift-right function. May be used as digit shifted into the highorder position of a right shifted result.

Used as a byte pointer for the A-source during arithmetic word using indirect byte addressing. Usually set by a specual function of a storage word read operation.

The T-register is used in a variety of ways:

- 1. Bits 4 and 5 and/or 6 and 7 are used in indirect-byte addressing and branching operations.
- Bits 0 and 1 are used to form a portion of the nextcontrol-word address when a module-switching operation is specified in the branch word.
- 3. Bits 0 through 3 are used in arithmetic fullword shift operations.
- 4. In certain storage word read operations, bits 4 and 5 or 6 and 7 are set to the value of the two low-order storageaddress bits before the address is updated.
- 5. In certain storage-word store operations, bits 0 through 3 are used to specify which bytes of a source are to be stored and what constant, if any, is to be used to update the storage address.
- 6. May be used as a working register.

L-REGISTER



Used with high-order bits of the P-register address of local-storage register containing second operand.

5 *

The primary purpose of the L-register is to hold the addresses of the general or floating-point registers, all of which are in local storage. The address of a general register can be specified by LO through L3 or L4 through L7. For example, L0-3=0111 can specify general register 7. Note, however, that while this address corresponds to the hexadecimal address of general register 7, that register's address in local storage is determined by C, P, and Lregister bits when the L-register is used in the addressing.

The address of a floating-point register is usually specified by L0-3 only (not L4-7).

The L-register may be used as a working register.

SDBO PRE-ASM, ASM

- The Storage Data Bus-Out (SDBO) preassembler receives a doubleword of data from internal or external storage.
- The output of the SDBO preassembler is gated by M3 bits 5, 6, and 7 to provide word, halfword, or byte selection.
- The SBDO assembler receives inputs from the SDBO preassembler, the storage-protect stack, and the D-register. The assembler provides an output that is used as data for External Bus-In (EBI) and local storage.

The selection of data fed to the SDBO assembler is accomplished by decoding M3 bits 5, 6, and 7. The decode of the M3 bits causes corresponding gating lines to be activated, which cause data from the SDBO preassembler to be routed to the SDBO assembler.

M3 Re	g	Selects			
bit 5	V3 Reg xit 5 0 E 1 0 pit 6 0 E 1 0 1 0	Even word bytes 0-3			
	1	Odd word bytes 4-7			
bit 6 0	Even halfword, bytes 0,				
	1	Odd halfword, bytes 2, 3			
bit 7	0	Even bytes 0, 2			
	1	Odd bytes 1, 3			

16)-• ; L -----H BUN CHICK H S.A ACU 3 CARRY LCCK ANSAD LOGICAL 0, 1, 2 CONT ADL CPU CLOCK íR N TREASUREMENTALISE CONTROL . Dischoptie Lactica --







To Byte 3 LS/EXT



-

TT

0 1

2

X

1.10

Read Word









> M3 bit 6,7=00, Byte 0 to X M3 bit 6,7=01, Byte 1 to X M3 bit 6,7=10, Byte 2 to X M3 bit 6,7=11, Byte 3 to X

CPU 8 SDBO PRE-ASM, ASM

SDBO PRE-ASSEMBLER UNIT DATA FLOW



3145 TM CPU 9

SDBO Asm

0

1

2

3

A-LS

B-LS

LOCAL STORAGE

- Local Storage (LS) consists of two monolithic stacks of 64 words each (A-LS and B-LS).
- Destined data is written into both A- and B-LS so that both stacks contain the same information at any corresponding address. This permits comparison checking of LS data.
- LS is used by the microprogram as a high-speed buffer. Access time is 24 nanoseconds.
- Readout is nondestructive.
- Address range within each stack is 00 to 3F (Hex).
- Addressing is accomplished with combinations of control-word bits, P-register bits, L-register bits, T-register bits, selector-channel share-cycle forced bits, and console-file command-register bits.

LS has assigned locations for specified functions. Refer to "Local Storage Map (370 Microprogram in Control Storage)." Locations included are:

- 16 general registers
- 4 floating-point registers
- Selector-channel work area
- CPU work area

These locations are valid when the 370 microprogram is located in control storage. When diagnostics are running, another set of LS assignments is in effect.

LS is external to main and control storage. Each 64-word stack is located on two MST cards:

	Bytes	Card Location
A-LS	0 and 1	A-84P2
	2 and 3	A-B4M2
B-LS	0 and 1	A-C4B2
	2 and 3	A-C4C2

Note: Do not remove or replace LS array cards with power on.

LOCAL STORAGE OPERATION

Read

- Either or both A-LS and B-LS can be accessed in one cycle.
- Data from A-LS is gated to the A-register.
- Data from B-LS is gated to the B-register.
- A-LS and B-LS sources can be different addresses.

Write/Read

- Data destined to LS always is written into both A- and B-LS.
- Data destined during any cycle is written during the next cycle.
- A write LS is always followed by a read LS. The read LS data is used for flush-through checking and A- and B-LS comparing.

DATA CHECKING

Flush-Through Check (FTC)

Data destined to local storage as a result of some control-word operation, other than a storage word read, is gated from the D-register through the SDBO assembler to local storage. The data in the D-register is compared with the data from the A-LS address that was the destination. If the compare is not equal, bit 2 of MCKA 1 sets to indicate an FTC error.

A and B Local Storage Compare

Data destined to local storage is stored at corresponding addresses in both local-storage stacks. The data is then read from these addresses and compared. If the compare is not equal, bit 1 of MCKA 1 sets to indicate the error.



LOCAL STORAGE TIMING

L	0-Time	1 1-1	ime
	0-Time	Delayed	1-Time
		Del	aved

Read	
Write/Read	
FTC	
A/B-LS Compare	

Write/Read, FTC, and A/B-LS compare occur during the cycle following the control-word cycle that the data was destined.





LOCAL STORAGE DESENATION ADDRESSING

The type/form of the control word selects the source address (A or B) that is used for the destination address.

- 1. During fast gate, decoded source addresses are stored in the A and B buffers.
- 2. At the beginning of slow gate, the previous control-word destination address is gated from the A and B destination address latches to the address decoders.
- 3. The data destined during the previous cycle is written.
- 4. The buffer (A or B) selected by the word type/form is gated to both the A and B destination address latches.
- 5. At slow gate of the next cycle, this address is gated to the address decoders for destination write/read.



A new source address may be the same local-storage address as the previous destination address. When this condition occurs, the destined data in the Z-register is not stored into local storage in time to be accessed by the following control word as source data. Destination look ahead detects this condition by comparing A and B new source addresses with the previous destination address. An unequal compare (previous destination *not* new source) gates the new source data from local storage to the A- or B-register. An equal compare (previous destination *is* new source) gates the new source data directly from the Z-register to the A- or B-register. This compare is done on a byte basis.





LOCAL STORAGE

CPU 12

A-LOCAL STORAGE UNIT DATA FLOW

CIRCUIT	CARD
FAST DECODE	C4F2.C4N2.C4M2
DEST BFR LATCHES 2.3.4	C4G4
DEST BFR LATCHES 5.6.7	C4J4
DEST ADDR LATCHES 2.3.4	C4G4
DEST ADDR LATCHES 5.6.7	C4J4
SLOW X ASSEMBLER	C4G4
SLOW Y ASSEMBLER	C4J4
COMPARE	C4N2,C4L2
X DECODE	C4F2
Y DECODE	C4M2
X CHECK	C4G4
Y CHECK	C4J4
MONO BUFFER	84P2.84M2

LA011-LA018,LA021,LA111-LA117 LA212 LA222 LA212 LA222 LA211 LA221 LA022-LA024, LA031-LA032 LA015-LA016 LA114 LA212 LA222 LA311,LA327

ALD







A LOCAL STORAGE CONTROL ASSEMBLER

A control word read from control storage (on the SDBO), or assembled in the secondary control assembler, is gated to the local-storage control assembler. In the local-storage control assembler, the word type is identified to specify the type of addressing needed.

B DIAGNOSTIC FUNCTION:

With DIAG0 bit 5 on, and RTY backup Asm byte 0 bit 5 off, an extra X-line is forced up to cause an X-check. With DIAGO bit 5 on, and RTY backup Asm byte 0 bit 6 off, an extra Y-line is forced up to cause a Y-check.

C DIAGNOSTIC FUNCTION:

With DIAG0 bit 4 or 5 on, and RTY backup Asm byte 3 bit 6 off, the storing of data into A local storage is blocked. This causes both a flush-through check and an A- and B-local store compare error.

D X- and Y-CHECKS

1-Time

Gate Slow Path

D

Write-Read Previous Destination

1-Time Dly

Dest Addr Latches

0-Time Dly

0-Time

Read Source

Dest Bfr Latches

The X- and Y-checks are tests to determine whether if at least one, but not more than one, X- and Y-line is active at a time,

A-LOCAL STORAGE ADDRESS ASSEMBLY



LOCAL STORAGE

CPU 14

B-LOCAL STORAGE UNIT DATA FLOW

CIRCUIT	CARD	ALD
FAST DECODE	C4F2,C4N2,C4H2	LA011-LA018,LA021,LA121-LA127
DEST BFR LATCHES2,3,4	C4G2	LA232
DEST BFR LATCHES 5,6,7	C4J2	LA242
DEST ADDR LATCHES 2,3,4	C4G2	LA232
DEST ADDR LATCHES 5.6.7	C4J2	LA242
SLOW X ASSEMBLER	C4G2	LA231
SLOW Y ASSEMBLER	C4J2	LA241
COMPARE	C4N2,C4L2	LA022-LA024,LA031-LA032
X-DECODE	C4F2	LA017-LA018
Y-DECODE	C4H2	LA124
X-CHECK	C4G2	LA232
Y-CHECK	C4J2	LA242



С



A LOCAL STORAGE CONTROL ASSEMBLER

A control word read from control storage on the SDBO or assembled in the secondary control assembler is gated to the local-storage control assembler. In the local-storage control assembler, the fields that indicate the source accessing are tested to determine the type of address formation needed to address local storage.

B DIAGNOSTIC FUNCTION

With DIAG0 bit 5 on, and RTY backup Asm byte 0 bit 5 off, an extra X-line is forced up to cause an X-check. With DIAG0 bit 5 on, and RTY backup Asm byte 0 bit 6 off, an extra Y-line is forced up to cause a Y-check.

C X- and Y- CHECKS

The X- and Y-checks are tests to determine whether at least one but not more than one X- and Y-line is active at a time.

B-LOCAL STORAGE ADDRESS ASSEMBLY



LOCAL STORAGE CPU 16

LOCAL STORAGE MAP (370 MICROPROGRAM IN CONTROL STORAGE)

							P low	P high
Word	LS	Byte O	Byte 1	Byte 2	Byte 3	X and Y	Direct	Indira
Name	Location					Line	Access	Acces
	00		General Reg	ster O		X0 Y0		
	01		General Reg	ister 1		X0 Y1		
	02	· 1	General Reg	ister 2		X0 Y2		
	03		General Reg	ister 3		X0 Y3	ò	
	04		General Reg	ister 4		X0 Y4		
	05		General Reg	ister 5		X0 Y5		
	06		General Reg	ister 6		X0 Y6		1.1
-	07	1	General Reg	ister 7		X0 Y7		
	08		General Reg	ister 8		X1 Y0		ò
	09	1	General Reg	ister 9	-	X1 Y1	1.1	
	0A		General Reg	ister A		X1 Y2		
	08	-	General Reg	ister B		X1 Y3		
	0C		General Reg	ister C		X1 Y4		-
	OD		General Reg	ister D		X1 Y5		
	OE		General Reg	ister E		X1 Y6		
	OF		General Reg	ister F		X1 Y7		
AX	10		SRTN Temp	Link		X2 Y0		
DI	11		Alter/Displa	y Log Link		X2 Y1	i i	
RTX	12		Retry Link			X2 Y2		
DTX	13		Translate Li	nk		X2 Y3	2	
X	14		Working			X2 Y4	- 1	- 1 - E
R	15		Working			X2 Y5		1.1.1
Y .	16		Working			X2 Y6		
Q.	17		Working			X2 Y7		1.1
MA	18		Working			X3 Y0	-	2
MBS	19		Working	1		X3 Y1		
MX	1A .		Working			X3 Y2		
MC	1B		Working			X3 Y3	3	
MD	10		Working			X3 Y4	. (
MF	1D		Working			X3 Y5		1
MW	1E		Working			X3 Y6		
cx	1F	CPU	Link	Register	1	X3 Y7		

						1				PION	P high
		Word	IFA	LS	Byte 0	Byte 1	Byte 2	Byte 3	X and Y	Direct	Indirect
		Name	Name	Location	1	1	1		Line	Access	Access
	r	GD		20					X4 Y0		
SX 2	$\left \right $	GC		21			Cou	int	X4 Y1		
		GM		22	1	Protect CC	W Address		X4 Y2		
	ι	GW	1	23			1.		X4 Y3	4	
	٢	GD		24	1	1		1 N N	X4 Y4		
SX 3	4	GC	1	25			Cou	int	X4 Y5		
		GM		26	1	Protect CO	CW Address		X4 Y6	1 1	
	Ľ	GW		27		1			X4 Y7		
	r	GD	FD	28				1	X5 Y0		
SX 1	4	GC	FC	29			Cou	int	X5 Y1		
		GM	FM	2A		Protect CO	CW Address		X5 Y2		
	ι	GW	FW	28					X5 Y3	5	
	r	GD	FA	2C					X5 Y4]	1
SX 4	-	GC	FB	2D			Cou	int	X5 Y5		
		GM	FS	2E		Protect CO	CW Address	T	X5 Y6	1 1	
	ι	GW	FL	2F					X5 Y7	1	•
				30		Floating-P	oint Register	D	X6. Y0		
				31	1	Floating-P	oint Register	0	X6 Y1	1	
•			1	32		Floating-P	oint Register	2	X6 Y2	1	
				33		Floating-P	oint Register	2	X6 Y3	6	
				34		Floating-P	oint Register	4	X6 Y4	1 1	
				35		Floating-P	oint Register	4	X6 Y5	1	
				36		Floating-P	oint Register	6	X6 Y6	1	
			1	37		Floating-P	oint Register	6	X6 Y7	1	
		so		38					X7 Y0		
		РМ		39	PE Contro	ol PE Code Gr	oup After Mas	ik .	X7 Y1] [
		DM	1	3A		Adjustme	nt Factor		X7 Y2	1	
		RW		38		Address A	djustment Wo	rking	X7 Y3	1 7	
		DP		30		IF A Low	Priority Link		X7 Y4	1 1	
		LNK		3D		I-Cycle Li	nk		X7 Y5	1	
		P4X		3E		SX-4 Link	Register	1.1.1.1.1.1	X7 Y6	1	
		P3X		3F		SX-1, 2, 3	Link Register		X7 Y7	1	
					-						

Note: Words 28 through 2F are shown with selector channel designations.

MPX Channel

Current	PSW

Displayed in:

BC																•				
Syster	m Mask		KEY		OMWP	Interr	upt Co	de	ILC		CC		1	rogram	Mask		Γ	Instruction Address		
0		7	8	11	12 15	16		31	32	33	34	35	36	37	38	39	40	la de la composition	63	
E	ternal 10		EXP LS 50	1	EXP LS 53	1				EXPL	\$ 53		1		EXP LS 53		Ī	EXP LS 50	1.1	
1			Byte O	ĺ	Byte 1	!			Byte	0	Byte O		ł		Byte 0		ł			
NOTE	The PSW can	bern	anually displa	yediu	sing	1			Bit 0	1	Bit 2, 3		Í.		Bits 4-7		i –	Bytes 1, 2, 3		
1	the Console f	PR-KE	. This proced	ure is		1				- ^ *							!		19 J. A.	
1	contained in	CPK.				1			- ⁻ -		-						i.			
E	,	. 1			4	i											!			1
IEC	<u>.</u>								<u> </u>								1			
Syste	m Mask		KEY	-	1MWP	00	cc	Prog	000	000000		0000	0000		. Inst	tructio	n Ado	iress	1	
0		7	8	11	12 15	16	118	20 23	24		31 32				40				63	

SCOPE PROCEDURE FOR LOCAL STORAGE

5.1

+0-Time

Y5

X7

Use Tektronix* Type 454, or equivalent 10X Probes Set Time/Div: .05 us Set Channel 1 Volt/Div: 50 MV Set Channel 2 Volt/Div: 50 MV

Store 385E6FC8 (using system console rotary switches A through H) in an unused location of control storage. This is a word-move word: Y = LNK, SF, STOP

Set the P-register = 02

The word-move word is the version 1 type that carries the Xand Y-lines of the Source register in byte 1 of the control word. The mask of F specifies that all four bytes of the source are to be moved to the destination register. The STOP function is active (bit 4 of byte 3 = 1); therefore, the word-move word is continually executed.

- After storing the word and setting the P-register,
- 1. Dial the address of the word-move word into switches E through H.
- 2. Operate the control address set key.
- 3. Operate the start key.

Note: The manual indicator is on because the soft-stop condition is set by the STOP function of the word-move word.

4. Sync with channel 1 on +0-time (gate A C4 E2 G05). With channel 2, scope the + A-LS ADDR Y EQUALS 5 line (C4 M2 J12).



For example:

Original word-385E6FC8 Statement-Y = LNK, SF, STOP Change word to-385E7FC8 Statement-Q = LNK, SF, STOP

Change P-register setting

to 03; leave the original

control word in control

storage. Statement-MN=LNK, SF, STOP

For the P-register settings necessary for addressing local storage, refer to the local-storage map in the CPU Hardware (CPU) section.

For variations of this word, refer to the bit definition of the word-move word in the Microprogram (MIC) section.









With channel 2, scope the + SLOW AY PATH DECODE 6(C4 J4 D02) !



With channel 2, scope the + A-LS ADDR X EQUALS 7 line (C4 F2 G12).

Scope pictures 11 and 22 show the X- and Y- lines for the source register LNK being activated early in the cycle of the word-move word.



Scope pictures and an show the X- and Y-lines for the destination Y-register being activated in the second half of the word-move word cycle. Destination addressing is always activated for the destining of the previous control word's results.



EXPANDED LOCAL STORAGE (EXPLS)

- Composed of hardware registers that are physically externals but are logically connected as local storage.
- Source addresses are formed through the expanded localstorage address assembler.
- Destination addresses are formed through the localstorage-control assembler and the A-local-storage-address assembler.
- Expanded local-storage registers are not duplicated as are the local-storage registers.
- Used with I-cycles, selector channels, and address-adjustment circuits.
- In order to access expanded local storage :
- 1. Mode register bit 1 must be on.
- 2. Direct local-storage addressing must be specified (C1 or C2 bit 0 = 0).

P-register bits 0, 3, and 4 control the expanded localstorage inputs to the A- and B-registers (shown on page CPU 20).

Only one expanded local-storage register can be accessed as a source in any one control word.

- The branch and link/return word and the word-move word version 1 cannot access expanded local-storage registers. All expanded local-storage registers may be displayed from the console, but altered only from the console printer-keyboard.
- The I, V, U, and W-registers are not under control of bits 0 and 4 of the P-register, but do require that P low = 2 or A.

If different expanded local-storage addresses are called for with A and B inputs, the register designated as the A source is used for both A and B sources.





EXPANDED A-LOCAL STORAGE INPUT

I, V, U, and W are not dependent on the setting of PO or P4.





EXPANDED B-LOCAL STORAGE INPUT

LOCAL STORAGE GATING

EXPANDED LOCAL STORAGE (EXPLS)

CPU 20
EXPANDED LOCAL STORAGE MAP

Note: Expanded local storage may be altered from the printer-keyboard with the CE key on.

						X and Y
EXPLS	Word Name	Byte O	Byte 1	Byte 2	Byte 3	Line
50	1	Key		1-Register		X2 Y0
51	v			V-Register		X2 Y1
52	Ŵ			W-Register		X2 Y2
53	U			U-Register		X2 Y3
54	IBU			IBU-Regist	ter	X2 Y4
55	TR			TR-Registe	er	X2 Y5
56	ICS	I-Cycle	Control Disp	olay		X2 Y6
		57 throug	h 5F unassig	ned		
60	G2DRL	T	DA	TA ADDR ISX	2)	X4 Y0
61	G2DBRL	1	BA	CKUP DATA A	DDR	X4 Y1
62				1		X4 Y2
63				1		X4 Y3
64	G3DRL		DA	TA ADDR ISX	3)	X4 Y4
65	G3DBRL		BA	CKUP DATA A	DDR	X4 Y5
66				1		X4 Y6
67				1		X4 Y7
68	GIDRL		DA	TA ADDR (SX	1)	X5 Y0
69	GIUBRL		BA	CKUP DATA A	DDR	X5 Y1
6A	1					X5 Y2
68						X5 Y3
6C	G4DRL		DA	TA ADDR ISX	4)	X5 Y4
6D	G4DBRL		BA	CKUP DATA A	DDR	X5 Y5
êE						X5 Y6
6F						X5 Y7
		70 throug	h 77 unassig	ned		
73	SN		SEGMEN	T NUMBER GA	TE	X7 Y0
79	PN.		PAGEN	UMBER GATE		X7 Y1
7A	WK		Working	Register		X7 Y2
78	NP	÷	ŧ	CONTROL	CONTROL	X7 Y3
7C	DK	Local Add	dr. Reg.	TLB DESTIN	ATION GATE	X7 Y4
7D	SS	T		SEG SIZE		X7 Y5
7E						X7 Y6
7F						X7Y7

SEGNENT AND FAGE NO. FROM

I-Register (EXPLS 50)

When the instruction counter register is used as a destination, byte 0, 24 bits, or fullword-loading is possible.

Byte 0: If the I-Reg is destined, byte 0 is gated to the Key-Reg Bytes 1,2, and 3 contain the instruction address

V-Register (EXPLS 51)*

Bytes 1, 2, and 3 usually contain the second operand address generated during I-cycles.

W-Register (EXPLS 52)*

Bytes 1, 2, and 3 usually contain the first operand address generated during I-cycles.

 If used in a storage word, the key register is gated as byte 0. The key register contains the storage protect key (bits 0-3, bits 4-7=0).

If V or W is used as a storage address in a storage word, the KEY reg is gated as byte 0. The KEY reg is always gated as byte 0 for I, TR, and IBU.

U-Register (EXPLS 53)

When used as a destination, byte 2 may not be changed (loaded via hardware)

- Byte 0 bits 0-1 used for ILC bits 2-3 used for CC bits 4-7 used for program mask
- Byte 1 bits 0-1 reserved for FLP mult. and divide bit 2 Indicates that GRs 0-3 need restoring. bit 3 indicates LEX MODE
- bit 4-7 used for OWMP
- Byte 2 used for Op-code Byte 3 used for immediate byte information
- *Note:* Byte 0 bits 0 and 1 and Byte 2 bits 0 through 7 are set only by hardware.

IBU-Register (EXPLS 54) *

Upon entering 1-cycles, 1-Reg bytes 1, 2, and 3 are set into IBU. If a retry condition is encountered during 1-cycles, the instruction may be repeated (return to DFOC). In this event, IBU is moved to the 1-Reg. This source-only register may not be used as a destination, IBU is loaded via hardware only from the 1-register.

TR-Register (EXPLS 55) *

This register is the address of next doubleword after the address in the I-register. TR-Reg may not be used as a destination, but changes if the I-Reg is destined.

ICS (I-Cycle Status) Register (EXPLS 56)

This unique register is provided for manual display only, and is not accessible via microcode. While the specific signals occupy an expanded local-storage register address, no such register exists. Instead, various key signals from the I-cycle hardware (under hardware control) are gated via the register address to form a display. Interpretation of the display requires a fundamental knowledge of the functional operation of the I-cycle hardware.

- Byte 0 forced to zeros Byte 1 forced to zeros
- Byte 2
- bit 0 BR Read Latch
- bit 1 Op Load Latch
- bit 2 Op L2 bit 3 Op L1
- bit 4 Prefetch required
- bit 5 Prefetch inhibit
- bit 6 FLP Long
- bit 7 Op BR to DF
- Byte 3
- bit 0 1-Bfr 0 parity check latch
- bit 1 I-Bfr 1 parity check latch bit 2 Half Adder check latch
- bit 3 IMM byte modifier parity check
- bit 4 X=0
- bit 5 B=0
- bit 6 Set Control Address
- bit 7 Low bit

ICS Bits: Functional Significance

- Byte 2: Bit 0 (BR Read Latch), when on, indicates that a RTN to I-cycles has forced an initial I-cycle address of DF0C.
 - Bit 1 (Op Load Latch), when on, indicates a hardware attempt to:
 - a) provide an initial I-cycle address of DF14 (if byte 2 bit 0 is off).
 - b) provide an address of DF14 for a further fetch of the instruction when within I-cycles and Set Control Address (Set CA) (byte 3 bit 6) is on.

Op	Lengt	h
1	2	Format
1	0	RR
0	1	RX SI
1	1	SS

- Bit 2 (Op Length 2) from decode of two high-order bits of Op Reg, when on, denotes that the data currently in Op Reg is not an RR format.
- Bit 3 (Op Length 1) from decode of two high-order bits of Op Reg, when on, denotes that the data currently in Op Reg is not of RX, RS, SI format,
- Bit 4 (Prefetch required) when on indicates (as a function of I-Reg, I-Bfrs, and current instruction conditions) that the next instruction should be prefetched.
- Bit 5 (Prefetch inhibit) When on, indicates that a Prefetch will not be allowed. This signal is also a function of I-Reg, I-Bfrs, and current instruction. Note that this signal does not take into account other functions, such as Real Instruction Address Compare mode.
- Bit 6 (Floating Pt Long), when on, represents a partial decode of the data in the Op Reg. This signal is used with RR-instruction format to determine a specific I-cycle path.
- Bit 7 (Op Branch to DF), when on, indicates that the end of the hardware 1-cycles branched to the read and align phase of 1-cycles (the other half of the DF module instead of a CX module).
- Byte 3: Bit 0 (I-Bfr 0 parity check latch), when on, indicates an incorrect parity condition for I-Bfr 0.
 - Bit 1 (I-Bfr 1 parity check latch) when on, indicates an incorrect parity condition for I-Bfr 1.
 - Bit 2 (Half Adder check latch), when on, indicates that a check condition occurred in the half adder during an I-register hardware update,
 - Bit 3 (Immediate Byte Modifier parity check), when on, indicates a parity check of the immediate byte modifier register. This signal is the check latch input.
 - Note: The above four signals are combined to form the signal 'I-cycle hardware check'.
 - Bit 4 (X=0) Has particular significance when the op reg data is for an RX format. When this bit is off (and byte 3 bit 5 is off) double indexing is indicated for RX format instructions. Note that RS, SI, and SS format instructions force this signal on.
 - Bit 5 (B=0), when on, indicates that the data being gated through the base assembler of the I-Bfrs is zero. (refers to GPR00). This signal has no significance RR format instructions.

EXPANDED LOCAL STORAGE (EXPLS)

CPU 22

- Bit 6 (Set CA), when on, indicates either that the current control store address is not within the hardware I-cycle range, or that a branch point within that address range has been encountered.
- Bit 7 (Low Bit) when on, indicates that hardwaredetected conditions are currently satisfied for deviating from the normal branch to the execution: the starting address is C (Op code) 4 instead of C (Op code) 4. This signal has significance for certain branch instructions, floating-point, and shift-double instructions.

Byte 3 bits 0-3 provide a further definition of the cause of an I-cycle hardware check. Byte 2, bits 0-6 and byte 3, bits 4-6 may be used, with discretion, to determine the starting address for I-cycle sequences. Byte 2 bit 7 indicates when the "read and align" phase is used; and Byte 3 bit 7 indicates the status of hardware tests for branch on condition, floating-point reg, specification, etc.

G2DRL, G3DRL, G1DRL, and G4DRL (EXPLS 60, 64, 68, and 6C)

These registers function as a pointer to the next storage location used for a share cycle. BYTE 0 contains the protect key.

G2DBRL, G3DBRL, G1DBRL, and G4DBRL (EXPLS 61, 65, 69, and 6D) Only bytes 1 and 2 exist.

SN-Register (EXPLS 78)

Byte O FF hex Byte 1 00 hex

PN-Register (EXPLS 79)

Byte 0 FF hex Byte 1 00 hex 4

WK-Register (EXPLS 7A)

Working Register Byte 0 = FF

NP-Registe	er (EXPLS 7B)	
Bytes 0 and	1 Logical Address	
Byte 2	-	7
Bit 0		1
Bit 1		
Bit 2		1.1
Bit 3		
Bit 4		1
Bit 5	Lex Mode	1
Bit 6		
Bit 7	Reset Tables	5
Byte 3		
Bit 0		
Bit 1		1.
Bit 2		
Bit 3		
Bit 4	Execute Instruction	
Bit 5		4
Bit 6		1 .

Bit 7

)	
1	
1 .	
See "I and N	Dynamic Address Translation NP2 PS Register," page CPU 155.

DK-Regist	ter (EXI	PLS 7	C)
Byte 0	Bit 0)	
	Bit 1	1	
	Bit 2		
	Bit 3		
	Bit 4	1.4	
	Bit 5	1.0	
	Bit 6		
	Bit 7	2	Logical Address
Byte 1	Bit O		
	Bit 1		
	Bit 2		
	Bit 3		
	Bit 4)	
	Bit 5	0	
	Bit 6	0	
	Bit 7	0	
Byte 2	Bit O)	
	Bit 1		
	Bit 2	1	
	Bit 3	1.	
	Bit 4		
	Bit 5		
	Bit 6		
	Bit 7	2	Real Address
Byte 3	Bit O	1 .	
	Bit 1		
	Bit 2		
	Bit 3	1	
	Bit 4	J	
	Bit 5		
	Bit 6		
	Dia 7		

EXPANDED LOCAL STURAGE: SOURCE GATING

When used as a source, expanded local storage is addressed by the expanded local-storage address assembler. The address bits from the control word are intercepted as the control word is being read from control storage, and gated to the expanded local-storage address assembler. Gating lines generated by the address assembler gate the proper expanded local-storage register to either the A- or B-register.

If expanded local storage is being gated to the A-register, the sense latches for A-local storage are blocked from being set. If expanded local storage is being gated to the B-register, the sense latches for B-local storage are blocked.

If the expanded local-storage register is also the destination, the destination latches in local storage are set to be used in the following control-word cycle.

Examples on the following pages show the various ways the expanded local-storage gates may be formed for source addressing.

Whether the expanded local-storage register is an A or a B source, it is gated to both the A-LS and B-LS assemblers. The source gating circuits then gate the A or B assembler to the A- or B register.

There are control words in the microprogram listings that appear to be addressing two different expanded local-storage registers as sources in the same word, However, the B-source address defaults to the A-source. For example:

Control Word-----WK1 = NP2, OE, WK1

NP2 is the A-source WK1 is the B-source

The decode of this word defaults to read effectively

WK1 = NP2, OE, NP20

This type of control word is valid only if the B-source is being blocked from ALU entry.



EXPANDED LOCAL STORAGE (EXPLS)



CPU 24



Word-Move Version 0

P Low=2 (X2 decode)

Statement C0 Bits 0-4=00110 C2 Bits 0-3=0010 Q=W,D7

In this word-move example, the expanded local-storage W-register is the B-source. W is one of the four expanded LS-registers that do not rely on PO or P4 to bring up a gating line. The upper AND circuit is activated by this control word. The line 'Gate B Exp LS' gates the W-register (from the B-Asm) to the B-local-storage bus-out. This gating line also blocks the B-localstorage sense latch set.



Arithmetic Word (A destination) P Low=2 (X2 decode) Statement C0 Bits 0.4=11000 C1 Bits 0.3=0001 V0=0

The decode of this arithmetic word specifies that the expanded local-storage V-register is to be accessed as the A-source and is also to be destination of the arithmetic result.

The 'Gate A Exp LS' line is activated through the AND circuit highlighted in the diagram. The X and Y decode for the V-register is also set up in the A-local storage destination latches for use in the following control word cycle.

EXPANDED LOCAL STORAGE (EXPLS) CPU 26



Storage Word (Read Word)

P Low = F (X7 decode)

Statement	CO Bits 0-4=01000
	C2 Bits 0-3=0010
RDW RW WK, NOP	

The expanded local-storage WK-register is the address source for this Read Word. The highlighted AND circuit brings up the 'Gate B Expls' line. The set to the B-local-storage sense latches is blocked to prevent any conflict on the B-register input.



Arithmetic Word (A destination)

P=87

Statement	CO Bits 0-4=10001
	C1 Bits 0-3=0011

NP2=NP2, OR, K05

The expanded local-storage NP2 register is the A-source and destination in this arithmetic word. The 'Gate A Exp LS' line is brought up by the highlighted AND circuit. The X- and Y-decode for NP is set up in the A-local storage destination latches for use in the following control word cycle.

EXPANDED LOCAL RAGE: DESTINING

When expanded local storage is addressed as a destination, the destination address latches of A-local storage are used to retain the address until the following cycle.

\$*.

At destination time, the bit gates for both A- and B-local storage are blocked to prevent local storage from being set. The data destined to expanded local storage is gated from the D-register through the SDBO assembler on the EBI to the expanded local-storage register addressed by the Alocal storage destination latches.







- External facilities are composed of registers, buses, status lines, and other circuitry that form the communications line between the microprogram and:
 - Channels Console File Console Printer-Keyboard Checking facilities Retry circuits
 - Integrated File Adapter Features
- Addresses are formed from control words, console-file data, selector-channel circuits, console switches, retry information, and local-storage address data.
- Data from the externals enters the data flow through the external assembler to the A-Reg only. Externals cannot be gated to the B-Reg.
- Data to the externals is gated through the SDBO assembler on the External Bus-In (EBI).
- External facilities have restrictions associated with them because of the manner in which they are used. For example, certain externals cannot be addressed as destinations for data, others cannot be sources of data.

А

B

С

EXTERNAL CONTROL ASSEMBLER

- Receives data from the SDBO early in the CPU cycle to form the source gates necessary to gate-in external data to the data flow in time to be used in source-controlword operation.
- Receives data from the secondary control assembler to form addresses in conjunction with source selector channel, console file, or display operation.

X-Y DECODES

The X- and Y-lines are brought up only for destination addresses. The X- and Y-combinations are routed to the various external hardware locations to bring up set and gating lines,

The X- and Y-lines are checked to assure that one and only one X-line, and one and only one Y-line is activated for a destination address. An X-compare check sets MCKA3 bit 0, A Y-compare check sets MCKA3 bit 1.

SOURCE ADDRESSING

7

Source addressing is performed by generating gating lines that allow selected external buses to feed through the external or expanded external assembler into the CPU data flow.

FLUSH-THROUGH CHECK D

Data destined to some external facility is gated from the D-register through the SD

Bus In to the selected external facility. The data from the D-register is gated to the flush-through-check match circuit where it is compared with the data from the external that

was the destination. If the data does not compare, bit 2 of MCKA1 is set to indicate the error. Data gated from the console file to the CFDR is not flush-through-checked.





Note: Refer to "External Assembler" diagram in the 3145 Processing Unit Maintenance Diagrams, SY24-3580.

EXPANDED EXTERNAL ASSEMBLER DATA FLOW



Note: Refer to "Expanded External Assembler" diagram in the 3145 Processing Unit Maintenance Diagrams, SY24-3580.

EXTERNAL ASSIGNMENT AND INDEX MAP

The one SPTL word and the one SYS word appear in every eightword group. These two registers have direct-type addressing and are accessible with any P-register setting. SPTL is addressed when the hex digit C is specified in either the A-source or the B-source fields of a control word. SYS is addressed when the hex digit D is specified in the A-source field.

Bit MCKA and MCKB are set to zero when MCKA is used as a destination in a word-move word, with the NOREG as the source.

WORD	WORD	BYTE	BYTE	BYTE	BYTE	XY
ADDRESS	NAME	0	1 .	2	3	LINE
00	RTY	MB	MB 3	ECNT	RCNT	00
01	NOREG	NOREGO	NOREG:	NOREG2	NOREG3	0.1
02	DIAG	DIAGO	DIAG1	FEAT 2	FEAT 3	02
03	XXXXXXXX	XXXXXXXX	XXXXXXX	XXXXXXX	XXXXXXXX	03
04	SPTL .	S-REG	P-REG	T-REG	L-REG	04
05	SYS .	SYS0	SYS1	SYS2	H-REG	05
06	МСКВ .	МСКВО	MCKB1	MCKB2	MCKB3	06
07	MCKA	MCKAO	MCKA1	MCKA2	MCKA3	07
08	CPU	MODE	CFDAR	LRUM	MATCH	10
09	CFDR	CFDR	CFDR	CFDR	CFDR	11
0A	ACB	ACB0	ACB1	XXXXXX	XXXXXXXX	12
OB	SW	SWO	SW1	SW2	SW3	13
0C	SPTL .	S-REG	P-REG	T-REG	L-REG	14
0D	SYS .	SYS0	SYS1	SYS2	H-REG	15
OE	MPX	MTO	MT1	MB1	MBO	16
OF	DOC	TI	TA	TΓ	TE	17
10	PSWCTL			MSKA	MSKB	20
11	CTCAX	CTCAX0	CTCAX1	CTCAX2	CTCAX3	21
12	MISC	EXTINT		EC LEVEL	SER 1	22
13	CTCAY	CTCAYO	CTCAY1	CTCAY2	CTCAY3	23
14	SPTL .	S-REG	P-REG	T-REG	L-REG	24
15	SYS *	SYS0	SYS1	SYS2	H-REG	25
16	IN	INTA	INTB	SER2	SER3	26
17	DC	DCB0	DCH1	TRBO	DCB1	27
18	ABRTY	ABRTYO	ABRTY1	ABRTY2	ABRTY3	30
19	SPTLB	SRTY	PRTY	TRTY	LRTY	31
1A	HMRTY		HRTY	MRTY2	MRTY3	32
1B	CPURTY	BYDST	RTYFLG	LSDST	EXTDST	33
10	SPTL .	S-REG	P-REG	T-REG	L-REG	34
1D	SYS .	SYS0	SYS1	SYS2	H-REG	35
1E					Nggan panananan	36
1F		1.5				37

Described In Address REC 00 01 CPU 02 DIAG (Bytes 0, 1) CPU (Bytes 2, 3) 03 ---04 CPU REC (Bytes 0, 1, 2) CPU (Byte 3) 05 06 REC 07 REC 80 CPU 09 CFA 0A CPU CPU 08 00 CPU REC (Bytes 0, 1, 2) CPU (Byte 3) 0D 0E CHNL OF CPK 10 CPU 11 FEAT 12 FEAT (Byte 0)CPU (Bytes 2, 3) 13 FEAT 14 CPU REC (Bytes 0, 1, 2) CPU (Byte 3) 15 CPU 16 17 FEAT 18 **REC and DIAG** 19 REC REC 1A 18 REC 10 CPU 1D REC (Bytes 0, 1, 2) CPU (Byte 3)

18

1F

daže spanskov z

EXTERNAL FACILITIES

CPU 32

May not be used as a destination

*Not Flush-Through Checked.

Both MCKA and MCKB are set to zero when MCKA is used as a destination in a word-move word, with the NOREG as the source.

WORD	WORD	BYTE	BYTE	BYTE	BYTE	XY		Address	Describe
ADDRESS	NAME	0	1	2	3	LINE		20	CHNL o
in the second	See See See See		ad an Arris	e Althé				21	CHNL o
20	GBUF FBAK	GBO FWB	GB1 FCH	GB2 FCL	GB3 FOP	40	D	22	CHNLO
21	GBS FCND	GSP FDS	GBF FHC	GCT FED	GBD FMOD	4 1		23	CHNLO
22	GSTAT FSTAT	GF FFL	GE FSC	GS FST	GLFGL	4 2] / "	24	CPU
23	GTAG FTAG	GTO FTO	GT1 FT1	GO FBO	GR FDR	4 3) ,	25	REC (B)
24	SPTL .	S-REG	P-REG	TREG	L-REG	4 4]	26	IFA
25	SYS *	SYS0	SYS1	SYS2	H-REG	4 5		27	-
26	FERR	FSB	FGT	FTS	FAT	4 6	IFA	28	CHNL o
27						4 7		29	CHNL
28	GBUF FRR	GBO FRRA	GB1 FRRC	GB2 FSC	GB3 FSR	50	\mathbf{h}	2A	CHNL
29	GBS	GSP	GBF	GCT	GBD ·	5.1	11	28	CHNL
2A	GSTAT	GF	GE	GS	GL	52	> 5X4/IFA	2C	CPU
2B	GTAG	GTO	GT1	GO	GR	53	1)	2D	REC (B)
2C	SPTL .	S-REG	P-REG	T-REG	L-REG	54		2E	
2D	SYS .	SYS0	SYS1	SYS2	H-REG	55		2F	-
2E	ADJT	LOGICA	LADDR	REAL	ADDR	56		30	CHNL
2F					1	57	1	31	CHNL
30	GBUF	GBO	GB1	GB2	GB3	60	1	32	CHNL
31	GBS	GSP	GBF	GCT	GBD	6 1	1 (33	CHNL
32	GSTAT	GF	GE	GS	GL	6 2	5x2	34	CPU
33	GTAG	GTO	GTI	GO	GR	63	1)	35	REC (B)
34	SPTL *	SREG	P-REG	TREG	L-REG	6.4	1	36	FEAT
35	SYS .	SYSO	SYS1	SYS2	H-REG	6 5	1	37	
36	TODH	TODHO	TODH1	TODH2	TODH3	6 6	1	38	CHNL
37		1				67	1	39	CHNL
38	GBUE	GBO	GB1	GB2	GB3	170	1	3A	CHNL
39	GBS	GSP	GBF	GCT	SX3	71	1(3B	CHNL
34	GSTAT	GF	GE	GS	GL	72	SX3	3C	CPU
38	GTAG	GTO	GT1	GO	GR	7 3	1) Assertion	3D	REC (B
30	SPTL *	SIREG	P-BEG	T-REG	LIREG	74	11	3E	FEAT
3D	SYS .	SYSO	SYS1	SYS2	H-REG	7 5	1	3F	-
3E	TODL	TODLO	TODL1	TODL2	TODL3	7 6	1		
3F		1	1	[1	177	1		

Address	Described In
20	CHNL or IFA
21	CHNL or IFA
22	CHNL or IFA
23	CHNL or IFA
24	CPU
25	REC (Bytes 0, 1, 2) CPU (Byte 3)
26	IFA
27	
28	CHNL or IFA
29	CHNL
2A	CHNL
28	CHNL
2C	CPU
2D	REC (Bytes 0, 1, 2) CPU (Byte 3)
2E	그는 그는 것이 같은 것이 같은 것이야.
2F	그 그 그 그 가지 않는 것 같아. 이 가 같아?
30	CHNL
31	CHNL
32	CHNL
33	CHNL
34	CPU
35	BEC (Bytes 0 1. 2) CPU (Byte 3)
36	FFAT
37	
38	CHNI /
39	CHNI
3A	CHNI
38	CHNL
30	CPU
3D	REC (Butes 0, 1, 2) CPU (Bute 3)
3E	FEAT
3F	-

may not be used as a destination *Not Flush-Through Checked

NOREG Word

This fullword facility is not really a register. It is used to zeroout other locations. For example, if a word-move control word specifies that bytes 1 and 3 of the NOREG are to be moved to a local-storage location, then bytes 1 and 3 of that location are set to all zeros with odd parity.

Diag Word Byte 2 3145 Models FED, GE, GFD, H, HG, and I (Feat 2) Byte 2. Bits 0-3 Main Storage Size 1 = 112k2 = 160k 3 = 208k 4 = 256k 5 = 384k 6 = 512k Bit 4 IFA Bits 5, 6 Channels (Note: IFA counts as one channel) 00 = 1 01 = 2 10 = 3 11 = 4 Bit 7 Word Buffer 3145 Models H2, HG2, I2, IH2, J2, J12, and K2 (Feat 2) Byte 2, Bits 0-3 Main Storage Size 1 = 768k 2 = 1024k3 = 1536k

		5 = 384k	
		6 = 512k	
the Maria States		7 = 2048k	
	Bit 4	Reserved	
	Bits 5, 6	Channels	
		00 = 1	
		01 = 2	
		10 = 3	
		11 = 4	
	Bit 7	Word Buffer	
Diag Word By	te 3 .		
(Feat 3) Byte 3,	Bit O	Model Configuration	
		0 = 3145 Models FED, 0 HG, and I	SFD, H,
		1 = 3145 Models H2, H0 J2, J12, and K2	62, 12, 1H2,
	Bit 1	3215	
	Bit 2	Second 3210	
	Bit 3	3145 Models JI2 and K2	

Spare

Direct Control

Clock Comparator, CPU Timer

Bit 4

Bits 5, 6

4 = 256k

CPU Word

Byte 1 EPSWB Bit

0

5

6

7

Name

Machine-check mask

Wait state

Problem state

0.0 10.0		
	MODE Register	
Byte 0		
Bit O	Hard-stop latch (control register 14 bit 0)	
Bit 1	Enable I-cycle and Adr Adj Ctrl and expanded	
	local storage.	
Bit 2	Enable hardware retry	
Bit 3	Full recording mode for single-bit failures in main	
	storage	
Bit 4	Full recording mode for single-bit failures in	
	control storage	
Bit 5	Threshold mode for single-bit failures in control	
	storage	
Bit 6	Reserved	
Bit 7	Reserved	
Byte 1	CFDAR (Console-file data-address register)	
	Track and sector address used by console file.	
Byte 2	LRUM (Least Recently Used Matrix)	
Bits 0-7	Indicate which adr adj table register was least	
	recently used.	
Byte 3	MATCH	
Bits 0-7	Indicate which adr adj table register matches	
	preaddress assembler, (useful only under diagnostic	
	control).	
SW Word (Console Switches)	
SWO through	SW3 are the rotary console address/data switches:	
SW Byte	Console Switches	
SWO	AB	
SW1	CD	
SW2	EF	
SW3	GH	
•		
PSWCTL V	lord	
Byte 0 EPS	WA Bit Name	
0		
1		
2		
2		
4		
5	Translation mode	
6	I/O master mask	
7	External master mask	
· · · · · ·	External master mast	

Byte 2 MSKA Bit Name Timer mask Interrupt mask External signal mask Reserved Reserved Reserved Reserved Reserved Byte 3 MSKB Bit Name MPX channel mask Selector channel 1 mask Selector channel 2 mask Selector channel 3 mask Selector channel 4 mask Reserved Reserved Reserved

EXTERNAL FACILITIES CPU 34

Byte 1 INTB Bit	Name	
0	Multiplex channel	
1. J. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.	Selector channel 1	
2	Selector channel 2	
3	Selector channel 3	
4	Selector channel 4	
5	I/O interrupt	
6	Timer update	
7*	External	
* External is set on if	all of the following con	ditions exist:
1. An external interru register of the CPU	ption signal is on (that is signal from the SCPU re	s, from the EXTINT gister).
2. The external mask	bit = 1 (bit 7 of the EPS	WA register).
3. For the timer MSK MSKA bit 2 = 1.	A bit 0 = 1 or for extern	als 1 through 6,

Bytes 2 and 3 contain the last four digits of the serial number.

MISC Word Bytes 2 and 3

2. EC Level: External register 12 byte 2 The last two digits in the 370 microprogram EC number are plugged. A test is performed before the go-no-go test to determine whether the disk being loaded is at the proper level.

3. Serial Number

0

1

2

3

4

5

6

7

0

4

2

Ŕ

4

5

6

7

six-digit serial number. These digits are always plugged as: 01 = U.S. manufacture 73 = German manufacture

IN Word (Interrupt Register)

An INTA or INTB (interruption) register bit is set on when the corresponding source has an interruption pending and the system mask is set to allow such an interruption. Bit names in the INT register are:

Byte C	INTA Bit	Name
	0	Spare
	1	Spare
	2	Timer
	3	External signal
	4	System contro
	5	CPU signal 0
	6	CPU signal 1
	7	Process stop

External register 12 byte 3 contains the first two digits of the

82 = Brazilian manufacture

ACB (Address Check and dary) Register

- The ACB-register is a two-byte hardware register that contains. boundary information used to check main- and control-storage accesses.
- The ACB-register is loaded at IMPL and is reloaded each time the system reset routine (GRST) is executed,
- The ACB-register is addressed by the external address 0A and can be used as a source or a destination.

The ACB-register is set at IMPL with a specific value determined by the main-storage and control-storage configuration. Certain feature mixes may require additional control storage, above the 32k bytes that are standard. This expansion of control storage is made at the expense of main storage. The movement of the lower control-storage boundary into the main-storage area is done in 2k-byte increments. The change in the boundary location between main and control storage results in a different setting for the ACB-register.

Once the feature mix and control-storage size is established, the 370 microprogram disk generated at the plant contains the proper ACB setting for that configuration.

For each access of main or control storage, a comparison is made between the ACB-register and the M-register. If a mainstorage access attempts to address the control-storage area, an address check occurs. If a control-storage access is made to a main-storage location, a machine check occurs.

ACB-REGISTER

BYTE 0 BYTE 1 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7

Spares Compared with M1 bits 3-7 on all main-storage accesses. Compared with M2 5=0 bits 0-4 for all storage accesses. (Bits 0 and 1 may be altered for

control-storage

accases.)

M2 5=0 Internal storage only 67=00-16k boundary 67=01-32k boundary internal for a fo

> 5=1 External storage attached 67=00-128k external storage or 1256k external storage 67=01-256k external storage or 1768k external storage 67=10-512k external storage 67=11-758k external storage

Note: The ACB setting for each 370 microprogram load may be found in the module chart in the back of the microlisting. Look up address FF08, the ACB setting is in bytes 0 and 1.

3145 TM CPU 35

	018A	194k	46k	0308	370k	46k	05CC	754k	46k	OBCE	1522k	46k
	0182	192 k	48k	0300	368k	48k	05C4	752k	48k	OBC6	1520k	48k
	017A	190 k	50k	02F8	366k	50 k	05BC	750k	50k	OBBE	1518k	50k
	0172	188 k	52k	02F0	364 k	52k	0584	748k	52k	OBB6	1516k	52k
	016A	186k	54k	02E8	362 k	54k	05AC	746k	54k	OBAE	1514k	54k
	0162	184 k	56k	02E0	360 k	56k	05A4	744k	56k	0BA6	1512k	56k
	015A	182 k	58k	02D8	358k	58k	059C	742k	58k	0B9E	1510k	58k
	0152	180 k	60k	02D0	356k	60k	0594	740k	60k	0896	1508k	60k
	014A	178k	62k	02C8	354 k	62k	058C	738k	62k	OBBE	1506k	62k
	0142	176k	64k	02C0	352k	64k	0584	736k	64k	0B86	1504k	64k
	0281	256 k	32k	0403	512k	32 k	0805	1024k	32k	1007	2048k	32k
	0279	254 k	34k	03FB	510k	34 k	07FD	1022k	34k	OFFF	2046k	34k
	0271	252 k	36k	03F3	508k	36 k	07F5	1020k	36k	OFF7	2044k	36k
	0269	250 k	38k	03EB	506k	38 k	07ED	1018k	38k	OFEF	2042k	38k
	0261	248 k	40k	03E3	504k	40k	07E5	1016k	40k	OFE7	2040k	40k
	0259	246k	42k	03DB	502k	42k	07DD	1014k	42k	0FDF	2038k	42k
	0251	244 k	44k	03D3	500k	44k	07D5	1012k	44k	0FD7	2036k	44k
	0249	242k	46k	03CB	498k	46k	07CD	1010k	46k	OFCF	2034k	46k
	0241	240 k	48k	03C3	496k	48k	07C5	1008k	48k	0FC7	2032 k	48k
	0239	238k	50k	03BB	494 k	50k	07BD	1006k	50k	OFBF	2030k	50k
	0231	236k	52k	0383	492k	52k	0785	1004k	52k	OFB7	2028k	52k
	0229	234 k	54 k	03AB	490k	54k	07AD	1002k	54k	OFAF	2026k	54k
	0221	232k	56k	03A3	488k	56k	07A5	1000k	56k	OFA7	2024k	56k
	0219	230k	58k	039B	486k	58k	079D	998k	58k	OF9F	2022 k	58k
	0211	228k	60k	0393	484k	60k	0795	996k	60k	0F97	2020 k	60k
	0209	226k	62k	038B	482k	62k	078D	994k	62k	OF8F	2018k	62k
	0201	224k	64k	0383	480k	64k	0785	992k	64k	0F87	2016k	64k
÷				18 a								

Main

Storage

384 k

382k

380k

378k

376k

374k

372k

Control

Storage

32k

34k

36k

38k

40k

42k

44k

ACB

0604

05FC

05F4

05EC

05E4

05DC

05D4

Main

Storage

768k

766k

764k

762k

760k

758k

756k

Control

Storage

32k

34k

36k

38k

40k

42k

44k

ACB

0C06

OBFE

0BF6

OBEE

OBE6

OBDE

0BD6

Control

Storage

32k

34k

36k

.38k

40k

42k

44k

ACB

1804

17FC

17F4

17EC

17E4

17DC

17D4

17CC

17C4

17BC

17B4

17AC

17A4

179C

1794

178C

1784

2005

1FFD

1FF5

1FED

1FE5

1FDD

1FD5 1FCD

1FC5

1FBD

1F85

1FAD 1FA5

1F9D

1F95

1F8D

1F85

Main

Storage

1536k

1534k

1532k

1530k

1528k

1526k

1524k

ACB SETTINGS

Control

Storage

32k

34k

36k

38k

40k

42k

44k

46k

48k

50k

52k

54k

56k

58k

60k

62k

64k

32k

34k

36k

38k

40k

42k

44k

46k

48k

50 k

52k

54k

56k

58k

60k

62k

64k

ACB

01C2

01BA

01B2

01AA

01A2

019A

0192

Main

Storage

208k

206k

204 k

202 k

200 k

198k

196k

Control

Storage

32 k

34k

36k

38k

40k

42k

44k

ACB

0340

0338

0330

0328

0320

0318

0310

Main

Storage

112 k

110 k

108 k

106k

104 k

102 k

100 k

98 k

96 k

94 k

92 k

90 k

88 k

86 k

84 k

82 k

80 k

160 k

158k

156k

154 k

152k

150 k

148k

146k

144k

142k

140k

138k

136k

134k

132k

130k

128k

ACB 1

Bits

0 1 2 3 4 5 6 7

M2

Compare

ACB 0

Bits

Invert

Main Storage Size 256k and Above

bit O

0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7

ACB 1

Bits

M2

ACB Compere For Control Storage Access



If the comparison indicates that the ACB value is more than the M2 value, a machine-check condition is specified.

ACB Compare for Main Storage Access



If the comparison indicates that the ACB value is equal to, or less than, the M-register value, an address check occurs.

ACB Byte 1, Bit 0 and 1 Gatage Compare Circuits (Control Storage Only)



Note: Refer to "ACB and M1-Registers" in the 3145 Processing Unit Maintenance Diagrams manual SY24-3580, for high-level diagram of ACB compare.

SYS (System) Register

The system register gives the status or condition of the processor SYS reg is an exterant hardware register located at word address 05,

Byte 0		
Bit 0	Machine-check interruption pending	
1	Retry routine	
2	Machine-check routine	
3	Documentary console 2	
4	Log present	
5	Sub-block protection mode	
6	Selector channel Start I/O latch	
7	Force module 0 to LSCS	
Byte 1		
Bit Q	Address contents	
.1 - 3	CPU interrupt force	
2	SAR interrupt force	
.3	PSW restart	
4		
5	System control interrupt	
6	Timer interrupt force	
7	Reserved	
Byte 2		-
Bit 0	Enable clear switch	
1	IMPL	
2	Load file wait bit	
3	CE key in CE mode	
4	00 System reset - 10 subsystem load (IPL)	
5) 01 Power-on reset - 11 system load (multipro	cess)
6	Error in stop word	
7	Instruction processing latch	14
Byte 3	H-Register	
Bit 0	Machine-check trap	
1	Retry trap	
2	CPU high trap	
3	Integrated file adapter (IFA) if installed	
	Selector channels 1,2,or 3 if no IFA	
4	Selector channels 1,2,or 3 if IFA installed	
	Selector channel 4 if no IFA	
5.	Multiplexer channel	
6	IFA if installed	
7	Store-display	

EXTERNAL FACILITIES CPU 38

PRIORITY OPERATIONS - H-REGISTER

Priority Operations

Priority operations, which may be related to the current operation, can cause delay of the current microprogram routine. Most (not all) of the various priority operations are initiated by traps. A *trap* is basically a circuit-forced branch out of the current microprogram routine to a priority routine. After the priority routine is completed, a return can be made to the interrupted routine so that its execution can continue.

The interrupted routine is delayed further when several priority operations occur at the same time. Or, in some cases, the interrupted routine may be ended by the occurrence of a priority operation. For example, if an instruction address that specifies an unavailable main-storage location is used, an address-check priority operation occurs. The microprogram routine, in which the invalid address is used, is discontinued.

A hierarchy of execution preference exists within the priorityoperation structure. Execution preference is exhibited in two instances:

- Circuit requests for two or more priority operations occur in the same CPU cycle. The highest-priority operation is executed first; the next highest second, etc.
- 2. During execution of a priority operation, a request for a higher-priority operation occurs. The higher-priority operation is executed; the lower-priority operation is delayed until completion of the higher-priority operation, subject to the rules of execution of priority operations specified later in this section.

Selector-channel data-transfer operations have the highest priority but do not use the trapping mechanism. All of the other priority operations use the trapping mechanism, which functions in the following manner.

- In a CPU cycle, a priority operation request is recognized.
 The control-storage address of the first control word in the priority microprogram routine is set into the M-register in what is known as the *trap-1 cycle*, the cycle following the one in which the request is recognized. (The normal next-controlword address, generated by execution of the word in progress, is set into the N-register.) The address set into the M-register is forced by circuitry and depends upon the priority operation for which the request is made.
- The first word of the priority operation is read out of control storage and set into the C-register. Normally, this first word is a branch and link word.
- 4. The branch and link word stores the contents of S, P, N2, and N3 into a link location in local storage. (N2 and N3 contain the address of the word that would have been executed next if the trap had not occurred.) The cycle in which this occurs is called a the *trap-2 cycle*.
- The priority routine is executed. Normally, the last word in the priority routine is a return word. This word loads the link information back into S, P, M2 (N2) and M3 (N3).
- 6. Execution of the interrupted routine is resumed at the control word specified by the link (return) address in M2 and M3.

3145 CPU 39

Trap Operation

TRAP-1 CYCLE	TRAP-2 CYCLE	TRAP-3 CYCLE	TRAP-4 CYCLE
Set By:	Set By:	Set By:	Set By:
I. (Not) Inhibit Traps.	1. Trap-1 INLK latch on.	1. Trap-2 INLK latch on.	1. Trap-3 INLK latch on.
2. Request 0-9	2. 0-45 Time	2. 0-45 Time	2. 0-45 Time
3. O-Time 4. (Not) Stg. 1 cycle			
Purpose:	Purpose:	Purpose:	Purpose:
1. Prevents any additional traps.	 Forces module address to M2 (N2 buffer). 	1. Normal address update to M2	1. Reset Block SPTL (Set at TR 1 cycle).
2. Sets TR 1 INLK	2. Normal M3 (N3	2. TR 3 INLK latch	
latch at 90-135 time.	buffer) addr update.	set at 90-135 time.	
3. Prevents normal set to M-Reg.	*3. BAL operation stores S, P, N2, N3 in link area (values that were set during the trap-1 cycle).	*Trap-3 and 4 cycles looping of Trap-1 and is in the SPTL area.	prevents continuous 2 cycles if an error
4. Forces trap address to M2 & M3.	4. TR 2 INLK latch set at 90-135 time.		
5. N2 and N3 set Normal	*BAL is normally the first word of a trap Routine.		
6. Set Block SPTL.			
7. Execute last word of inter-			



EXTERNAL FACILITIES CPU 40

H-Register

Many priority operations cause an H-register bit to be set on in s . the trap-2 cycle. The priority operations and associated H-register bits are:

Operation Selector share cycles	H-Register Bit	Trap Address	Operation	H-Register Bit	Trap Address
Machine check without I/O	HO	_	Selector Channels 2, 3/4 (with IFA: SX2, 3: without IFA:	H4 SX4	
a. Normal		D000	a. Exceptional status tran		D100
b. H0 is already on		D004	b. Chaining (command or dat	a)	D100
c. One or more machine			a UCW bandling	-,	D109
checks have already			d D ADB trap		D100
occurred (SYSO)					5100
Bit 2 = 1)		D008	Multiplexer channel	H5	D400
d. H0 and SYS0 Bit 2 are			Integrated File Adapter	H6	
already on		DOOC	a. Retùrn Iow		D480
Machine check with I/O	HO		b. Unused		D484
a Normal		D010	c. Unused		D488
b. H0 is already on	• · · · · · · · · · · · · · · · · · · ·	D014	d. Diagnostic		D48C
c. One or more machine			Store/display	H7	_
checks have already			a. Store/display		D840
occurred (SYS0				A1	
Bit 2 = 1)		D018	CPU low without 1/U	None	-
d. H0 and SYSO Bit 2 are			a. Spare		
already on		D01C	b. Storage protect		D004
Botes	U1	19 - 19 - 19 - 19 - 19 - 19 - 19 - 19 -	C. Address check		0806
neary Normal	п	D200	G. ADA ADJ exception		Dauc
b. H1 is already on		D200	CPU low with I/O	None	-
D. This already on	and they then	0204	a. Spare		-
is in progress (SYS0			b. Storage protect		D814
Bit 1 = 1)		0208	c. Address check		D818
d H1 and SYS0 Bit 1 are			d. Spare		-
already on		D20C	Scan/Clear	None	-
			a. Scan storage		D380
CPU High	HZ	-	b. Clear storage		D384
a. Set IC		D300	The following rules apply to ex	ecution of priorit	v operations:
D. CA trap		D304	1. A selector share cycle can bre	ak into any opera	tion except
c. Address contents		D300	the first cycle of a storage con	trol word and du	ring an ECC
a. System reset		0300	retry of a control word.		•
Integrated File Adapter	НЗ	··· -	2. Any trap priority operation ca	an not break into	either:
a. Mini-Op end		D128	a the first cycle of a storage	word operation	NT .
b. Error end		D12C	a. the hast cycle of a storage		
c. Index		D124	b. a trap-2 cycle operation. (i hat is, a trap-2 cy	cle can not
d. Gated Attn or D ADH		0120	be a trap-1 cycle for anoth	er trap.)	
Selector Channels 1, 2, 3	H3		3. If H1 is on, all other priority of	operations (excep	H0-machine
(without IFA)			check-and selector share cycl	es) are prevented.	If, however,
a. Exceptional status trap		D120	a diagnostic trap occurs, it is	executed, even th	ough H1 is on.
b. Chaining (command or data	a)	D124	Also, if the system is in a sing	le-cycle mode of	operation, a
c. UCW handling		- D128	store/display trap can be exec	uted even if H1 is	s on.
d. D ADR trap		D12C	4 IF 112 is an an 112 114 115 -		ha takan 16114
2			······································		JC (dKen, 11 114)

is on, an H4, H5, or H6 trap cannot be taken. If H5 is on, H5 or H6 cannot be taken. If H6 is on, an H6 trap cannot be taken. In any of these cases, the H3, H4, H5, or H6 trap remains pending until after H3 (or H4 or H5) is turned off.

- 5. Ector share cycles can delay execution of other traps for a number of cycles, depending upon the rate at which share cycles occur.
- 6. Discounting the effects of the various non-H-Reg priorities (share cycles, CPU low, scan/clear), the following hierarchy applies.

H-Reg	Blocks Trap Request
Bit	for H-Reg Bit
HO	None
H1	H2,3,4,5,6,7
H2	H2
H3	H3,4,5,6
H4	H4,5,6
H5	H5,6
H6	H6
H7	H7

Share Cycle Priority Operation (Applies to: Selector Channel, Block Multiplexer, IFA).

Priority operation for selector-share cycles is as follows: Without IFA

- 1. The priority sequence is selector channel 1, 2, 3, and 4.
- 2. A share-request for selector channel 4 is taken if no other request is pending.

With IFA

1. The priority sequence is selector channel 2, IFA, and selector channel 3.

Machine-Check Priority Operation (H0)

A machine-check trap occurs (if allowed by the machine-check bit in the PSW) because a series of retry operations has been unsuccessful. The number of retry attempts is determined by a hardware counter. Basically, a machine-check trap occurs either because errors are occurring faster than can be handled or because a hard error cannot be successfully retried.

An attempt is made to form logout information and initiate a machine-check interruption (depending upon the value of the machine check bit in the PSW). The validity of such logout information may be unpredictable if the machine-check trap is called for.

Retry Priority Operation (H1)

The retry routine is entered through the retry priority operation (trap). The retry priority operation occurs when any machine check occurs if the retry counter is not full, retries are not masked off, and system register byte 2 bit 6 (indicates stop word error) is off. Depending upon the nature of the error and the word type, the error may be detected during execution of the failing microprogram word (Type 1), during execution of the following word (Type 2), or may be detectable but uncorrectable (Type 3).

CPU High Priority Operation (H2)

System Reset Microprogram

The system reset microprogram is executed after a circuit system reset has been performed. This action is initiated by operating: 1. The system reset key 2. The load key. System reset causes various CPU registers and controls to be reset.

Integrated File Adapter High Priority Operations (H3)

Four trap addresses are provided for Mini-Op End, for Error End, for Index, and for Gated Attention or D ADR.

Selector Channels or Block-Multiplexer Channels 1, 2 and 3 (H3)

When IFA is not present, four trap addresses are provided for channels 1, 2, and 3: for Exceptional Status Trap, for Chaining (command or data), for UCW Handling, and to protect the next entry of the DADR list.

Selector Channels and Block-Multiplexer Channels 2, 3/4 (H4)

Four trap addresses are provided: for Exceptional Status Trap, for Chaining (command or data), for UCW Handling, and to protect the next entry of the DADR list. When IFA is preset, this trap is shared by channels 2 and 3. For non-IFA, this trap is for the sole use of selector channel 4.

Multiplexer Channel (H5)

This trap is for the sole use of the multiplexer channel for handling data, status, and chaining functions.

Integrated File Adapter Low Priority Operations (H6)

Four trap addresses are provided for Return Low, and for Diagnostics. The other two are not assigned.

Store/Display (H7)

Store/Display pertains to system control panel operations.

CPU Low (No H-Register Bit)

Address Check This trap occurs when an access to an unavailable main-storage area is attempted.

Storage Protection

This trap occurs because of a storage-protection violation.

Address Adjustment Exception This trap is used with DOS emulator.

Scan/Clear (No H-Register Bit) These traps are used for a clear-storage and a scan-storage operation.

EXTERNAL FACILITIES CPU 42





M3 Gating (Traps)



BFR

RM045, RM052

8

N3 BFR Set/Reset

RM113 (45-Time)

N3 Set/Reset

RM113 (135-Time)

REG

RM045, RM052

8

Term for RTY BE243

M3 Bits RM044

OR

RM045,RM052

8

B-Reg Byte 3 🔳

.....

M3 Asm

I-CYCLES

Processing a single software instruction may be divided into two parts: the I (instruction) phase and the E (execution) phase. Instructions are defined to be in different groups according to . their format, length, and general form of execution. The I-phase of processing performs the following basic functions:

Fetch instruction

• Initialize the CPU facilities for the completion of the processing.

.

. . 1

During the E-phase, the CPU performs the unique functions specified by the instruction op-code.

					T					
		First H	alfword			Second H	alfword	I hird Halfword		
		Byte 0	Byte	1	В	yte 2	Byte 3	Byte 4		Byte 5
RR, 00-3F		OP - CODE	R ₁	R ₂						4
DV 4075		00 0005			, 	.	-1			
KX, 40-/F		OP - CODE	<u> ^R1 </u>	×2	B2	Uis	placement ₂			
RS		OP - CODE	R ₁	R ₃	^B 2	Dis	placement ₂			
SI		OP - CODE	l ₂		81	Dis	placement ₁			
							na Star Maria			
SS, CO-FF		OP CODE	L1	L2	⁸ 1	Dis	placement ₁	^B 2	Displacen	^{nent} 2
				Са. (у		The imm	ediate byte is th	e byte follow	ing the op-c	ode
	[OP - CODE]					- 4 - ¹⁴ 16 -		
	-0 -1	2 -3 -4 -5	-6 -7-						•	
	Format	OPERATI	ON							
ILC=1	0 0	0 1 1 0	1 0	ADD	(1A)			la se stand		
ILC=2	0 1	0 1 1 0	1 0	ADD	(5A)					
ILC=2	0 0	0 1 0 1		OR (1	50) (6)	•				
ILC=2	0 1	0 1 0 1	1 0	OR (5	56)	•				
ILC=2	1 0	0 1 0 1	1 0	OR (9	6)					
ILC=3	1 1	0 1 0 1	1 0	OR (C)6)					
Length of Inst	ruction									
in halfword	5						a de la composición d Composición de la composición de la comp			

I-PHASE FUNCTIONS

The initialization of CPU facilities for the E-phase depends partially upon instruction type. All instructions require an updating of the instruction counter, the setting of the specified CPU Regs, and a branch to the start of the execution routine. In addition, some instructions require the fetching of the second operand from a general register, or the calculation of operand addresses.

At this point, some observations may be made about the I-phase functions. For example, the RX and RS/SI functions are very similar. In fact, during the I-phase, an RS/SI instruction is handled exactly the same way as an RX instruction with the X2 field equal to zero. Also, some functions are identical, with only the data value depending upon the format and op code (SPTL, U, and I update).

Note also that the E-phase for some instructions is identical; such as, AR and A, NR and N. The difference between these RR and RX types of instructions occurs only in the source of the second operand (general register or storage). Saving some controlstorage words and time is possible by including the operand fetch as an I-phase function for such RX format op codes.

The I-phase functions may now be illustrated as follows:

I-PHASE FUNCTIONS BY INSTRUCTION





I-CYCLES

CPU 48

Hardware Functions

....

Each software instruction processed requires performing the previously mentioned I-phase functions. Minimizing this time and thereby reducing the time required to process a given instruction is desirable. To minimize the number of machine cycles required during the I-phase (that is, the I-cycles) some functions are performed by hardware. Additionally, some other characteristics of the machine are more fully exploited by hard-ware.

First consider the previously defined 1-phase functions which apply to all instructions. Hardware is used to perform the setting of IBU, SPTL. U, and the 1-Reg update. These functions do not require microwords to be performed; hence, they do not require any additional time during 1-cycles.

Now consider the function Proceed to Execution Routine. To perform a hardware forced branch, define the starting controlstorage address of each execution routine as a function of instruction op code. The hardware branch on the op code does not require any additional time, because no microword is used to perform the branch and module switch.

The interface between storage and the CPU provides a doubleword transfer of eight bytes of data. During a read type of microword, the SDBO assembler provides the selection of the addressed word (halfword, or byte) from the doubleword actually read. The I-cycle hardware provides for buffering the entire doubleword from storage, via a time-slotting of data from SDBO to EBI. When an instruction is fetched from storage, the addressed word is routed from SDBO to the 1-buffer, via EBI, during the normal destination time in storage-2 cycle. During the next cycle time, the odd word is gated to EBI, and placed in the I-buffer. This time-slot action occurs when no decrement count function is specified by the storage microword. Therefore, up to two words of data from the instruction stream can be buffered when fetching one instruction from storage. Upon completion of the instruction being processed, the next instruction may be available in the buffer and, therefore, need not be fetched from storage.

A savings in processing time becomes obvious, especially if the doubleword being buffered represents four RR instructions. The concept of buffering a portion of the instruction stream can now be extened to include pre-fetching. Although the buffer does speed subsequent instructions, the fetch of the first (current) instruction does require some time. Having the current instruction resident in the buffer is always desirable. To get to this I-buffer condition, the instruction must have been obtained at some point during the previous instruction. This function of reading the next instruction form storage to the I-buffer is termed prefetching and is performed during I-cycles.

As described in "Expanded Local Storage," the TR Reg always contains a value representing the next doubleword address beyond the current I-Reg value. The TR-Reg is always used as the storage address during a prefetch, and the SDBO time-slopting is forced to provide the even word; then the odd word. This guarantees that the I-buffers are loaded with sequential data.



· I-CYCLES CPU 50

Microcode-Hardware Functions

The complexity of the I-cycle functions are increasing. On a prevous diagram (CPU 49) selecting one of three paths after fetching the instruction was necessary. Here also required is to:

- Determine source of instruction (storage vs. buffer).
- Select path if instruction is in the I-buffer.
- Determine whether a prefetch is required.

Also minimizing the time required to perform each of these functions and the basic l-phase functions is desirable.

Microcode branch operations requires CPU time, I-cycle hardware can force a control-storage address to the M-Reg. (for example branch on the op code). This facility is expanded to include all addressing within I-cycles. The I-cycle hardware provides the starting address of the I-cycle routine to the M-Reg, as a function of I-buffer status, instruction format and prefetch requirement. When the CPU encounters a RTN word (to I-cycles) this address is set into the M-Reg and the I-phase of the instruction begins. Thereafter, except for some parts of the RX-align routine, the I-cycle hardware provides the next control-storage address and a gating signal to the M-Reg, until the execution routine has begun. The I-cycle hardware then initializes for the start of the next I-phase.

The minimization of time spent performing the basic I-phase functions requires more hardware control. The I-cycle hardware controls the data inputs and setting of the SPT L Regs and uses this facility to select general registers from local store. By setting the P-Reg to a value of 02 or (62) and gating a portion of the instruction to the L-Reg (R2, X, or B), the microcode can indirectly address any general register, including floating-point registers. This gating to SPTL is done by hardware, and is, therefore, transparent to the microcode. Note that setting SPTL to the desired value one machine cycle before use is necessary.

Furthermore, the I-cycle hardware can force and/or block gating of data through a portion of the Expanded Local Store. This capability is utilized as follows: a microword is executed performing the arithmetic operation of V = LL+V. During the previous machine cycle, a value is set into the P- and L-Regs of: P = 02, and L = R1B2 (for an RX format instruction). The underlined data value gives the general register specified by the B2 field of the instruction as the data source of the A-Reg. Although the microword is attempting to source the V-register, the I-cycle hardware blocks this Expanded Local Store source, and forces the Disp 2 field through the gating to the B-register. The microword function of adding the A- and B-Regs is then completed, with the result destined to the V-register. Thus, the microcode and I-cycle hardware are combined to perform the function of: V = Base + Displacement

Most of the I-cycle microcode/hardware functions is performed this way.

Another significant interaction of microcode and hardware occurs when prefetching and calculation of an operand address are performed simultaneously. The microword executed during a prefetch is of the form: RDW LL ADJ, V + 4. First, the V-Reg is blocked as an address source. Then, the TR-Reg is substituted as the address source for the M-Reg, with gating performed via the PAA, and I-cycle/ADR ADJ path to the M-Reg. At the same time, the Disp field is gated from the I-buffer through the Expanded Local Store to the B-Reg. The A-Reg data source is an indirectly addressed general register, as previously described. This form of microword normally performs an update of the Bregister value; however, this function is blocked and changed to an A + B operation. The function, V = Base + Displacement. is, therefore, performed during the storage-1 cycle. During storage-2 cycle, the destination of data to local store is blocked, and SDBO is time-slotted to the I-buffer, as previously described.

Because the I-cycle hardware and microcode are expected to operate simultaneously, the microcode must consist of specific microwords at fixed addresses. This is obvious because the hardware is providing the control-storage address for the microwords, and then performing hardware functions coincident with the microword execution. What has not been obvious is how the hardware remains in sync with the microcode. This function is performed by routing the M-Reg output back to the I-cycle hardware. Thus, when the M-Reg contains a value corresponding to the control-storage location of an I-cycle microword, the I-cycle hardware can determine what functions are to be performed at the next 0-time (that is coincident with the microword execution).

This introduction has provided the basic functional concepts of the hardware I-cycles. Significant omissions include trapping, share cycles, correction cycles, error conditions, etc. The hardware description in the rest of this section has sufficient explanation for these conditions. The basic I-cycle functions are described "Microcode-Hardware Relationship," page CPU 51. Microcode-Hardware Relationship



I-Cycles Microcode Module Assignment

The I-cycle microcode routine (GAAI) resides in the DF module of control storage. The control-storage address, hardware functions generated, and microcode are directly related. Specifically, the hardware generates control signals (and a next control-storage address) from the contents of the M-register, so as to be active (and coincident) when the control word executed from that address.

The DF module is active if bit 1 of the Mode Reg (external address 08) is on.

MICROPROGRAM MODULE ASSIGNMENT



I-CYCLES CPU 52

I-Cycles Microcode an Control Hardware-Loading of I-Buffers



ycle Entry

I-cycles may be entered (and the I-cycle controls enabled) by two RTN words:

- Conditional: testing for interrupt. C3 bits 5-7 are 111. If an interrupt is pending, the RTN word is executed normally, and does not go to the I-cycle routine. If no interrupt, the return is to I-cycles.
- Unconditional: goes directly to I-cycles when C3 bits 5-7 are 011.

The M-register controls the decode of the return and accesses control storage using the address inputs to the M-register from I-cycles. The I-cycles inputs to the P- and L-registers are also gated. Data is maintained on these inputs by the I-cycle hardware when not in I-cycles (except when performing a storage address adjustment access).

Initial I-Cycle Address

All I-cycle addresses are in the DF module. The initial address for a given instruction is determined by:

- The instruction itself, and
- The requirement for prefetching the next instruction.

Current Instruction Not Fully Contained in I-Bfrs

First, consider the two cases when the instruction is not completely contained in I-Bfr.

- The first case (and highest address priority) occurs when the branch read latch is on. This occurs not only from the most obvious case of macroprogram branch, (detected by the I-register being loaded from EBI), but also from:
- Program modification (detected by storing within the present, or next, storage doubleword address as compared to the I-register).
- A prefetch condition that was not filled during the last I-cycle phase.
- Blocking a trap during a prefetch in the last 1-cycle phase.
- Being in real instruction address compare mode.
- Performing an execute macro-instruction.

0

CPU 54

These examples are summarized as "whenever the instruction must be read from storage." When the branch read latch is on, all other initial addresses are blocked and an address of DF0C is sent to the M-register input.

2. The second case (and next highest address priority) occurs when the op load latch is on. The condition for setting this latch occurs when part of the instruction is in 1-Bfr, but the remainder is in storage. The latch is set during the previous I-cycle phase if it is determined that a prefetch is required, but blocked, and only part of the instruction is in 1-Bfr. When this latch is on, all other initial addresses are blocked and an address of DF14 is sent to the M-register input.

Current Instruction Fully Contained In I-B

With the instruction fully contained in I-Bfr, the next condition considered for an initial address is prefetching. The rule for prefetching is that a prefetch is performed if the:

- Present instruction ends at a doubleword boundary (the next op code is not available) or,
- Next instruction crosses a doubleword boundary. This is determined by hardware as a function of:
- a. The halfword address of the present instruction within the doubleword,
- b. The length of the present instruction, and
- c. The length of the next instruction.
- A prefetch is blocked if the present instruction is decoded to be a branch type, or a special addressing case of an SS instruction, at an address of 6 or E

The initial address is gated to the M-register according to the following table:

Instruction	Without Prefetch	With Prefetch
RR (but not fit. pt. long)	DF20	DF34
RR (fit. pt. long only)	DF24	DF3C
RX (double index only)	DF4C	DF5C
RS, SI, RX (not double index)	DF48	DF58
SS	DF6C	DF7C

I-BFR SET CONDITIONS FOR MOVE I-BFRs

1. RR op and even halfword	=do nothing	
2. SS op and odd halfword	=S/R 0,1	S/R 1
3. Neither of the above conditions	=S/R 0	S/R 1
4. Prefetch and the next instruction is not	=S/R 0, 1	S/R 1
fully contained in the I-BFRs		
I-BUFFER SET CONDITIONS FOR STOR	AGE WORDS	
	(begin in	Stg 2 cycl

1. Prefetch and the next op is not available	=S/R 0, 1, 2	S/R 1.2
2. Prefetch and the next op is available	=S/R 1,2	S/R 2
3. Branch load latch	=S/R 0, 1, 2	S/R 1, 2
4. Op load latch	=S/R 1, 2	S/R 2

I-CYCLES





CPU 54



I-CYCLES CPU 56

I-Cycle Hardware Locations

Selector Channel 4 Direct Control	3215 Console Printer- Keyboard Channel to Channel	CPU Timer Clock Comperator
Phase 2I STG (control Stg. and high main Stg.)	Phase 2I STG (112 or 160K)	Phase 2] STG (208) or 256K)
ECC	ADDR Adjust I, V, W, U, I, BU, TR Regs Logical Regs	Channel Ctris. LRU Reg CPU ADDR ADJ CTRLS I-Cycle Ctrl. Op Code and I-Buffers
Phase 21 STG (control stg. and high main stg.)	Phase 2I STG (112 or 160K)	Phase 2] STG (208 or 256K)

B GATE (CARD SIDE)

CARD LOC	ATION & TYPE	ALD PAG	
8-C3D2	Туре 8551	RU011	Op-U2 Reg-Op Decode
	and the second sec	RU012	Op-U2 Reg-Op Decode
		RU013	Op-U2 Reg-Op Decode
		RU014	Op-U2 Reg-Op Decode
		RU015	Op-U2 Reg-Op Decode
		RU016	Op-U2 Reg-Op Decode
B-C3E2	Type 8552	RU021	Imm Byte-U3 Regs
1. A. 1. 14 1	日本の新聞的なりにつ	RU022	Imm Byte-U3 Regs
	사람이 승규는 것이다.	RU023	Imm Byte-U3 Regs
	이 것 같은 것 같은 것 같은 것 같이 같이 같이 같이 같이 하는 것이 같이 하는 것이 같이 하는 것이 같이 않는 것이 같이 않는 것이 같이 않는	RU024	Imm Byte-U3 Regs
		RU025	Imm Byte-U3 Regs
이 가슴을 물을		RU026	Imm Byte-U3 Regs
나 같은 것이다.	2월 - 1977년 - 1971 - 1971 1971 - 1 1971 - 197		
B-C3F2	Туре 8553	RU031	I-Cycles Generation
		RU032	I-Cycles Generation
	and the second secon	RU033	I-Cycles Generation
		RU034	I-Cycles Generation
		RU035	I-Cycles Generation
B-C3G2	Type 8554	RU041	I-Buff Ctrls and Gates
997 D		RU042	I-Buff Ctris and Gates
		RU043	I-Buff Ctris and Gates
		RU044	I-Buff Ctrls and Gates
	성공 가슴을 가는	RU045	I-Buff Ctris and Gates
B-C3H2	Туре 8558	RU051	PAA Latches
1997 (M. 1997)		RU052	PAA Latches
e de la la		RU053	I-Cycles Controls
		RU054	I-Cycles Error Latches
B-C382	Type 7771	RU111	I-Buffer
11 - 17 - 17 (A).	그는 것은 것은 것을 가지?	RU112	I-Buffer
		RU113	I-Buffer
		RU114	I-Buffer
		RU115	I-Buffer
		RU116	I-Buffer
		RU117	I-Buffer
		RU118	l-Buffer
B-C3C2	Туре 7771	RU121	I-Buffer
		RU122	I-Buffer
		RU123	I-Buffer
		RU124	I-Buffer
		RU125	I-Buffer
	and the second second	RU126	I-Buffer
		011107	I Duffer
		HU12/	1-Duilei
		RU128	I-Buffer

.

1

I-Cycle Hardware Description

The hardware I-cycle concept increases CPU performance by:

- Buffering instructions and prefetching (using a hardware generated address for the next doubleword storage location) instructions while calculating an operand address.
- Performing hardware controls concurrent with microprogram execution.
- Instruction decoding via a hardware forced branch on the eight-bit Op code.
- Windows in Data Flow Diagram show the hardware used by I-cycles.






С Op-Register

The op-register is used to hold the present instruction op code during I-cycles. The output is gated to the U- (2) and S-registers, and is also used for I-cycle decoding and branching to the instruction execution routine.

The decode of the op-register is used to build the ILC, which is gated to the U-register byte 0, bits 0 and 1.



Immediate Byte Modifier Register

- Used only by the Execute software instruction to modify the second byte of the subject instruction (if the R1 field of the Execute instruction is not zero).
- The register is set when the W-reg is destined, with data from byte 3 of the GR specified by the R1 field. (Refer to "Execute Phase (I-Cycles).")



Ε





D Immediate Byte Register

The Imm byte reg holds the second byte of the present instruction during I-cycles. The output:

• Loads the U3-register.

- Is assembled with the base field to be gated to the L-register.
- Is gated to the I-cycle controls.



*Only with branch-read (DF0C)

U-Register (Exp LS 53)

F

Part of this register is set only by hardware. The two-bit Instruction Length Code (ILC) is set to a value determined by the op-reg decode. The condition code (two bits) is used by the I-cycle controls to determine whether a branch-oncondition code instruction branches. Byte 2 is set to the op-code by hardware only. Byte 3 is initialized to the immediate byte by hardware. Bytes 0 (except bits 0, and 1), 1, and 3 may be loaded from EBI by microcode.

Byte 0 Bits 0-1 Instr. length code (hardware set only) Bits 2-3 Condition code Bits 4-7 Program mask

Byte 1 Bits 0-3 Special CPU use

Bits 4-7 OMWP bits

Byte 2 Bits 0-7 Op code (hardware set only) Byte 3 Bits 0-7 Immediate byte



G

Special address-matching function--on a doubleword basis (bits 8-28).

- 1-reg is compared to PAA (V or W). TR-reg is compared
- to PAA (V or W). (These matches are required to determine whether program modification is taking place in that part of the instruction stream that may have been loaded in the I-Bfrs.)
- If the comparison is equal during a store operation, the BR read latch is set. (This forces a new loading of the 1-buffer op-reg and imm byte reg.)

H

V-Register (EXP LS 51)*

Bytes 1, 2, and 3 usually contain the second operand address.

۵

W-Register (EXP LS 52)*

Bytes 1, 2, and 3 usually contain the first operand address.

*If used in a storage word as a storage address, the key register is gated as byte zero.

J Key Register

The key register acts as byte 0 of the I-register when the I-register is destined. The key register contains the storage protect key in bits 0-3. Bit 4 is the fetch protect bit. Bit 7 is 0. With the DAT feature installed, bit 5 is the reference bit and bit 6 is the change bit. For a more detailed description of bits 5 and 6, see "Reference and Change Bit Recording" under DAT in the CPU section.

If the V-register or W-register is used in a storage word as a source, the key register is gated as byte 0 of the V or W-register. If the I-register, IBU-register, or TR-register is used, the key register is gated as byte 0 of that register. The key register is set when the I-register is destined.

IBU-Register (EXPLS 54)*

K

Upon entering I-cycles, I-reg bytes 1, 2, and 3 are set into IBU. If a retry condition is encountered during I-cycles, the instruction cycles may be repeated (return to DF0C). In this case, IBU is moved to the I-reg by the retry microprogram.







I-Register (EXPLS 50)

Instruction counter register, byte 0: if the I-reg is destined, byte 0 is gated to the key register. Bytes 1, 2, and 3 contain the instruction address.

The I-reg is updated as follows: ----I-Phase I-Phase align

М Add Carry (Adder)

- The two low-order adder positions (24-bit adder) add the instruction length to the I-register bits 29-30.
- Adder (except the low two bits) adds a "1" to I-reg bit position 28. This sum represents the next doubleword storage location.
- A carry-out of position 28 is added to bits 27 to 8.
- A carry-out of position 29 indicates that TR bits 8 to 28 are to be loaded into I for a hardware update.
- An adder check turns the I-cycle hardware indicator on.

N TR-Register (EXPLS 55)

- The TR-reg consists of bits 8 to 30, with bit 31 always forced to a zero. (Bit 31 has no latch.)
- Contains an address within the next doubleword after the address in the I-register; used when prefetch or further fetch (DF14) is required.
- Used to buffer the adder output during hardware updating of the I-register.
- I-hardware update, consists of loading I bits 29-31 from TR 29-31. Bit 31 of the I-register is not gated through the adder. I bits 8 to 28 from TR 8 to 28 if the Adder bit 29 had a carry-out. (TR = I + ILC + 8)



CPU 62

I-CYCLES

bered. At update-I time, a carry remembered causes all of TR to be gated back to I. Otherwise, only bits 29 and 30 are used to update I-reg.

1.04

I=1000

N

Update-I-Reg





I-CYCLES **CPU 64**

I-Cycles Control Line Generation

		/	' /	/ /		' . /	' : /	1	/ /		/ /	' /	/ /	' /	' /)
		/.	<u>}</u>		8/	1.		• / :	1	÷/.	÷ /				1.1
			/	/ శి	7	Jen l	, See	/	1	50	Ser 1	/	/ ,	12	
	· . /	0	່ 🤋 /	No.	2/	2	5/	2/	8	Dug	~ /	∿ /		2 ⁴ /	E.
ADDRESS	/ d	e los))))))		\$ 5	" / ð		»/ ~	*/~ e		e'/ .	/ 4	3 5 5 5	5 / S	", "
DF00	ľ —	-	((((2	<u> </u>	(<u>.</u>	2	1 6		(·
DF04	1.1			2				2		1.1			16		
DFOC	2	2	1944	14.5		14		2 5					16		
DF 10	10.5	198					- 6.5 g					2	16		
DF14			2	2		$w^{(1)}$							16		- 전문
DF20						2		6. S. S. S.	2	L			16		
/ DF24							2		2				16		
DF28						2							16		
DF2C							2						16		1. T
DF 30			L	13	2	2			4				1 6		
DF 34				<u> </u>									1.6		
DF3C			<u> </u>	13	2				4				1 6		
DE40					 	2	1.00		2	2			1 6	1	
DE4C									2	2			1 6		
DE 58			1	13		2			4	2			1 6	1	
DESC	-		7	13	2				· · · ·	2			1 6		
DF 60						2		1	2		2		1 6	1	1
DF6C	<u> </u>		194							2			1 6		
DF70				13	2	2		1.1	4		2		1 6	ें 1	
DF7C									1	2	1		1 6	11.1	
(Not) DF		1.1	5.1			1.552					$\{y_i \in \mathcal{T}\}$	1	1	1.2	
RTN	1	1.0			1.000		1.1					111			
to	1 · · ·						1.1			18 L 1				$A^{\prime} \geq 2$	
I-Cycles			1.1.1	1	a da b						1	1.14	1. 7.		1

l C	I-CYCLES CTRL REG BITS				CONTROL RE	GISTER DECODE
-1	2	3	4	5	CONTROL LINE	FUNCTION
0	0	-	1	-	Command Branch Load	Load I-Bfr 0 (I-Bfr 1 in next cycle)
0	0	-	1	-	Force I	Gate I-Reg to B (and Address Adjust)
0	0	1	0	1	Command Op Load	Load I-Bfr 1 (I-Bfr 2 in the next cycle)
0	0	1	0	1	Force TR	Gate TR-Reg to B (and Address Adjust)
	1 0 1	1 1 1	 1 1	0	Command Prefetch	Activate Prefetch Force TR to ADR ADJ Asm (and B if DF30 or DF3C)
-	1	-	=	0	Cp Branch Command	Use op-reg to define the next Cxxx address; or go to the align routine
0	1	0	-	1	L plus one	Force bits 7 and P of data being gated to L-reg to be inverted for FLP long
0	0	0	-	-	Load Op, Imm Byte	Initial load of Op and Imm Byte
1	1 0	0	0	-	Command Move I-Bfr	Used with I-reg to activate the set or reset of the I-Bfrs
1	-		1	-	Gate D1 and B2	Used to gate the correct base or displacement
1	1	-	0	-	Gate D2	field from the I-Bfrs 0 and 1 to L-low or B-Reg
0	0	~	-	0	Set Control Address	Set next address for I-cycles sequence

۰.

These controls are not the result of the control register decode.
 The control line is activated by the corresponding address.
 This line is activated by command prefetch.

- 4. The set/reset of the I-Birs is also controlled by command prefetch.
- 5. This control line is activated, but not used.
- 6. From DF00 through DF7C SPTL is controlled by I-cycles. DFE0 or DFF0 activates this control line again to restore SPTL

after an align. 7. Set P-set LL.



Unique Conditions During I-Cycles

Share Cycle

If a share cycle is attempted during I-cycles, the I-cycles hardware is deconditioned. This occurs when the line 'Not I/O Op' becomes inactive. (See "I-CY Address Generation and Control Decode.") With '(Not)I/O Op' inactive, the generate address, and generate controls latches are deconditioned.



Trap (Not Machine Check)

In the event of a trap during I-cycles, the M-reg is set to the appropriate trap address, and the trap is taken. When the trap address enters the M-reg, the DF Decode turns off, and suspends I-cycle operation. '(Not) I/O Op' is also deconditioned whenever H1, 3, 4, 5, 6, or 7 are on.

Trap (Machine Check)

If a machine-check trap occurs during I-cycles, the failing instruction is retried eight times before entering the hard machine-check routine.

The Retry Microroutine takes the contents of IBU (I-Reg Backup; that is, the current instruction address) and places it in the I-Reg. The retry procedure is to then return to I-cycles to begin processing the instruction again.

I-Cycle Error Conditions

Parity-check errors for I-Bfr 0, I-Bfr 1 Op and Imm Byte regs as well as half-sum check errors are indicated in expanded local-storage register 56 (ICS) I-cycle control display.

Storage Correction Cycle

If a storage correction cycle occurs during I-cycles, the latches Generate Address and Generate Controls are not S/R. The line 'good data' blocks the clock pulse. (See "I-CY Address Generation and Control Decode.")



I-CYCLES

CPU 66

CPU Low Request (Not Prefetch)

- The line 'CPU Low Request' may be activated by:
- Storage-protect check (caused by a mismatch of the storage keys)
- Address check (TR pointing to the doubleword above the top of storage)
- Address translation trap

The two check conditions may be considered as program errors that cause a program check and go to the GICM routine. The address translation trap may cause a similar check, depending upon the availability of the addressed area.



CPU Low Request (Prefetch)

CPU low request is blocked during Prefetch. If a storage check occurs during a Prefetch, it sets the branch read latch. Upon return to 1-cycles, the I-Reg is used (instead of the TR-Reg) to refill the I-Buffer. If a second CPU low request occurs, it indicates a program check.

I-Cycle Timings

	RTN (To I-Cycles)	L. I-CYCLES	EXECUTION RTN (To I-Cycles)	I I I-CYCLES	I I EXECUTION RTN I (To I-Cycles)	ICYCLES		EXECUTION OF SUBJECT INSTRUCTION
1 I-CYCLE LATCH RU044								
2 EXECUTE INSTRUCTION LATCH RU044					 A state of the sta		الحد 	NOTE: BR RD Latch is set
3 ADDRESS RANGE DF00-DF7C								
4 ADDRESS RANGE DF84-DFFC								
S SET IBU REG RU044								
6 UPDATE I REG								
7 S/R (I or V) REG RU044	<u> </u>	l or V = (l)	{	l or V = (()		lorV=(1)	l or V = (V)	 {
8 S/R OP, IMM BYTE REGS	ـــــر	If BR LD Sequence	<i></i>		<i>;</i>			-
9 I-CYCLE CONTROLS (S.P.L. M-REG-not STG-1 Cycle) -								
		NORMAL I-CYCLES	 	i Cycles with read and a	LIGN STATES	F I-CYCLES FOR EXECUT	E INSTRUCTION	

I-Cycle Operational Description

Software instruction decoding on the 3145 is accomplished by a unique interaction of microprogram and hardware.

The microprogram used for instruction decoding is the GAAI routine, which resides in the DF module of control storage.

I-Phase

During I-phase, the instruction is read out of storage and placed in I-buffers. Certain determinations may then be made concerning format, Op-code, and instruction length code. The purpose of the I-phase is to ensure that the correct data is available for use during the E (execute) phase. Upon exit from the I-phase, the Op-code is used to point to the next control-word address of the microroutine for that format. (for a 1A add instruction, address C1AO is used as the entry to the GARR routine).

E-Phase

In the E (execute) phase, data is read, stored, and the correct condition code (CC) is set. The operations indicated in I-phase are performed by using the operands fetched during I-phase. The address sent to the M-reg is that of the entry to the execution routine.



I-CYCLES

CPU 68



- T-Dirs are initialized.
- May have occurred by a program branch to the instruction at address 1004.

Note: See Fetch Sequence and Loading I-Buffers for additional information.

<u> </u>	D2 OF	1D 0	1i,2D	02 47	41 0	5 01	<u>1A</u> 5A
100)4		1.5	1.1			
~0	11853		것없는			2 - N.C	
Ad	dram					1.11	
ma	UP-STOP	202 SP	1. S. S. S.		Sec. 2. 1. 2.		

Further fetch required because the current instruction is not fully contained in the I-Bfrs.

FETCH SEQUENCE

(BRANCH LUAD SEQUENCE)	일양 날 옷에 가슴을 다닌 것이 없는 것이다.		and the second second	
CONTROL WORD RTN	RDW Y, V +	4 Y=	Y, SF Y =	Y, SF
	STG-1 ST	3-2		
M-REG	DFOC INST ADDR	DF04	DF00	DFXX
	DFoC	DF04	DF00	DFXX
	Force I (Ex L not ON)	Block LS Dest		
CONTROLS	Load Op,	mm Byte Regs		an dia mandri dan 1995. Ny INSEE dia mampika ma Ny INSEE dia mampika ma

	Fetch instr	ме 	Op, Imm Byte-R	egs Valid
FUNCTION	;			
an a				termine Next Address DFXX
	I-Bfr 2	S/R S/R	an the second second	in a strangen generation Strangen generation Strangen strangen stra Strangen strangen stra
	i-Bfr 1		an an Alain an Anna an Anna Anna Anna Anna Anna An	
	I-Bfr O			

FURTHER FETCH

- Required when the current instruction is not fully contained in the I-Bfrs.
 - Recognized by:
 - 1. The decoding of the current Op-code, which determines the length of the current instruction.
 - 2. The I-register points to an address within the doubleword that the first byte of the present instruction is located.
 - Accomplished by: (storage word at DF14)

1A 5A 47 30 A0 20 47 FO A2 50

Prefetch

- 1. Setting the Op load latch during DF04 and forcing DF00 to branch to DF14 with the gating line Set Control Address.
- DF14, a storage word, forces the TR-Reg to the B-Reg, thus fetching the next doubleword from storage by using S/R I-buffers 1 and 2 with the even word late in the storage-2 cycle of DF14. During DF10 (early in the cycle), I-buffer 2 is S/R with the odd word.

Note: See "Fetch Sequence and Loading I-Buffers" for additional information.

PREFETCHING

- Required when the next instruction is not fully contained in the I-Bfrs.
- Recognized by a combination of decoding:
 1. The I-Reg points to where the instructions come from within the doubleword.
- 2. The Op code indicates the length of the present instruction.
- 3. Instruction look-ahead knows the format of the next instruction and, therefore, knows the length of the next instruction.
- Prefetch is blocked under the following conditions:
- All branch-type instructions.
 Two SS instructions in succession (TR is pointing to the doubleword just fetched).

NTROL WORD	(Previous)	RD	WY, V	Y-	Y, SF	
	ļ	STG-1	STG.2			
M-REG	DF14	4 Next	DBLE Wd	DF 10	DFXX	
ONTROL REG		DF14	1	DF 10	DFXX	
CONTROLS		Force TR	B	lock LS Dest		
FUNCTION		Feta	ch Next Instr		Determine Next	Address DFXX
	ante de la composition de la compositio Esta de la composition de la compositio Esta de la composition de la compositio		i-Bfr 2	S/R S/R		
			I-Bfr 1			

CPU 70 I-CYCI FS

÷ ...

= LL, DF	(Previous)	Y - LL, DF	RDW LL AD	J. V	
			STG-1	STG-2	
Execution	DF3	14 0	F30 (TR)	Execution	
Routine	,	DF 34	DF 30	Routine	
Bfrs (), 1 Operand to REG		R ₂ to Y REG	orce TR Move I Bfrs 0, 1	Blocks LS Dest Load Bfrs 2, 1 Instruction	
IG POINT		NOT FI PREFI	OATING POINT LONG		

---- 1. CONTROL WORD 2. TIME REFERENCE

4. I-CY CTRL REG

5. CONTROLS

6. FUNCTION

and printer to the

3. M-REG

RR SEQUENCES

1.	CONTROL WORD	
2.	TIME REFERENCE	
3.	M-REG	

4. I-CY CTRL REG

5. CONTROLS

6. FUNCTION

NOT FLOATING POINT NOT PREFETCH

(Previous)

DF 20

DF 20

Move I-Bfrs 0, 1

R₂ Operand to Y-REG

Y = LL, DF

1			6.260-11-1 9.11.15111-1		1		
	(Previous	s) <u>Y</u> •	LL, DF	3 = LL, DF		(Previous) RDW STG-1
		DF24	DF28	Execution)	DF3C
		DF 24	DF:	28		•	DF3C
			Move I-B	ifrs 0, 1			Force TR

Previou	15) j	20.00	RDWI	L ADJ			_ ¥ ≠	LL, DF	<u> </u>	= LL, DF
			STG-1		STG-2		•			
	DF30		, (T	R)		DF2C		DF2	8	Execution
			DF3C			DF	2C		DF 28	Routine
	Fo	rce TR	4		Block L	S Dest		For	ce L Od	ď
		M	ove I-Bfrs C Prefetch for), 1 next Ins	truction	Loa	d I-Bfr R ₂ T	s 2, 1 5 Y-Reg	R21	1 to Q-Reg

FLOATING POINT LONG

R2 to Y-Reg R2 + 1 to Q-Reg

NOT PREFETCH FLOATING POINT LONG PREFETCH

 \mathbb{N}

.....

ì

RX SEQUENCES



2. TIME REFERENCE

I-CY CTRL REG

M-REG

5. CONTROLS

FUNCTION

14.14

3.

4.

6.

DF48

DF48

Execution Routine

Move I-Bfrs

NO DOUBLE INDEX

NO PREFETCH

(or Read and Align Routine)

Force Disp. (U-Bus) to PBA Base to L-Reg (or X if RX instruction

and X # 0, base = 0)

Block ALU A entry

Calculate Operand

Address

if Base is used and = 0



RDW LL ADJ, V + 4





I-Cycles Alignment Re

I-Cycle Entry with No Prefetch

(Previous word was not a storage word.) The align routine is entered at address DFBO, which is a delay word. The delay word allows the destine of the V-Reg. .

le Entry with Prefetch

(Last word was a storage word, requiring a storage-1 and storage-2 cycle.)

The align routine is entered at address DFB4, thereby eliminating the delay word. The delay word is not necessary in this instance because the storage-2 cycle allows sufficient time for the V-Reg to be destined.



I-Cycle Alignment

The align routine enables data in main storage not on a word boundary to be aligned to appear on a word boundary. Consider the following example:

GR 1 = 00 00 10 00 GR 2 = 00 00 20 00



Effective Operand 2 Address = 3D02

Operand 1 is in GR4

The data at main-storage address 3D02 is to be added to the contents of GR-4 and the result placed in GR-4.

In order to get the data at address 3D02, the problem arises that the desired data is not on a word boundary.



During the storage access, the bytes 0, 1, 2, and 3 are placed in the Y-Reg. Through the use of the TA-TB control-word function, bytes 2 and 3 are moved to bytes 0 and 1.

0 1 2 3 Y Before 2 3 2 3 Y After

Y-Reg contents; with operations RDW and MOVE (indirect)



Q-Reg contents; with operation RDW

The information is then placed in the Y-Reg (via an indirect move operation) in the form;



. I-CYCLES CPU 74

I-Cycle Exit from the Alignn Routine

In some instances, an RX instruction can place the operand 2 data into a local-storage register. Once in the register, the operand 2 information may use the RR format-execution routine, Consider the following example:

5 .

During the align routine, the open 22 data is placed in a local-storage register. Upon exiting the align routine, the RR execution routine may be used.









I-Cycles Program Modification

Program modification is detected by a store operation that uses V or W as a storage address within the present, or next storage doubleword address, as compared to the I- or TR-Reg. Only bits 8-28 are compared.

When program modification is detected, the branch read latch is set, and the modified instruction is loaded into the I-Bfrs.

In this example, the Move Immediate Op-code causes the modification of a portion of storage that is already loaded in the I-Bfrs. To continue operating on the old information in the I-Bfrs would cause the wrong result. To replace the old information, the branch read latch is set, and the modified information is loaded into the I-Bfrs early in the next I-phase.



D2	00	F1	24		
			·	ADDR.	INSTR.
	I - BI	R 1		400	92FF0405 D200F124
D2	00	F1	24		
••••••	I - 8	FRO			

92 FF 04 05

405 MVI FF, INOUT + 1 124F700 INOUT MVC OUTPUT (00), INPUT

LABEL







	Branch Loop Example										I-CYCLES
$\frac{1}{10} + \frac{1}{100} + \frac{1}{$	C428 RTN LNK 1 +0-Time -Gate Rtn to I-cycle	DF0C RDW Y ADJ, V + 4 force I → M STG-1 Cycle	STG-2 Cycle	DF04 Delay Load Op-Reg Imm Byte SPTL Complete I-bfr load	DF00 Delay Load Op-Reg Imm Byte and PSL. Branch on Format	DF48 V = LL + V Calculate Address update I-Reg from TR-Reg	C470 I = V, D7 Successful Branch On Condition Instr.	C428 RTN LNK Check for pending interrupt.		DFOC Automatic next Address = DF04	
8 - Str 54, from TCY 7 - Str 54, from TCY 9 - Str 14, from TCY 9 - Str 14, from TCY 9 - Str 54, from TCY	3 + Force I-Reg to B-Reg 4 + Gate I-CY to M-Reg 5 - Select EVEN data 6 + S/R I-Bir 0 7 + S/R I-Bir 1 8 + S/R I-Bir 1 9 + S/R OP + Imm Byte 10 + S/R I-Bir 2 11 - Gate D_1 or B_2 12 - Force Displ 13 - Base * Zero state 4 - Gate Base to LL									DF04 Automatic next Address = DF00 Bit 5 = 0, set CA Branch on Format no Prefetch - DF48 DF48	
$\frac{24}{3} - \frac{54}{5} Cr. Address$ $\frac{24}{3} - \frac{24}{5} Cr. Address$ 24	5 - Set SPL from ICY 6 - OB Branch Cmd 17 - Move I-Bfr Cmd 18 - S/R I-Reg 19 + S/R I-lor V-Reg 20 - Update I-Reg 21 - Branch Read Lt. 22 + S/R I-Back-Up 23 + I-Cycle Latch							Early reco Reg Distin Branch RE	gnition of I- etion = Set) Lth	No Special RX. Instr. Branch on Op-Code XCC0 C470 Successful Branch (Not Successful = C474)	
Register Contents* 47 : F0 : 15 : 00 5A : 24 : 2D : 02 5A : 102 5A : 100 5A : 100	24 – Set Ctr. Address	Save I → IBU								C428 Branch Read Litch on = DFOC	
LBfr 2 47 i F0 i 15 i 00 5A i 24 i 2D i 02 5A i i 1 02 5A i 1 0 0 Fo i 1 7 02 00 Fo i 1 7 02 Fo i 1 5 0 Fo i 1 7 02 Fo i 1 5 0 Fo i 1 7 02 Fo i 1 5 0 Fo i 1 7 0	Register Contents *										
$\frac{1}{12610}$ $\frac{1}{12610}$ $\frac{1}{12610}$ $\frac{1}{12610}$ $\frac{1}{12610}$ $\frac{1}{102}$ $\frac{1}{100}$ $\frac{1}{1$	1-Bfr 2	+	47 1 F0 1 15 00	5A 1 24 1 2D 1 02	5A 1 24 1 2D 1 02	5A 24 2D 02	5A 102	5A_1 1.02			
* The Register Contents shows the value after execution of the microword, (M-Reg. C-Reg contain next word.) EGISTER CONTENT IS VALID IF THE OPERATION IS ERFORMED ONLY ONCE Reg Reg Reg BU-Reg 500 5	J-Bfr O		47 F0 15 00	47 F0 15 00	47 F0 15 00	5A 24 2D 02	5A 1 02	5A 1 1 02		500 Main Storage 5	07
(M-Reg. C-Reg contain next word.) 5A 02 00 24 47 02 00 F0 F0 F0 F0 F0 F0 F0 </td <td>* The Register Contents shows the value after execution of the microword.</td> <td></td> <td>Op-Reg Imm Byte 5A 24</td> <td>IOp Imm Byte 47 F0 I</td> <td>Op Imm Byte 47 F0</td> <td>I Op Imm Byte 47 F0 I</td> <td>Op Imm Byte 5A 24</td> <td>Op Imm Byte 5A 24</td> <td></td> <td>Even WD Odd</td> <td></td>	* The Register Contents shows the value after execution of the microword.		Op-Reg Imm Byte 5A 24	IOp Imm Byte 47 F0 I	Op Imm Byte 47 F0	I Op Imm Byte 47 F0 I	Op Imm Byte 5A 24	Op Imm Byte 5A 24		Even WD Odd	
IEGISTER CONTENT IS VALID IF THE OPERATION IS ERFORMED ONLY ONCE Image: Content is valid if the operation of the base register is assumed to be zero for this example. This causes an unconditioned in the first unconditioned in the first instruction (47). The ADD instruction (5A) is used only to provide data for the registers. Reg 500 Image: Content is operation of the base register is assumed to be zero for this example. This causes an unconditioned in instruction (47). The ADD instruction (5A) is used only to provide data for the registers. Reg 500 Image: Content is operation of the provide data for the registers. BU-Reg 500 Image: Content is operation of the provide data for the registers.	(M-Reg, C-Reg contain next word.)	1 ·	5A 02 00 24 S P T L	47 02 00 F0	47 02 00 F1 S P T L	47 02 00 FC	0 47 02 00 F0 S P T L	47 02 00 FC S P T L		47 F0 15 00 5A 24 2D 04	
Reg 500 1 1 504 (from TR) 504 (from TR) 504 (from V) (R.Reg 500 1 1 506 508 - BU-Reg 500 1 1 1 508 - (Reg 500 1 1 1 1	EGISTER CONTENT IS VALID IF THE OP ERFORMED ONLY ONCE	ERATION IS								GR1 = 00000000 The base register is assumed to be zero for this example. This causes an unconditional branch to the first instruction (47). The ADD instruction (5A) is used only to provide data for	
S00	l-Reg TB-Reg	500	I	1 1 500	1 1	<mark></mark> 5	504 (from TR)	50E	00 (from V)	, the registers.	
	BU-Reg V-Reg	500	500	1	$\frac{1}{1}$ 504 (1 + 4)	1					

Partial Instruction Stream Cles Example

The following instructions are contained in main storage. 1A 32 5A 41 2D 02 1B 46 (IC set to address 1000)



Add (1A) Instruction

The 1A instruction starts at address 1000 in main storage. To start processing at that address, assume that the set IC key is pressed. During the set IC microroutine, the I-Reg is destined. (This sets the instruction counter to address 1000.) The IBU-Reg address 1000 and the TR-Reg contains address 1000, an address within the next doubleword. Destining the I-Reg as a result of the set IC microroutine causes the branch read latch (RU 043) to set. Upon completion of the set IC routine, the start key must be preseed. This initiates the start microroutine. At the end of the start microroutine is a 10 RTN LNK microword. The combination of the 10 RTN LNK and the M-Reg not being at address DFXX along with branch read latch (previously set on) forces the M-Reg to address DF0C. The M-Reg addresses control storage and reads out the control word at address DF0C.

CONTROL WORD AT DFOC (RDW Y ADJ, V + 4).

- 1. M-reg is at DFOC.
- 2. This is a RDW, force I to B and read the doubleword at MS 1000 and put it on SDB0.
- a. This doubleword (1A32 5A41 2D02 1B46) is in the SDBO Pre-Asm early in storage-2 cycle (storage-1 cycle was addressing MS).
- b. Late in Stg-2 cycle, the even word from the SDBO Pre-Asm is gated to the SDBO Asm through EBI to the I-Bfrs. I-Bfrs 0, 1, and 2 are "set/reset" (S/R). At the end of storage-2 cycle, all three buffers contain the same information (the even word).
- During the control word DF0C, the M-reg was at address DF0C. Through hardware, the M-Reg is forced to DF04 by using the gating line 'Generate Address' (*RU 031*), d. The M-Reg reads out the control word from controlstorage address DF04.

CONTROL WORD AT DF04 - (Y

 Early in the cycle, the odd word of the doubleword read out during DFOC is gated from the Pre-Asm to the SDBO Asm through EBI to the I-Bfrs. This time only I-Bfrs 1 and 2 are 'S/R'. I-Buffers 1 and 2 contain the same information. The odd word I-Bfr 0 contains the even word.

I-BUFFERS

2	2D02	1B46
1	2D02	1B46
0	1A32	5A41

- 2. The Op-code (RU 118) and the Imm byte (RU 128) are set into the Op-Reg and Imm byte Reg, respectively. (I-Reg indicates which byte to gate to the Op-Reg.)
- 3. The I-CY controls decode the Op-code and set the ILC in U0 Bits 0 and 1. The ILC is also made available to the add-carry. (The ILC is available to the add-carry at the time the I-Reg is updated.)
- 4. The Op-Reg decode and the I-Reg are used to determine whether the present instruction is fully contained in the buffers. The Op-code indicates the length of the current instruction, and the I-Reg indicates what part of the doubleword the current instruction came from. Therefore, through this combination of decoding the need for further fetching can be determined. (*Further fetch* is defined as the condition when the present instruction is not fully contained in the I-buffers.) In this case, no further fetching is required. The present instruction is fully contained in the I-Brfs. The Op-load latch (RU 043) is not turned on.
- 5. The present instruction is fully contained in the I-Bfrs. The I-Bfrs must be decoded to see whether the next instruction is fully contained in the I-Bfrs. Instruction Look-Ahead is used to decode the format of the Op-code that is following the current instruction. The decoding of this format indicates the length of the instruction that follow the current instruction. The I-Reg keeps track of the address that the data came from within the doubleword. Through this combination of decoding whether the next instruction is fully contained in the I-Bfrs can be determined. If not, a prefetch is performed. (A prefetch is defined as the condition when the next instruction is not fully contained in the I-Bfrs.)The next doubleword is loaded using the TR-register contents as an address. If the I-Bfrs do not have enough room for the next doubleword, the prefetch is blocked. Prefetch is also blocked on all branch type O-codes. (In this case, a prefetch is not required.)
- The S- and L-Regs are set from the Op-Reg and Imm Byte Reg. This occurs during I-CY and when a cycle is taken. LL =operand 2. LH = Operand 1.
- During the control word DF04, the M-Reg was at address DF04. The M-Reg is forced via hardware to address DF00, by gating line 'Generate Address' (RU 031).
- M-Reg at DF00, the M-Reg reads out the control word from control storage address DF00.

CONTROL WORD AT DF00 (Y = Y, SO)

- 1. This is a delay word that allows the hardware to develop the next address.
- The gating line 'Set Control Address' (RU 032) is needed because a new sequence of control words is being started. Note that DF20 does not always follow DF00. In this case, the Op-code has been decoded to 1A and causes a branch to DF20.
- 3. During the control word DF00, the M-reg was forced to DF20.
- Once the instruction is contained within the I-buffers, the next address is formed as follows:



M-Reg at DF20, the M-Reg reads out the control word from control-storage address DF20.

CONTROL WORD AT DF20 (Y = LL, DF) (the last cycle in I-Phase)

- 1. The word Y = LL, DF. (LL points to operand 2.)
- 2. The contents of GR2 are placed in the Y-Reg.
- 3. At the end of this cyle, the value to be destined to the Y-Reg is in the Z-Reg.
- U2 and U3 are set from the Op-Reg and Imm-Byte-Reg, respectively, so that the information is available to the microprogram during E-phase.
- 5. Update I-Reg (I = 1002) (*Note:* TR is updated after the S/R to the I-Reg.)
- 6. At the conclusion of I-phase, the Op-code has been decoded (in this case 1A) and the next address is know (C1A0).
- 7. The M-Reg is forced to C1A0. With M-Reg at C1A0, the M-Reg and a set of

With M-Reg at C1A0, the M-Reg reads out the control word from control-storage address C1A0.

EXECUTE PHASE (I-CYCLES)

The control word at address C1A0 is read out (the first cycle of E-phase of the 1A instruction).

CONTROL WORD AT C1A0 - (LHC = LH + Y)

- A. The following hardware functions are completed for 1-cycles.
 1. S/R the Op-Reg and the Imm Byte Reg from the next Op-code and Imm Byte. The I-Reg was updated just before leaving 1-phase; therefore, the I-Reg is pointing to the next instruction.
- In E-Phase, the Op-Reg and Imm byte Reg cannot be set into the S- and L-Regs. The system must be in I-cycles to gate the Op-Reg and Imm byte Reg to the S- and L-Regs.
- 3. The TR-Reg is updated to point to the next doubleword.
- B. The control word read from address C1A0 performs the following.
- 1. LH is pointing to GR3.
- 2. Y has not been destined (Z = Y).
- 3. At source time, the Z-Reg is gated back to the B-Reg controlled by Destination Look-Ahead, and the contents of GR3 are gated to the A-Reg. At the end of this cycle, the sum of GR3 and 2 is in the Z-Reg and is destined to GR3 in the next cycle.
- 4. The 1A add instruction is now complete (during E-phase of the add instruction):
- a. The S-Reg is changed by Stat sets.
- b. The U2- and U3-Regs still contain the Op code and Imm byte of the current instruction.
- c. The L-Reg still contains the address of operations 1 and 2.
- d. The microcode sets the condition code, test overflow, etc.
- e. While in E-phase of the 1A instruction, the I-CY controls are decoding the next Op code to determine the controls required for the next I-phase operation and the first address that will be used upon returning to I-CY.

I-CYCLES CPU 82



The second operand is added to the first operand, and the sum is placed in the first operand location.

RX With Double Indexing and Alignment

- The definition of *double indexing* is: Neither base nor index fields are using GR0 (zero). Another control word is needed to calculate the operand.
- Alignment: (operand 2 is pointing to a storage address; the data at that address is to be added to the contents of the GR that operand 1 is pointing to).
- Fetch data from storage (using operand 2).
- Align data in the GR.
- Both operands are now in local storage.

The same routine may be used as in the RR Add to add the two operands because both operands are in local-storage registers.

time for this. Developing address DF4C after E-Phase of the 1A Op-code. During E-phase of the 1A op code, the op-reg contains the next Op-code 5A. During this time, I-CY controls are setting up for the next address in the SF module by decoding the next op code 5A. The next address 4C is developed by the lines ILC, RX, RS, SI, SS, and IF MORE THAN ONE M-WORD. (The line 'if more than one M-word' is developed from the decode of double index.) After the execution phase of the Add (1A) operation is complete, the RTN LNK microword develops the gate line 'set control address' (not DF). 'Set control address' gates the developed address 4C to the M3 assembler, and gates DF to the M2 assembler. Rt nto I-CY gates the assemblers, on the M-Reg, setting the M-reg to the next address DF4C.

At the end of DF4C, the V-Reg = Base + Displacement (2D02).

B The transition from address DF4C to DF40 is accomplished by using the decode of the I-CY Ctrl Reg 4C and gating line 'Generate Address'.

At the end of DF40, the V-Reg contains X + B + D (3D02).

Address DF40 to address DFB0 (going to the Alignment Routine):
 1. 'RX withalignment' develops the gating line 'Op-Branch to DF'.

main-storage address, must be

taken through the address Asm

because Destination Look-Ahead can not be used at this time. The DFB0 delay word allows

> 2. 'Op Branch to DF' gates 'Align Entry' (in this example, the last control word in I-CY is not a prefetch. Therefore, the 'Low Bit Y' is zero and the Op code 5A develops a B) to the M3 assembler and DF to M2 Assembler.

'Gate I-CY to M-Reg' gates the assemblers to the M-Reg. The M-Reg contains DFB0, which is the first address in the align routine.

Being in the align routine deactivates the I-CY controls. This is done by M3 bit 0 being on, which deactivates 'generate address' latch and 'Generate Controls'. With the I-CY controls deactivated, the control words in the align routine develop the next address. (The exception to this is the first and last addresses of the align routine. which are developed by I-CY controls.)

Note the setting of U2-U3, updating I-Reg, etc., when going from DF40 to DFB0 (Op-Reg and Imm byte are set in the first control word of E-phase of the 5A Op-code.)

The first control word in the align routine is a delay word to allow the destining of the V-Reg. This is necessary because the next control word 'DFB4' in the alignment routine is RDW Y ADJ, V + 4, TB.

The V-Reg is addressing storage, and 'ADJ' (hardware adjustment) specifies that the V-Reg must be taken through the Addr Asm to address storage. Destination Look-Ahead may not be used to get data to the Addr Asm. Therefore, DFB0 must be used to allow destining of the V-Reg so that it is available to the address Asm during 'DFB4'.

At the end of the alignment routine, the data to which operand 2 was pointing is fully aligned and is in local storage in the Y-Reg .

The last address in the align routine is DFE0. M3 bit 0 being on and the line 'F0, E0 (M3)' activate the gating line 'Op Branch' (on Op-code). Upon exiting the align routine, the Op code 5A is still in the op-code register. The gating lines 'Op Branch' (on Opcode) cause a hardware branch. LH is pointing to operand 1 (GR4), and the data to which operand 2 was pointing is aligned and in the Y-Reg. Branching to the RR add routine and adding the two locations in local storage is now possible. The first control word found in the RR add (at address C1A0) is LHC = LH + Y. Hardware takes the present op-code 5A and 'minus 4' from the left hex digit of the present op-code of 1A. This is how the I-CY controls set up the next address when leaving the alignment routine of some RX instructions, op-code) to the execute routine for a RR instruction.

MVC (D2) Instruction Example

MAIN-STORAGE

ADDRESS 1004

D2 0F 1D 01 2D 02 47 F1 05 01 1A 5A

Further fetch required because the current instruction is not fully contained in the I-Bfrs.

I-Phase for MVC (D2) Requiring Further Fetch

- Set I/C to 1004.
- DF0C: (M-3 bit 5 is on, forcing the odd/odd time slot) late in storage cycle 2, I-Bfrs 0, 1, and 2 are S/R with the odd word.
- DF04: early in the cycle, I-Bfrs 1 and 2 are again S/R with the odd word. The Op-Reg. Imm byte, ILC and S and L are set. If the current instruction is not fully contained in the bfrs the 'Op latch' turns on.
- DF00: delay 'set control address'.
- DF14: 'Op load' latch on and 'set control address' caused generation of DF14. RDW: force TR to B-Reg (fetch next doubleword). Storage-2 cycle S/R Bfrs 1 and 2 comes from even word. Next, the 'generate address' gating line is needed.
- DF10: (delay) S/R Bfr 2 from odd word. Format the branch. A new sequence of words is being started in I-CY. Therefore, the gating line 'set control address' is needed. The op-code in the next address' 6C' is developed by:
- a. ILC, and SS develop the '6'.
- b. More than one control word and the control line 'not RR or FLP with prefetch' develops the "C".
- DF6C W = LL + W (add base 1 + displ 1 to W-Reg) Block W' as a source and gate Displ. 1 from Bfr 0 to the B-Reg. LL points to base register. At the end of this cycle, the Z-Reg contains operand 1. B2 is gated from Bfr 1 to LL. (B2 must be in LL for the next address to calculate operand 2.) To develop the next address, the gating line 'generate address' is activated.

- DF60 V = LL + V, Block V as a source, gate displ. 2 from Bfr 1 to the B-Reg. LL is pointing to Base Reg 2. During this cycle, the W-Reg was destined. At the end of this cycle, the Z-Reg contains operand 2. 'Op Branch' to CD20. Leaving I-phase to E-phase, sets U2 U3. Update the I-Reg.(Update I-reg before the bfrs are moved, and set Op-Reg and Imm byte.) Move Bfrs (this overlaps into first cycle of E-phase).
- CD20: Set Op Reg and Imm byte with Op-code and Imm byte of the next instruction. (The I-Reg indicates where the Op-code and Imm byte are in the Bfrs.)
- CD20: Set ILC (from next Op.code). Update TR. The S and L Regs are set from the Op and Imm byte because the addressing range is out of I-CY (DFXX Module).
- During E-phase of the MVC, the microprogrammer has the Opcode and Imm byte available in U2 and U3. The length is decremented in U3. The condition code is set in U0 bits 2 and 3. The S-Reg is changed by Stat sets. Move characters. During E-phase, the I-CY controls are decoding the next Op-code 47 to develop the first address that is to be used when returning to I-CY after E-phase of the MVC Op-code.
- XXXX RTN LNK I1 The I-CY controls provide an I-cycle starting address of DF48. If no interrupt is pending, this is the data value set into the M-register.
- DF48 V = LL + V Block V as a source, gate displ 2 from Bfr 1 to the B-Reg.LL is pointing to base Reg 1. 'Op Branch' to C470. Leaving the I-phase, set U2U3. Update the I-Reg and move the I-buffers.
- C470: Beginning of E-phase for branch instruction. V-Reg is destined with sum Base + Displacement.
- Note that a software branch has occurred. Refer to example 1
 - for any further explanation.

I-CYCLES CPU 84

Execute 44 Instruction Example

The execute instruction causes one instruction in main storage to be executed out of sequence without actually branching to the object instruction. For example, assume that a move (SI) instruction is located at address 3820, with format as follows:

INSTRUCTION

STREAM_



ADDRESS

CPU 86

I-CYCLES

ADDRESS

However, after execution:

Register 1 is unchanged. The instruction at 3820 is unchanged. Storage location 8919 contains FF. The CPU next executes the instruction at address 5004 (PSW bits 40-63 contain 00 50 04)

- 1. a. Assume that the instruction being processed had prefetched storage location 5000.
- b. During the E-phase, the I-cycle starting address of DF48 is provided as data input to M-Reg.
- 2. 10 for Instruction O-Code '44'
- Base 1 was gated to LL during the RTN to I-CY.
- I-Reg = 5000
- During E-phase of the previous instruction, I-CY controls recognized that the current and the next instructions are fully contained in the I-Bfrs. Therefore, the first address of I-CY for Op-code 44 is DF48.
- DF48 'set control address' and the lines 'ILC, RX, RS, SI, SS, and not RR or FLP with prefetch' cause address DF48, to be gated to the M-Reg.

Note: Op Branch to DF. The three control words used for the 'Execute Instr' are referred to as align. That is, they are in the DF module. When entering these three words, the I-CY controls are inactive. The I-Reg is updated to 5004 (next sequential instruction).

- DFA4: 'Op Branch to DF gates out align entry A4.
- The word at DFA4 moves the GR specified by the R1 field to the W-Reg, and doing so:
- a. Sets the execute latch on (I- or V-Reg = V)
- Loads the immediate byte modifier reg and ORs it with Imm byte reg because R1 is not GR0. (This is done during I-phase of the instruction.)
- RTN LNK (DF84): The first address in I-CY is DF0C because the 'branch read' Latch was found on during DF84.

I-Phase for the Subject Instruction

- DFOC: The 'execute' latch is on, indicating that the V-Reg should be used in place of the I-Reg to address storage. The V-Reg contains the address of the subject instruction that was calculated as operand 2 during I-phase of the execute instruction.
- V + 4, used if further fetch is required to load the subject instruction.
- After E-phase of the subject instruction, returner DF0C and reload the Bfrs.

The I-Reg was not updated during the subject instruction; therefore, the next sequential instruction is loaded from the instruction stream (from address 5004).

A-REGISTER, B-REGISTER, AND ALUS

Arithmetic and logic operations in the CPU are processed by two one-byte arithmetic and logic units (ALUs). ALU operations and program symbols:

Symbol	Operation
Α,	AND
OR,	OR
,OE,	Exclusive OR
4	True ADD
	Complement ADD
,D+-,	Decimal ADD
+	Binary ADD
A	Complement AND

The two input operands for the ALUs are entered into the A-register and the B-register and gated through their output assemblers.

Two ALUs are provided to allow simultaneous operation in halfword operations and to provide checking during logic byte operations.

By using two ALU cycles during a CPU cycle, the ALU system can binary add two fullword operands.

Normal operations are for either byte operands or fullword operands as defined by the control word.

Special gating controls are provided in the entry of each ALU to allow manipulation of byte operands for logic and arithmetic operations.

The description of ALU operation in control words is given in the CPU Hardware (CPU)section.

A-REGI

A-REGISTER, B-REGISTER, and ALUS CPU 88



A-REGISTER AND A-BYTE ASSEMBLER

The A-register is the A-source entry for the ALU system and enters data into main storage through SDBI. Incut data for the A-register comes from the A-local storage,

1 S 🕈

the external registers, or previous ALU output from the Z-register.

An error condition is reported in MCKA byte 2 bit 4. The A-register is a fullword (four-byte) register that normally enters the fullword.

The A-byte assembler provides the means to present the Aregister bytes to the ALU system and to the SDBI. The A-byte assembler has a four-byte output to allow

assembling a fullword to feed the SDBI.

Only the byte 2 and byte 3 assemblers feed the ALU system and have gating to enter any of the four A-register bytes into either output.

Byte 0 assembler cannot gate the A-register byte 1, and byte 1 assembler cannot gate the A-register byte 0 into their outputs, because they are not required in defined operations.

Parity is checked on all four bytes of the A-register during arithmetic fullword cycle, storage-word, and word-move cycles. On CPU byte cycles, only the byte being specified is checked at the A-byte assembler.

The high-order three bytes of the A-byte-assembler do not have a parity check but depend on the receiving area checks.

The byte 3 assembler is parity-checked because of its use in decimal adding, and the error condition is reported as an A-register error.

withits and



A-REGISTER, B-REGISTER, and ALUs CPU 90

B-REGISTER AND B-BYTE ASSEMBLER

The B-register serves as the B-source entry for the ALU system and the data address entry for the M- and N-registers. Input data for the B-register comes from the B-local storage, the A-register, the SPTL external, or a previous ALU output from the Z-register.

An error condition is reported in MCKA byte 2 bit 5.

The B-register is a fullword (four-byte) register that normally enters a fullword.

The B-byte-assembler provides the means to present the B-register bytes to the ALU system.

The B-byte-assembler can gate any byte of the B-register into both byte 2 and byte 3 outputs to feed the respective ALUs. Parity is checked in all four bytes of the B-register during arithmetic fullword, storage word, and word-move cycles. On CPU byte cycles, only the byte being specified is checked.

The B-byte assembler has a parity check on both outputs, and an error condition is reported as a B-register error.



ALU A-ENTRY GAT

The logic requirements of the system make it necessary to be able, to block or transpose (cross) a part or all of the A-entry byte operand.

The high (0-3) and low (4-7) portions of the operand are separately gated to allow moving a portion straight to the ALU or to cross high and low in entry to the ALU.

When none of the ALU entry gates are activated, the entry is blocked and the operand is presented to the ALU as zeros. For a normal entry, both straight gates are raised and the full

byte is entered. For a crossed or transposed entry, both cross gates are raised. For operations requiring only one portion of the operand, the appropriate gate is raised to enter the desired portion into the specified position of the ALU with zeros in the ungated portion,

ALU A ENTRY GATING



Op	1. A.		·	Entry	Byte	1	Gate		
Sym	Operation			нннн	LLLL	SH	СН	CL	SL
BS	Straight	• • • • • • •		нннн	ī.ī.	- × -			
BSO	Block High and Low		2.1	0000	0000	· ·			
BSH	Gate High; Block Low	and the set		нннн	0000	×		(1,2)	
BSL	Gate Low, Block High			0000	LLLL			1.44	X
BSX	Cross High and Low			LLLL	нннн		×	×	
BSXH	Cross; Gate High; Block Low			LLLL	0000		x		
BSXL	Cross; Gate Low: Block High			0000	нннн		L _	Ľ	

B-ENTRY GATING

The logic requirements of the system make it necessary to be able to block all or part of the B-entry byte operand and to enter the shift and K factors.

The high (0-3) and low (4-7) portions of the operand are separately gated to the ALU entry in order that one or both portions may be blocked and entered as zeros for the operand. For a normal entry, both gates are raised and the data is transferred straight to the ALU.

When the normal entry is blocked, the byte developed in either the shift assembler or the K-assembler can be gated. The ALU B-entry has a true/complement gating that reverses the binary bit levels of the operand byte when the complement line is raised.

The complement line is under control of the minus operation sign or the presence of the S0 bit when the operation sign is \pm in the control word.

ALU B-ENTRY GATING



чн ссссТ		• •
HH LLLL	×	×
0 0000		
HH 0000	x	
O LLLL		L×
	H LLLL 0 0000 H 0000 0 LLLL	H LLLL X 0 0000 H 0000 X 0 LLLL

SHIFT GATING

Special ALU entry gating is provided to allow the two-cycle right shift (four bits) operation.

During both cycles, the low-order four bits of byte 2 and the high-order four bits of byte 3 are assembled as a byte for the B entry of ALU-3.

During the first cycle, the high-order four bits of the T-register are set into buffer latches in the shift assembler, and the low-order four bits of byte 3 enter the T-register.

During the first cycle, the low-order four bits (4-7) of byte 1 and the high-order four bits (0-3) of byte 2 are assembled as a byte for the B-entry of ALU-2.

During the second cycle, the buffered T-register bits and the high-order four bits (0-3) of byte 2 are assembled as a byte for the B-entry of ALU-2.

The A-entries for both ALUs are blocked and enter zeros for both cycles.

SHIFT ASSEMBLER



ALU K-ASSEMBLER

The K-assembler for the ALU gates fixed constant and constants defined by the control word to the B-entries of the ALUs. Byte 2 of the control word in the C-register can be gated directly as a full byte to the ALUs.

The low-order four bits (4-7) of byte 2 in the C-register can be gated to the low-order (4-7) of the K-byte with zeros in the high-order (0-3).

The low-order four bits (4-7) of byte 2 in the C-register can be gated to the high-order (0-3) of the K-byte with zeros in the low-order (4-7).

The high-order four bits (0-3) of the T-register define the bytes to be read or stored are converted to a binary count byte for adjusting the address and the count,

The four memory flag bits from the selector-channel buffers are converted to a binary count byte for adjusting the address and count during share cycles.

When C-register byte 0 bits 2 and 3 are used to define the size of the data transfer in a storage word, these bits are converted to a binary count.

For a decimal operation requiring binary-to-decimal adjustments, K60 and K06 are used as inputs on the second pass. The resulting output developed is K60, K06, or K66.



CPU 92

ARITHMETIC AND LOC UNIT (ALU)

The arithmetic and logic unit (ALU) performs the logic manipulation and adding operations in the CPU.

Two ALU units are provided to allow halfword binary and word-move operations in one pass.

- * Two ALU passes can be made during a CPU cycle to complete
- a fullword binary or word-move operation.

Each ALU consists of the A- and B-entry gates, the logic and arithmetic circuits, and output gating to position the output byte in the Z-register.

ALU-3 entry lines are checked for invalid decimal digits to ensure a correct decimal output with errors reported in S1 bit. A carry look-ahead circuit is shared by the ALUs to allow simultaneous arithmetic processing of two successive digits or bytes.

During logical operations, both ALUs are fed with the same data, and a logical check circuit compares the results and reports errors through MCKA2 bit 2

The ALU logic circuits develop four outputs for any input presentation, and gates defined by the control-word operation select the appropriate output to set the Z-register.

Three of the ALU circuits provide the logic AND, OR, and OE outputs; the fourth combines these with the carry inputs to develop a full-sum output used for binary arithmetic.

The A-logic function is performed by raising the complement

line in the B-entry and gating the AND output. A parity prediction (generator) circuit on each ALU develops an output parity based on the inputs and the operation and is proven by the parity check of the Z-register.

In word operations the ALU-2 output is set into Z0 and Z2 and ALU-3 output is set into Z1 and Z3 on the first pass with an

update of Z0 and Z1 on the second pass. For byte operations ALU-3 output is set into all Z-register bytes to allow the SDBO byte assembler to enter the byte into any position.

Half-Sum Checking

The half-sum lines (OE) developed from the two ALU operands are tested against the parity bits of the two operands to check the parity of the entry data.

E, ∴h half-sum line indicates the odd/even relation of a bit position within the operands.

Developing an odd/even count from the eight half-sum lines indicates the level of the entry data bits (16) that should check with the level of the entry parity bits. A separate latch is set f. each pass and each ALU, but the

- ALU-2 indications are blocked for decimal operations.
- A detected error sets an ALU check indicator in MCKA2 (bit 0 for ALU-2; bit 1 for ALU-3).





Z-REGISTER

The ALU results are set into the four-byte Z-register. The ALU result data can then be routed to:

- The S, P, T, or L-register
- The A- and B-registers
- The D-register

Z-register data (ALU result) is tested, if so specified in the control word being executed, to set or reset S-register bits.



The direct path from the Z-register to the S, P, T, and L-registers permits the setting of these externals earlier than other external facilities. This capability is necessary because these registers are frequently used by the next control word for status information and for local-storage and externalregister addresses. Data destined to all other facilities is set into the D-register and destined during the next controlword cycle.

ALU data set into an external register (other than S, P, T, or L) in one control-word cycle is not available as source data for the next control-word operation.

One-byte Ops feed all four bytes of the Z-Reg with the same data.

A path is provided from the Z-register to the A- and Bregisters. This path allows ALU data destined to local storage (not externals)by one control word to be used as source data by the next control word. Example:

A control word is executed, and the result is gated to the R-register.

The next control word requires the R-register as the B-source.

The Z-register to B-register gate is activated because the result of the previous control word has not been set into the R-register. The Z-register is also gated to the D-register in normal fashion.

Refer to "Local Storage, Destination-Look-Ahead" on CPU 11 Storage.

Any combination of byte selection is possible. For example: if only byte 2 of a local-storage location is altered, then only that byte is routed to the A- and B-registers from the Zregister. Bytes 0, 1, and 3 come from the local-storage location addressed by the control word being executed. Consequently, no matter which bytes of a local-storage location are altered and destined by a control-word operation, the altered data is available as source data for the next control-word operation. Address information derived from the control word being executed determines whether Zregister data is routed to the A-register, the B-register, or both,





D-REGISTER

<u>____</u>

The four-byte D (destination) register is set to the ALU result data (from Z-register) early in the next controlword cycle.

.

.

The D-register data is sent to the SDBO assembler, where it can be routed to externals (via external bus-in) or to local storage.




CPU 96

C-REGISTER (CONTROL WORD DECODE)

C-REGISTER

The C-register contains 32 bits plus four parity bits. The purpose of the C-register is to decode the control word and provide control and gating of CPU functions. Once read out of control storage and gated to the C-register, the control word is decoded to determine:

- Word type
- CPU function
- CPU clock cycle and length

The functions of the bytes of the C-register are:

- C0 Define word type and format Branch High Address (M3 B4)
- C1 Specify A-source or destination (an external register or an A-local-storage address). Stat sets and/or special functions.
- C2 Specify a B-local-storage source or destination. Contains Mask or K values special Stat sets.
- C3 Specify next control-word address Branch Low Address (M3 B5).

The C-register is set by O-time of each control cycle.





C-REGISTER (CONTROL WORD DECODE) CPU 98

CPU CLOCK

CPU CLOCK OSCILLATOR

Timing for the 3145 is developed from a 22.222-MHz crystalcontrolled oscillator. This oscillator is fed to an oscillator control card, where it goes through one frequency divider stage and is controlled for distribution to each board clock. From this each board clock then develops six basic timing signals to time the CPU circuitry.

CPU CLOCK TIMING

This clock is a variable-cycle clock that is designed to operate 180-, 225-, and 270-nanosecond cycles, with each cycle having the further capability to extend by 22.5-nanosecond increments (referred to as pauses). This then allows (with one pause) 202.5 ns from a 180-ns cycle; 247.5 ns from a 225-ns cycle; and 292.5 ns from a 270-ns cycle. With two pauses, a cycle may be extended 45 ns as in the case of the Storage-1 Cycle Write. (270 becomes 315.) If the cycle after a Storage-2 cycle is a selector channel share cycle, the pause in storage-2 cycle is eliminated (270-ns cycle).

The CPU control word decodes determine cycle length. They provide 180-, 225-, and 270-cycle control signals that determine the cycle with which the clock should operate and the number of pauses the cycle should contain.

The CPU clock runs for one cycle under control of the clock start latch. The clock start latch has many input controls (start switch, set IC, CF clock start, etc.).

The clock consists of six latches operated in an overlapped configuration to produce six timing pulses.





L	90	180 225 247.5	90	STOR	AGE-1 CYCLE WRI	TE 315.0 m	
0-TIME	1-TIME	· · · · · · · · · · · · · · · · · · ·					
90 ns	90 ns FIME DELAY , 1-1	IME DELAY	0-TIME DELAY	0	45 90	135 180	0 225 315 45
90	ns	112.5 ns 2-TIME 112.5 ns					2)8
270-CYCLE	292.5 ns						
	90	180 27	0 292.5	90 I			
90 ns	90 ns IIME DELAY 1-T	2-11ME 112.5 ns ME DELAY 2-TIM	90 ns E DELAY				
90	ns 90	ns 112.5 ms					

CPU Clock Checks and Adjustments

Equipment Required

Tektronix* type 454 oscilloscope, or equivalent. Oscilloscope probes of equal length and equal attenuation,

Note: All measurements should be made with the displays centered on the scope face.

Oscillator

Oscillator frequency is 22.5 MHz. Symmetry and frequency are checked at 01AB2L2S05.



Clock

- The CPU clocks are initially synchronized at the factory and should be readjusted only when additional boards are installed. a feature is added to a board, or a clock card is replaced.
- Each clock has a programmable delay-line adjustment. Pin G10 of each clock card (6735) is the oscillator test point. To ensure oscillator synchronization during the following adjustment, sync negative on the G10 test point.

The oscillator signal should now be synchronized for the system. Check the oscillator test point on each board. If the signal is found to be more than ±2.0 ns out of sync, resynchornize the clock on the card that is out of sync.

1. To determine 'late clock' sync:

- a. Set rate switch to SINGLE CYCLE HARD STOP (CLOCK STOP indicator on).
- b. Ensure that zero delay is plugged in the clock cards in boards A-A1, A-B1, and A-C1.
- c. Oscilloscope settings
- (1) A sweep Time/Div = .02 us.
- (2) X10 multiplier on.
- d. Sync channel 1 (minus) on clock card at A-C1G2G10 and display the signal.
- e. Using channel 2 to display the G10 pins of the clock cards in A-A1 and A-B1 boards, determine the latest (in time) of the three clocks.
- f. Place the channel 1 probe on the latest of the three clocks. Channel 1 is now synchronized on 'late clock'.

Note: By swapping the input signals at the oscilloscope, verify that the oscilloscope is calibrated. The relationship between the two signals must not change. If signal relationship does change, use another oscilloscope before continuing with this adjustment.



2. With channel 1 sync and display on 'late clock', display all other clocks. If any clock is more than 1.0 ns earlier or later than 'late clock', change the programmable delay line for that clock to being it to within ± 1 ns of 'late clock' (negative pulse at G10).

CPU Clock Locations

CPU Clock Lo	ocation	• _ (.e.)	_note	3		
Gate A-C1G2	C3G4	B1C4	B3H4	A1K2	A3C4	
C2J2	C4E2	B2M2	B4K2	A2C4	A4Q4	
Gate B-A1C4	B1	C1M2	B3V3	C3J2		
		- -	note	4		

Oscillator Location:

Oscillator and oscillator control card location: A-B4A3 A-B2C2

1. These numbers are initial programmable delay settings.

Notes:

- 2. Select has a separate oscillator signal.
- 3. For IFA version 003 machine, clock is located in B1B4 socket on the A-Gate.

CPU CLOCK CPU 100

4. Clock-card position is feature-sensitive. Refer to the KC ALD Dages.

*Trademark of Tektroniz Inc.



na 13 Na

M, N, and MB-REGISTERS

M-REGISTER

- Addresses main and control storage,
- · Feeds storage address buses (SAB).
- Made up of M1, M2, and M3, which provide a 21-bit (plus 3 parity bits) storage address.
- M1, M2, and M3 address both main and control storage. Storage is read out on a doubleword boundary and stored on a word boundary.
- M3 bits 5,6, and 7, and the storage word being executed, provide the following selections for storage-word operations.

Store

Read

or Byte

Odd/Even word, Halfword, Word, Halfword, or Byte Note: Information may be stored under mask (any byte selected).

201 V

A 18.



M. N. and MB-REGISTERS

CPU 102

Setting M-Register for Storage Addressing

- M1, M2, and M3 are set from the ADR/ADJ circuits or B-Reg, . bytes 1 (bits 4-7), 2, and 3.
- M1, M2 may be forced to zero for direct main-storage addressing.

Setting M-Register for Control Storage Addressing

- M1 is set to zero for display purposes. A line address CTRL store is sent to the ECC board.
- M2 selects the control-storage module and is set from C2, trap circuits, or N2 (no module-switch function). M2 may be forced to FF for direct control-storage addressing.
- A module is defined as a group of 64 words in storage. This increment of storage has particular significance in the control storage addressing structure. The address in M2 selects the specific module. Any word in the 64 word module can be specified by the address in M3 bits 0-5.
 Most control word operations cause only M3 to be changed.
- M3 selects one of 64 words within the module selected by M2. M3 is set from C2 (K-adr STW), C3, or trap circuits.

NREGISTER

- Made up of N2 and N3.
- Backup register for control-storage addressing.
- N2 is set with the same information as M2 and is changed only when the control word being executed performs a moduleswitch function.
- N3 is set with the same information as M3.
- N is not changed when a trap occurs.
- When a trap occurs, the M-register is set to the trap address. The trap routine stores the contents of N (the N-Reg contains the next address that would have been used had the trap not occurred). At the end of the trap routine, M and N are restored to their original value so that the control-word sequence may continue as if there had not been any trap.
- N2 sets M2 for every control-storage word access except when a module switch occurs.

MB-REGISTER

- Made up of MB2, MB3.
- Set with the control-word address in M2, and M3 from M2 BFR and M3 BFR.
- When the CPU clock is stopped, MB contains the address of the last word executed.
- MB-reg output is available to the retry and backup circuits as well as the external assembler (word RTY).

Buffer Registers

- The M-buffer registers are an interim set of latches between the
- M-register and the MB-register. This allows cycle-to-cycle communications.
- The N-buffer registers perform a similar function.



	First C	ycle	Next Cycle		
	0-Time	1-Time	0-Time	1-Time	
Set/Reset					
g Set/Reset					
leg Set/Reset					
Set/Reset			li Second Second		

M-Reg

M-Bfr F

MB-Re

N-Bfr F



M, N, and MB-REGISTERS CPU 104





CONTROL REGISTERS CPU 106

CONTROL REGISTERS

DESCRIPTION

The control registers provide a means for maintaining and manipulating control information and are an extension to the EC PSW. The 3145 has sixteen 32-bit registers that are located in control storage at addresses F480 through F48F.

Two instructions, Load Control and Store Control, move data to and from the control registers. The Load Control instruction provides a means for loading control information from main storage into control registers; whereas Store Control permits information to be transferred from control registers to main storage. These instructions operate in a manner similar to Load Multiple and Store Multiple.

Details of the register assignments are contained in the sections that discuss the features using the control registers.

CONTROL REGISTER ASSIGNMENTS

0	System Control	Translate Control	External Interrupt Masks
1	Segment Table Length	Segment Ta	ble Origin Address
2	Channel Mas	ks	
3			Reserved
4			Reserved
5			Reserved
6			Unassigned
7			Unassigned
8			Monitor Mask
9	PER Event Masks	00000000	PER General Register Alteration Mask
0	00000000	PER S	itarting Address
1	00000000	PER E	Inding Address
2			Unassigned
3			Unassigned
4	Error Re Control	ecovery and Masks	
5	00000000	MCEL A	ddress





STANDARD FEAT

TIME-OF-DAY CLOCK

The time-of-day (TOD) clock provides a consistent measure of time suitable for elasped time and time-of-day indications. The cycle of the clock is about 143 years when started from zero as an elapsed time measure. To provide a consistent time-of-day indication, the zero point must be defined to a calendar date. IBM programming systems have established this date as January 1, 1960, 0 AM Greenwich Mean Time.

Setting the TOD clock on the basis of a synchronization signal given by the operator introduces errors in the fractions of a second. This error is usually of small consequence in defining the time relating to human reaction. The error does not enter elapsed-time calculations because the difference between two time readouts does not consider the initial setting of the clock. For many TOD applications, only the high-order 32 bits need be considered. Position 31 of the counter is advanced every 1.048576 seconds. Operation in this mode still requires entering some value for the TODL destination, or the clock does not start.

The clock is a binary counter with a two-word format (64 bits) numbered 0 to 63 corresponding to the bit positions of a fixedpoint number of double precision. Time is measured by incrementing position 51 of the counter every microsecond. Only the high-order 52 positions of the counter are used for this configuration. The remaining low-order positions are not used for time indication and are normally set to zeros except for three positions that define the status of the clock

The program is not signaled of an overflow condition when the counter is advanced to the point of carry from either position 1 or position 0. At the point of carry out from position 0, the counter goes to zero and continues to count from that value.

The clock can be inspected by the instruction store-clock. The current value of the clock counter is stored in main storage. The clock can be set to a specific value by the instruction set-clock. The operand specified by the instruction replaces the current value in the clock counter. The set-clock instruction can be executed only when the clock security switch on the system control panel is set to enable.

The operation of the time-of-day clock is not affected or inhibited by any normal activity or event in the system other than turning off the CPU power or running the TOD clock diagnostics. The clock runs when the CPU is in wait state, stopped state, or instruction step mode, and its operation is not affected by system reset or the IPL procedure.

The 3145 time-of-day clock stops when the CPU power-off switch is operated. Execute the set-clock instruction each time the system is started after powering up or after running the TOD clock diagnostics.

Physical Description

The time-of-day clock is driven by a 1-MHz oscillator feeding a binary-coupled trigger to produce a 1-MHz output. These circuits also develop a 75.46-KHz output that drives the interval timer. The clock counter functions as a binary-connected counter but is modified with a set of adder latches that allow holding the output of the basic latches until such time that the CPU sampling is complete. The counter can be loaded by destining the appropriate binary values to the TODL and TODH externals after executing a loading sequence. The counter starts to advance, immediately following the loading by gating the 1-MHz drive signal to the low-order position of the counter. The counter advance is checked by predicting parity and then comparing the predicted value with the parity generated from the counter. The counter output with parity bits for each byte is available to the CPU through the store clock instruction that causes externals TODL and TODH to be transferred to main storage.

The TODH and TODL externals may be called out as a destination at any time, but the contents of the clock are not changed unless TODL byte 3 bit 0 and the clock-run latch have been reset. The FTC (flush-through check) is blocked by the TODL byte 3 bit 0 to prevent error signals for this condition.

The control bits are associated with the clock readout to convey information to the user as to whether the clock value is a true measure of elapsed time since the last time the clock was set. These control bits are stored as the three high-order bits of byte 3 of the low-order word (TODL). Bit 0 indicates that the clock is running. Bit 1 indicates that the clock was set. Bit 2 indicates that an error occurred. During the processing of the store-clock instruction, the indicators control the condition code to be set.



32 Bit definition:

TODH Bits J-31 Binary Counter

40

TODL Bits 32-51 Binary Counter

- TODL Bits 52-55 Spare Counter Positions
- (forced to zeros)
- TODL Bit 56 Run Bit, This bit is set during the destining of the TOD word following the destining of the other TOD word. This bit must be reset by microcode with the security switch in the enable-set position to reset the TOD Run latch and enable the TOD clock hardware to accept the TODL or TODH as a destination. Power-on reset also resets this bit to allow the poweron reset routine to set the TOD clock to a consistent value.

63

- TCDL Bit 57 TOD Clock Security Switch. This bit on signifies that the security switch is in the enable-set position. This bit must be on to allow TOD Clock destinations, except during the power-on reset routine.
- Bit 58 Validity Latch. This bit on indicates that the TODI clock is valid. The TOD validity latch is set by the AND of TODL byte 3 bit () and TODL byte 3 bit 1). This bit is reset by resetting TODL byte 3 bit 0, power-on reset, or TOD clock error. This bit off lights the TOD invalid light on the console.
- TODL Bits 59-63 CPU Identification.

TODH may be displayed through EXT 36. TODL may be displayed through EXT 3E.

Clock Security Switch (TOD CLK)

The clock security switch (TOD CLK) provides an interlock with the set-clock instruction to guard against inadvertent change of the clock value. The switch is spring-returned to the secure position. When the switch is in the enable-set position, execution of the set-clock instruction sets the clock to the value of the designated operand. When the switch is in the secure position. the set-clock instruction does not change the value of the clock. The switch does not have any other effect on the operation of the clock.

Clock Validity Indicator (TOD CLK INVAL)

The clock validity indicator (TOD CLK INVAL) is used to indicate when the time-of-day clock current value is not a true measure of the elapsed time since the last time the clock was set. The validity indicator is turned off when the set-clock instruction is executed with the TOD CLK switch in the enable-set position and no exceptions are encountered. The indicator is turned on whenever the clock misses a time increment or stops. This may result from a power failure or a malfunction in the clock circuits. When the indicator lights for an error condition, the machine-check indicator is set and an interrupt is requested. If the clock is started by the power-on reset routine and used for elapsed-time indications, the indicator remains lighted because it is invalid as a TOD indication. In this case, the indicator being lighted does not mean that an error has occurred.

Error Detection

The time-of-day clock checks its advance operation by a check on the progressive parity conditions. Before the advance, the counter value of each byte is fed into a parity predictor circuit to develop the updated parity bits. These parity bits are compared with parity bits generated from the updated value in the counter. Any failure within the counter advance circuits results in a difference between the predicted parity bits and the resultant parity bits. The difference signal sets the TOD clock check latch. The latch output sets the MCKA3 bit 6 and resets the bit 2 and bit 1 latches as indicators. A class 3 machine-check interrupt is requested. The setting of the MCKA latch resets the TOD clock check latch. The reset of the bit 2 latch lights the TOD invalid indicator on the system control panel.



STANDARD FEATURES CPU 110

Clock-Setting Sequence

The clock is set to zero and started by the power-on-reset sequence. With this start the clock output can be used to indicate running time or elapsed time. Under this mode of operation, the TOD-Invalid indicator on the system control panel remains lighted because the clock output is not TOD.

To obtain time-of-day output, the set-clock instruction must be executed to start the clock at the current time. When the setclock instruction is executed, the correct TOD is assumed and the TOD Invalid indicator on the system control panel is not lighted. If a clock error is detected during operation, the indicator is lighted to warn the operator.

The set-clock instruction must have the TOD CLK switch on the system control panel held in the enable-set position to allow execution. If the switch is not operated, the clock value and its operation are not affected. The first step in the clock-set sequence is to reset TODL byte 3 bit 0 along with the 'clock run' latch and the clock controls. With the 'clock run' latch reset, the TODH and TODL externals can be destined. The value destined for TODL must have byte 3 bit 0 set to 1. When both load latches have been set, the TODL byte 3 bit 0 line is developed, the 'clock run' latch is set, and the 'clock set' latch is set. At this point, the clock is set and starts to run with the next 1-MHz pulse.

The power-on-reset start differs only in that the TOD CLK switch does not need to be operated, and the first TODL destined microstep is not required.

For microdiagnostic operation, the enable-set output of the switch is forced, but the set sequence is the same as for the setclock instruction.

TOD Clock Update Sequence

Because of the asynchronous operation of the TOD clock, the advance pulse from the 1 MHz oscillator cannot be used directly to advance the clock. The clock readout must not change during the A-register set/reset time. The CPU 90-135 time is gated with the 1-MHz oscillator to develop the advance pulse. If the CPU clock is stopped, the CPU oscillator provides the timing. The advance pulse sets the start-update latch whose output provides a series of delayed outputs that control error sample and advance of the clock.

TOD CLOCK UPDATE SEQUENCE



TOD Manual Set

This procedure is available for manual setting of the TOD clock. 1. Hold TOD CLK switch at ENABLE SET.

- 2. Store in EXT 3E hex 00. Hex 00 is stored into all 4 bytes of TODL. This value resets the run latch.
- 3. Belease TOD CLK switch.
- 4. Store in EXT 36 hex FF (see Note 1).
- 5. Hold TOD CLK switch to ENABLE SET.
- 6. Store in EXT 3E hex 80. This value sets the run latch.

7. Release TOD CLK switch.

8. Display either TODH (EXT 36). Observe that bit 31 increments approximately every second, or

> TODL (EXT 3E). Observe that counter is rippling much faster.

Note 1: Any value put into switches A and B may be stored. This value is propagated to all 4 bytes (example: hex 12 stored from switches A and B is stored as 12 12 12 12). FF is inserted because the first time the clock is incremented, the counter flips to zero and starts from there.

> Compute the B1D1 address specified in the instruction. Test and branch to the B2 op-code routine (GGB2). GGB2

Decode SI-format insturction.

TOD Clock Instructions

storage.

operation code.

both instructions is:

Normal I-cycles.

The time-of-day clock has two instructions:

Store Clock used to enter the current clock value into main

Both instructions are SI-format instructions modified so that

byte 2 is an extension of the operation code (byte 1) instead of

Both instructions are decoded as operation code B2 in the

GAAI routine. The operation branches to the GGB2 routine to

is made when required. Then the operation is branched to the

routine of the function specified. The common data flow for

START

decode and validate the modifier. A test for privileged operation

the immediate operand. Byte 2 specifies the exact function of the

Set Clock used to set the initial time.

GAAI

Test for a valid code must be of the B20x format. The last four bits of the modifier select a decode byte from the 16-byte table. If the high-order bit of the decode byte is a 1, the instruction is a privileged operation and the CPU must be in supervisor mode (PSW bit 15 = 0). ______ If a privileged operation and not in supervisor mode, branch to the privileged operation check routine in GICM. If not a privileged operation or if in supervisor mode, use the decode byte as an address modifier and branch to the instruction execution routine.

BRANCH

Set-Clock Instruction

[\$1] SCK D1 (81)

82	04	81	D	1	
~	•		20		1.

B2 = Operation Code 04 = Set-Clock Function

B1D1 = Storage address of an eight-byte field. Must be on a doubleword boundary. Bits 52-63 of the field are ignored and are not used in the clock value.

.31

The set-clock instruction is a privileged operation used to place a value into the time-of-day clock. The location of the value is specified by the B1D1 portion of the instruction. The implied length of the value is eight bytes (two words). The address must be located on a doubleword boundary. Only the high-order 52 bits of the doubleword are used to set the clock counter. The remaining bits (52 to 63) are ignored by the operation in setting the clock value.

The value in the time-of-day clock is replaced by the designated value if the set-clock instruction is executed while the TOD CLK switch is in the enable-set position. If the TOD CLK switch is in the secure position when the set-clock instruction is executed, the value in the clock is not replaced and the condition code is set to indicate:

Condition code settings:

- 0 = Clock value set.
- 1 = Clock value secure (TOD CLK switch in secure position; therefore, clock value was not changed).
- 2 = Not used by the 3145.
- 3 = Not used by the 3145.

START	
GGCS	
Test B1D1 address for a doubleword boundary. Branch to specification check in GICM if not a doubleword boundary.	
Store high-order word of main-storage double-word in local storage.	
Store low-order word of main-storage doubleword in local storage. Reset byte 3 to zero.	
Attempt to reset counter with TODL destination of the low-order word (this include: byte 3 bit 0 and the clock-run latch).	
Test TOD1, byte 3 bit 0 to determine the TODL CLK	

Test switch setting. Bit 0 = 0 if in enable-set position. Bit 0 = 1 if in secure position.

If the switch is in the secure position, set condition code 1 and return to I-cycles without setting the clock.

If the switch is in the enable-set position, set the low-order word byte 3 bit 0 to 1 to set the clock control bit and set the clock-run latch when the word is destined to TODL.

set TODH from the high-order word in local storage.

Set TODL from the low-order word in local storage. Clock starts with next 1-MHz oscillator pulse.

Set condition code 0 and return to I-cycles.

END

e-Clock	Instructio

Store-Clock Instruction STCK D1(B1) (SI)

		· · · · ·			 -	
B 2	05	B1		D ₁		
)	8	16	20		31	

B2 = Operation code 05 = Store-clock function B1D1 = Storage address of an eight-byte field. May be located on a byte boundary. Bits 52-63 of field are:

52-55 = Set to zero. 56-58 = TODL Ctrl Bits 0-2. 59-63 = CPU Identification bits.

The store-clock instruction is used to place the current time-ofday clock value in the eight-byte field of main storage designated by the B1D1 portion of the instruction. The 52-bit clock value stores in the high-order of the doubleword assignment. The loworder byte stores the three clock control bits and the five-bit CPU pluggable identification. If the clock value is invalid, the doubleword is stored with all zeros.

Condition code settings:

0 = Clock in set state. 1 = Clock in not-set state. 2 = Clock in error state. 3 = Not used by the 3145.

re TODH in high-order word TODL in low-order word TODH for no change since DL is not latest, branch back n.	l local store. local store. store in local storage. If
TODL in low-order word	local store.
TODH for no change since DL is not latest, branch back n.	store in local storage. If
ورجيسي بينور فتكر أنبيت البكار عيوارين	to store TODH and TODL
TODL byte 3 bits 1 and 2 1 = 1 error state (invalid). 2 = 0 not-set state (not valic 2 = 1 set state (valid TOD).	for clock status. J TOD).
lock is in error state, set con Il store words to zero and br n storage.	idition code 2. Set both anch to GFST to store in
lock is in set state, set condi ST to store the clock words	ition code 0 and branch to in main storage.
lock is in not-set state, set connection of the	ondition code 1 and ock words in main storage.
T	
ddress is on word boundary, rations and return to I-cycle	, take two store-word s.
	dary, shift the data to align







CT225 Parity Parity Predict Parity Asm for Byte 1 CT226 **TOD Termination** CIRCUIT CARD LOCATION: A1R2

LOGIC/ALD PAGE:

CT311 TOD Ctrs 20-21-22 IFA/TOD Asm Byte 2 Bits 4-5-6 CT312 TOD Ctrs 23-24-25 IFA/TOD Asm Byte 2 Bit7 IFA/TOD Asm Byte 3 Bits 0-1 CT313 TOD Ctrs 26-27-28 IFA/TOD Asm Byte 3 Bits 2-3-4 CT314 TOD Ctrs 29-30-31

IFA/TOD Asm Byte 3 Bits 5-6-7 CT315

```
Parity Asm for Bytes 2 and 3
Gate Buffers
```

```
CT316
      Parity Predict for TOD Bytes 2 and 3
CT317
```

TOD Asm Byte 3 Bits 3-4-5-6-7 **CPU Identification Number**

STANDARD FEATURES CPU 114

INTERVAL TIMER

The interval timer provides program interruption on a programcontrolled time basis. Interval timer applications include:

- Job accounting
- Monitoring for perpetual program loops
- Time stamping
- Polling at timed intervals

The storage area allocated for the interval timer feature is in main storage locations 50-53 hex. If the interval timer switch is set to NORM (normal), any value stored at this location is decremented by the hardware.

The program being processed can be interrupted by an external interruption (if PSW bit 7 and control register 0 bit 24 are on) when the interval timer word changes from a positive to a negative value. The interruption is identified by the appropriate external interrupt register bit,

Description

Has a 32-position counter

- Stores last timer value in MS 50
- Contained on 3 cards:

A-B2K4 Controls ALD CH031-032 A-C1T2 and A-C1U2 Timer

- ALD CH211-225
- Uses same oscillator as TOD clock (High Resolution Timer 75.46-KHz oscillator)

Interval Timer Operation

Enable or disable timer

Decrement timer

Set timer

Read timer

Display Timer

Interval timer interrupt



Enable Timer

To enable the interval time

- 1. Set interval timer switch to NORMAL.
- 2. Set rate switch to PROCESS, and
- 3. Execute a return word (RTN with Br Lo = 111) to set the timer run latch.

Decrement Timer

The hardware circuits gate a signal to the counter. This signal is synchronized with the oscillator which has a 13-us, time interval. At 45 to 135-time in the CPU clock cycle at the start of the 13-us. Interval counter position 31 is decremented. This update is inhibited each time the interval timer is being set to a new value.

Position 23 of the counter is decremented every 3.3 ms. To obtain this degree of resolution, the ability to store in byte 3

(positions 24-31) is inhibited by the timer hardware. Therefore, these positions are used functionally to assure that position 23 is updated each 3.3 ms by a signal which is developed every 13 us.

Whenever an interrupt occurs, the timer value is stored in main storage location 80. If interrupts occur at time intervals greater than 2.048 ms the TOD clock carry-out of bit position 41 forces a branch to the section of microprogram (GICM) that transfers the timer value to main storage location 80.

The interval timer is disabled when:

1. The interval timer switch is in DISABLE

2. The rate switch is not in PROCESS

3. A timer error occurs.

Set Timer

The timer value is set into the hardware and main storage location 80 at 0 to 45-time of the storage 2 cycle of a storage word that specifies:

1. A not k-addressable storage word, and

2. A store word into location 80.

Read Timer

The contents of the counter cannot be displayed directly. To enable the timer value to be checked the timer contents are transferred by the 370 microprogram

Display Timer

The manual display of location 80 shows the timer value of the last update.

Interval Timer Switch

NORMAL

This switch position enables the interval timer control circuits, which allows the herdware to be decremented continually. DISABLE

This switch position disables the interval timer control circuits. Regardless of the interval-timer switch position, a store/display of location 80 displays only the contents of main storane loca-

of location 80 displays only the contents of main storage location 80.

Interval Timer Interrupt

Once enabled, the timer decrements continually. The instant that the value changes from positive to negative, an interrupt condition is indicated. The interrupt routine (GICM) tests for this timer value change in LS Y (LS 16). When LS 16 bit 0° changes from zero to one, the microprogram branches when Y0 bit 0=1 **I** and sets **G** the External Interrupt register bit 0 (Ext 12, byte 0 bit 0) on.

During program execution, whenever both PSW bit 7 and MSKA bit 0 are on, the timer interrupt request is gated to the interrupt latch. The system uses this to execute the interrupt handling routine.



Sample Routin	e:	a an an	
Label	Next Label	Statement	Comments
C TIMR 00 TIMR 10	TIMR 1, BO B TIMR 1, 0 MISC B1	PX0, OE, 52 VC = 0 + K50 RDW Y DM, 50 Y3 = X3 STW Y DM, 50 Y0 EXTINT, OR, K80 SYS1	Set P for interrupt routine V = 0000050 Read timer hardware Read location 80 [°] Preserve byte 3 Update location 80 Current value negative? Set timer interrupt request Continue GICM
TIMR 11	TIMR BO, 0	XO	Was last value negative?

Notes: 1. These statements A are found at the beginning of the GICM routine.

 The timer update D occurs only when either a system interrupt or a TOD forced interrupt occurs.



OS/DOS COMPATIBILITY

Introduction

- Consists of the OS/DOS emulator program and the hardware and microprogramming needed for execution.
- Two new instructions: Execute Local (EXL) and Adjust CCW String (ACCW), are used by the emulator-program.
- The DOS Emulator and the DOS system being emulated are located in main storage above the OS area.
- The minimum storage area needed for the DOS emulator and the DOS system is 38K bytes.
- The OS/DOS emulator operates in the same manner as any OS job.
- When operating in local mode (DOS programs being executed), all addresses pertaining to the DOS area are adjusted by the address-adjustment hardware.
- When local mode is terminated, addressing is performed in the standard manner.

Refer "OS/DOS Functional Units", and "Real Address Computation Example" for an explanation of the addressadjustment hardware.

Address translation is needed for DOS emulation because the addresses of the DOS supervisor (SV) are basically fixed and the DOS supervisor is located in an area of storage not normally used by DOS SV. All references to the fixed addresses of the DOS SV must be adjusted to reflect the real location of these fixed areas.

The EXL instruction operates with the LEX List, which is a table that the emulator program loads before the execution of the DOS area. This list is located in the emulator area and is used to handle entry to and exit from the local mode of operation.

The ACCW instruction operates with the Adjust CCW list (ACCW List). The ACCW list is loaded and maintained by the emulator program for use in the adjustment of CCW data addresses. The OS/DOS emulator and the DOS system being emulated (DOS supervisor and up to three processing program partitions) execute together in an MFT partition or MVT region, which must be a minimum of 38K. The OS/DOS emulator program and tables require 22K plus another 4K if I/O staging is used. Additional OS/DOS emulator program storage may be required, depending on the I/O devices used. Up to ten I/O devices are supported in 22K, and 250 bytes are required for each additional device. The I/O staging requirement of 4K supports unblocked reader, printer, and punch records and residence of the required QSAM routines in the OS/DOS emulator partition or region.

The DOS system being emulated can be 16K, 24K, or 32K and up, in 4K increments. The OS/DOS emulator is scheduled to operate in the same manner as any other OS job. Several OS/DOS emulator jobs can execute concurrently with OS jobs if enough I/O devices and processor storage are available. In addition, the Model 145 OS 1401/1440/1460 and 1410/7010 Emulator programs can execute concurrently with the OS/DOS emulator if enough resources are present.

EXAMPLE STORAGE ASSIGNMENT for 256K MODEL 145



STANDARD FEATURES

CPU 11

OS/DOS Functional Units

Translate Look-Aside Buffer (TLB)

- Eight 26-bit registers contain the local and real addresses used during the accessing of the local area when operating in local mode.
- The local address occupies bytes 0 (bits 0-7) and byte 1 (bits 0-3) of the TLB.
- The local address is gated to the registers from PAA.
- The real address occupies byte 2 (bits 0-7) and byte 3 (bits 0-3) of the TLB.
- The real address is gated to the registers from EBI.
- The register to be loaded is addressed by the LRU.
- The TLB can be displayed through EXT 2E with switch H selecting 1 of 8 registers.

Least Recently Used (LRU) Matrix

- The LRU is an address matrix that keeps track of the use of the translate look-aside buffer.
- The LRU addresses the least recently used translate look-aside
- buffer whenever a computed real address must be loaded.
- Composed of 28 latches and associated circuits.
- The LRU is reset to zero before the OS/DOS operation.
- The LRU can be displayed through EXT 08 byte 2,

When a mismatch occurs between the local address portion of the PAA and the local address portion of the TLB, the LRU determines the register to be loaded with the computed real address. In the GGST microroutine, the computed real address is loaded into TLB bytes 2 and 3 and the local address causing the mismatch is loaded into TLB bytes 0 and 1 addressed by the LRU.

The status of the LRU is changed each time a match occurs between the local address portion of the PAA and the local address from the translate look-aside buffer. This constant changing assures that the least recently used register is addressed when a mismatch occurs, Refer "LRU Operation Example," 化温度 建铁铁 中心的

Match Circuits

4.80

11

 $i_{\rm O}$

- Perform the comparison of the local address portion of the PAA and the local address portion of the translate look-aside buffer bytes 0 and 1.
- Output of the match circuits sets and resets specified combinations of the LRU,
- The match circuit can be displayed through EXT 08 byte 3. When displaying the match circuits, the PAA must contain valid information.

There is a match circuit for each translate look-aside buffer. When a match occurs, the corresponding LRU row is set to ones and the corresponding column is reset to zeros. The resulting status of the LRU provides the means of addressing the least recently used register.



LRU Operational Example

PAA local address (00A)

FIRST STORAGE ACCESS ATTEMPT









LRU 7 6 5 4 3 2 1 0 0 1 1 1 0 0 0 0 1 1 1 1 0 0 0 1 1 1 0 1 2 3 1 1 1 0 1 1 1 4 1 1 5 6 0 7

6 now least recently used (row 6=0s and column 6=1s)



Assume:	Storage access attempted for local address 0A000.
	Adjustment factor = 14000
	Translate look-aside buffers are set to values indicated.
	Table buffer Regs are set to values indicated.
Objectives:	Provide a real address to the M-register to access the local area specified as address 0A000. When the storage access is attempted in local mode
	the local address portion of the PAA and the local
	address portion of the translate look-aside buffers are compared.
	A no-match condition results from the match
	circuits. This no-match condition causes a trap
	to control-storage address D80C. The GGST
l in an	microroutine is executed. The GGST microroutine:
	Computes the real address by adding the local
	address and the adjustment factor,
	Re-executes recently used of the translate
	look-aside buffers with the real and local
· · · ,	address.
	Re-executes the storage word that caused the mismatch.
	Re-executing the storage word causes a match to
	occur from the number 4 TLB. the number 4 TLB
	was loaded in the GGST microroutine.
	The match line from the number 4 register sets
	row 4 and resets column 4 of the LRU. The status

of the LRU now indicates that TLB 6 is the least recently used. Should a mismatch occur on the next storage access, the computed real address is loaded into TLB 6.

Translate Look-Aside Buffer

PAA local address (00A)

New Instructions for CordOS Emulator

- The new instructions are Execute Local (EXL), and Adjust ** CCW string (ACCW).
- The Op code for both these instructions is B2.
- The immediate byte determines which of the two instructions is to be executed.

Execute Local

B2 OE B1 D1 (EXL)

 This instruction addresses the LEX list, performs certain initialization functions, and sets local mode for system operation.

When the EXL instruction is executed, the condition code, program mask, and instruction address in the current PSW are replaced by values from the LEX list. General registers 14 and 15 are loaded from the LEX list and the CPU is placed in Local mode.

During the execution of the EXL instruction, the modified PSW is not checked for program interruptions. Any such checks occur as part of the next instruction execution.

Condition Code

Upon completion of the EXL instruction, the condition code is set according to the condition code loaded from the LEX list.

Program Interruptions

Addressing:	The address of the LEX list is invalid. The address
물리 가지 못하는	formed by the addition of the origin address and
	the local address exceeds the maximum address
	allocated to the emulator program. The operation is suppressed.
Operation:	The instruction is not installed. The operation is suppressed.
Protection:	The LEX list is protected for fetching or storing.
	The operation is suppressed.
Specification:	The first operand address does not specify a 64
	byte boundary.
	The origin address is not a multiple of 4096.
	The local limit address is not one less than a
날 옷 있지?	multiple of 4096.
병원 승규는 물건한	The operation is suppressed.
Special	그는 그는 것을 가장 것을 가지 않는 것을 가지 않는 것을 가지 않는 것을 가지 않는 것을 하는 것을 수가 있다. 이렇게 나는 것을 하는 것을 하는 것을 하는 것을 하는 것을 하는 것을 하는 것을 수가 있다. 이렇게 나는 것을 수가 있는 것을 수가 있다. 이렇게 나는 것을 수가 있는 것을 수가 있다. 이렇게 나는 것을 수가 있는 것을 수가 있다. 이렇게 나는 것을 수가 있는 것을 수가 있다. 이렇게 나는 것을 수가 있는 것을 수가 있다. 이렇게 말 하는 것을 수가 있는 것을 수가 있다. 이렇게 말 하는 것을 수가 있는 것을 수가 있다. 이 것을 것을 수가 있는 것을 것을 수가 있는 것을 것을 것을 것을 수가 있는 것을 것을 것을 것을 수가 않았다. 것을
Operations:	If the EXL instruction, the ACCW instruction,
	the Monitor Call instruction, or the emulator
	- transformet and the second se Second second se Second second s Second second sec

the Monitor Call instruction, or the emulator instruction is encountered while in local mode, the operation is suppressed. The interruption is reflected in the LEX list of the program that placed the CPU in local mode.

A privileged operation is any privileged instruction encountered while in local mode.

Local List Format and Definition Updates the current PSW when the EXL instruction Bytes 4-7 is executed. The instruction address in bytes 5-7 is the address of the next instruction within the DOS area. Whenever local mode is terminated by an interruption, this area is updated. The ILC field is Bytes 0, 1 Reserved for emulator program use. This area is unpredictable when local mode is terminated by an not addressed by the EXL instruction. asynchronous interruption. Upon termination of local mode by a program or Bytes 8-11 The value in this field is loaded into general register Bytes 2.3 supervisor call interruption, the 16-bit interruption 14 when the EXL instruction is executed. When code describing the interruption is placed in this local mode is terminated by an interruption, the current contents of general register 14 are stored field. into this field. Bytes 12-15 The value in this field is lorded into general register 15 when the EXL instruction is executed. When local mode is terminated by an interruption, the current contents of general register 15 are stored into this field. Bytes 16-19 The address contained in bytes 17-19 correspondent to the address adjustment factor and points to the Programming Use Rutes 0.3 Interruption Code zero address of the emulated DOS supervisor. This address must be a multiple of 4096, or a specifica-Prog Local Instruction Address Bytes 4-7 tion interrupt occurs. The high-order byte of this Mask field is reserved and should be set to zero. Byte 8-11 General Register 14 Bytes 20-23 The address contained in bytes 21-23 specifies the upper address of the emulated environment. The **General Register 15** 12-15 Byte address must specify one less than a 4096 boundary, or a specification interrupt occurs. The Reserved Bytes 16-19 **Origin Address** high-order byte of this field is reserved and should be set to zero. Byte 20-23 Reserved Local Limit Address Bytes 24-27 When local mode is terminated by a program or supervisor call interruption, this address points to Last Instruction Address 8vtm 24.27 Reserved the instruction causing the interruption. This address is within the boundary of the emulator Byte 28-31 Reserved SVC Interruption Address program. If the instruction causing the interruption was the object of an Execute instruction, the 32-35 Reserved Program Interruption Address Bytes address placed in this field would be the address of the Execute instruction. Bytes 36-39 Reserved Asynchronous Interruption Address The contents of the address field are unpredictable when local mode is terminated by an asynchronous interruption. The high-order byte of this field is set to zero. Bytes 28-31 When local mode is terminated by a supervisor

call interrupt, the address located in bytes 29-31 is placed in the PSW. This address is within the boundary of the emulator program. The highorder byte is reserved and should be set to zero.

When local mode is terminated by a program

interrupt, the address located in bytes 33-35 is

placed in the PSW. This address is within the

boundary of the emulator program. The high-

order byte is reserved and should be set to zero.

Bytes 32-35

Bytes 36-39 When local mode is terminated by an asynchronous interruption (external, I/O, or machinecheck), the address located in bytes 37-39 is placed in the PSW. The high-order byte is reserved and should be set to zero. The adjusted PSW is stored in the corresponding low storage old PSW and an interrupt is taken to the OS supervisor. STANDARD FEATURES CPU 120

EXL EXAMPLE: EXECUTION and LOCAL MODE OPERATION START





register for a main-storage access and updates the LRU matrix.

Adjust CCW String



• The operand address designates the Adjust CCW (ACCW) list, * *

 With the information from the ACCW list, this instruction addresses CCWs and performs adjustment on the data addresses of the CCWs.

The ACCW instruction interprets successive doublewords as CCWs and adjusts their data addresses by algebraically adding the adjustment factor to them. This process continues until:

The last CCW adjusted did not specify chaining, or

A CCW whose command code specifies TIC has been adjusted, or

A CCW whose data address points outside the emulated environment is encountered, or

The address of the next CCW is outside the limits of the emulated environment or does not specify a doubleword boundary.

Any of these conditions terminates the instruction and sets the proper condition code to specify the reason for termination.

When the ACCW instruction is completed, the address of the last CCW adjusted +B is stored in bytes 17-19 of the ACCW list for condition codes 0, 1, or 2. For condition code 3, the address stored is CCW + 0. If data chaining was in progress, the command code and the address of the CCW containing the command code are set in the operation byte and operation pointer fields, respectively.

If the last CCW adjusted specified transfer in channel, bytes 21-23 of the ACCW list contain the unadjusted data address from the TIC CCW. If the TIC CCW is encountered in a data-chaining sequence, the operation byte and operation pointer of the ACCW list contain the values set from the first CCW of the chain. When the TIC is not data-chained, the operation byte in the ACCW list is set to zero. The CCW address field in the ACCW list is set to the address +8 of the TIC CCW.

Condition Code

- 0 End of the CCW string. The last CCW adjusted specified neither data chaining nor command chaining.
- 1 A TIC CCW was the last CCW adjusted.
- 2 An adjusted data address was encountered that fell outside the
- area of the emulated environment. 3. The address of the naxt CCW to be adjusted did not specify a
- doubleword boundary or fell outside the area of the emulated environment.

Program Interruptions

- Addressing: The address of the ACCW list is outside available storage. The operation is suppressed.
 - The address of a CCW is outside available storage. The operation is terminated.
- Operation: The instruction is not installed. The operation is suppressed.
- Protection: The ACCW list is protected for storing or fetching. The operation is suppressed.
 - A CCW is protected for fetching or storing. The operation is terminated.
- Specification: The first operand address does not specify a 64byte boundary; the signed adjustment factor is not a multiple of 4096; the local limit address is not one less than a multiple of 4096. The operation is suppressed.

Special

Operation: The ACCW instruction was encountered while in local mode. The operation is suppressed. The interruption is reflected to the program that placed the CPU in local mode by an address in the local mode or by an address in the local list.

Adjust CCW List Format and Definition

이 이 집에 관객을 하는 것이 가지 않는 것이 같아.

		والمتحدية والمتحد والمحمول والمتحد والمحمول والمحمول والمحمول والمحمول والمحمول والمحمول والمحمول والمحمول والم	Uy aug +1
Bytes 0-3		Signed Adjustment Factor	
Bytes 4-7		Local Limit Address	
Bytes 8-11		Reserved Operation Byte	
Bytes 12-15		Operation Pointer	
Bytes 16-19		CCW Address	Bytes 8-11
Bytes 20-23	Meducarian	TIC Data Address	

Bytes 0-3 The signed binary number located in this field is added to the data address of the CCW addressed by bytes 16-19 of the ACCW list. The 24 low-order bits of the result are set into the data address field of the CCW. The CCW data address, which is local to the emulated environment, is compared against the local limit address. If the comparison indicates that the local address is above the local limit address, a program interruption occurs.

Bytes 4-7

Bytes 12-15

The address contained in bytes 5-7 specifies the upper address of the emulated environment. The address must specify one less than a 4096 boundary, or a specification interrupt occurs. The local limit address is compared with the local CCW address, and the extreme local address of the storage area defined for each CCW by the data address, command code, and unit, to assure that the local address is within the limits of the storage area assigned to the emulated environment.

The operation byte (byte 11) carries the command code for CCWs that are data-chained. The operation byte is set to zero if the CCW being processed is not data-chained. When the operation byte is fetched from the ACCW list, the high-order bytes of this field are ignored. When the operation byte is stored in the ACCW list, the high-order bytes are set to zero. A nonzero operation byte encountered upon initiation of the instruction indicates that the first CCW to be adjusted is part of a data-chained sequence.

This field contains the address of the CCW that originated the operation byte for the last non-TIC CCW adjusted. When this address is fetched from the ACCW list, the high-order byte is ignored. When the address is stored in the ACCW list, the high-order byte is set to zero.

Bytes 16-19 The address contained in this field is the address of the first CCW of a string when the ACCW instruction is encountered. When the ACCW is completed, this address points to the doubleword above the last CCW adjusted when terminated with condition 0, 1, or 2. When terminated with condition 0, 1, or 2. When terminated with condition code 3, the address points to the CCW causing termination. When this address is fetched from the ACCW list, the high-order byte is ignored. When the address is stored in the ACCW list, the high-order byte is set to zero.

Bytes 20-23 Contains the unadjusted data address from the CCW whose command specifies transfer in channel. The high-order byte is set to zero.



The operation pointer contains the address of the CCW that provided the operation byte.

Each CCW in the chain is addressed from the ACCW list. The data address from the CCW is added with the signed ad-

justment factor from the ACCW list. The local CCW address is compared with the local limit address. If the comparison indicates a valid local address, the operation continues. If the comparison indicates an invalid local address, the operation is terminated and a condition code of 3 is set. The extreme local address of the storage area defined for each CCW by the data address, command code, and count are compared with the local limit address and zero. If the comparisons indicate a valid local storage area, the adjusted data address is placed in the data address field of the CCW. If the address compare is invalid, the operation is terminated and a condition code of 2 is set.

The CCW address in the ACCW list is updated +8 to point to the next CCW to be adjusted. When the last CCW in the string has been adjusted, the CCW address points to the next sequential doubleword.

STANDARD FEATURES CPU 126

START GHYP OPRBND 11 2 RDW Q ADJ, V + 4 GAAI (I Cycles) Place signed adjustment GHYP / OPRBND 01 V now points to the factor in Q-Reg. local limit address Store local list Decode 82 Op. in the ACCW list. pointer at FF28. First Operand address set in Set local list V-register. pointer to the RDW Y ADJ, V + 4 1982 A.S. E origin location. Place local limit RDWQ ADJ, V+4 V now points to the address in Y-Reg. V now points to local operation byte field GHYP OPRBND 11 limit address location in the ACCW list. GGB2 INVOP 01 Place origin in the local list. address in Q-Reg. Set the Q-Reg to the proper Place local limit address for address in Y-Reg. RDW Y ADJ, V +4 entry to the Q = 0000XX38 for EXL V now points to last GHYP routine. Q = 0000XX3C for ACCW instruction address Return using location in the local list. Q as link Reg. Check addresses. If valid, continue; if not, take specification (XX38) (XX3C) exception and branch to GICM for Interrupt. GHYP HYPERS GHYP RCCWS Yes Local Local Yes Mode Mode ACCW ACCW EXL or EXL No No **Special Operation** GHYP RELBND :0 GHYP Exception; go to RELBND 10 GICM routine for Check the sum of ۰. Do CCW adjust as interrupt. described for ACCW the local limit instruction. address and the origin, It valid, continue; if not, take address GHYP DRI exception and . 0 branch to GICM 1st for interrupt. Yes Ňc operand Set local pointer address valid back to start of local list. Set registers for EXL local execution. Specification ACCW EXL OR Set local mode and Exception; go ACCW return to I-cycles. to GICM for interrupt. 2 1

EXL and ACCW Instruction Execution

Interruptions

Any interrupt removes the CPU from local mode.

- All synchronous interrupts that occur while in local mode are handled by the emulator program.
- All asynchronous interrupts that occur while in local mode are first handled by the OS supervisor.

If a supervisor call, program, External, I/O, or recoverable machine-check interruption occurs while the CPU is in local mode, the following actions take place:

The 16-bit interruption code associated with the supervisor call or program interruption is stored in the interruption code field of the LEX list. The contents of this field after an external, I/O, or machine-check interruption are unpredictable.

The ILC, CC, program mask, and instruction address of the current PSW are stored in the bytes 4-7 of the LEX list. The value of the ILC after an asynchronous interruption is unpredictable.

The current contents of general registers 14 and 15 are stored into bytes 8-15 of the LEX list.

If the interrupt is a supervisor call or program interrupt, the local address of the instruction causing the interruption is stored into bytes 25-27 of the LEX list. If the instruction causing the interrupt was the object of an Execute instruction, the local address of the Execute instruction is stored.

The address of the corresponding interrupt (SVC, program, or asynchronous) is loaded into the current PSW from the LEX list. If the interrupt is an asynchronous interrupt, (I/O, external, or machine-check) the adjusted current PSW is stored in the corresponding low storage old PSW and interrupt is then sent to the OS supervisor.

The CPU is removed from local mode,

LEX LIST AFTER PROGRAM INTERRUPT

Bytes	0-3	Program	Program Interrupt Code
ytes	47	i C Program L C Mask	Local Instruction Address
tes	8-11	Gen	ieral Register 14
tes	12-15	Gen	ieral Register 15
rtes	16-19	Reserved	Origin Address
/tes	20-23	Reserved	Locef Limit Address
	24-27	Reserved	Addr of instruction causing interrupt or execute address
185	28-31	Reserved	SVC Interruption Address
yt es	32-35	Reserved	Program Interruption Address
ytes	36-39	Reserved	Asynchronous Interruption Address

LEX LIST AFTER SVC INTERRUPT

tes	0-3	Program	nming Use	SVC Interrupt Code							
tes	4-7	1 C Program L C Mask	Locai	Instruction Address							
tes	8-11		General Register 14								
es	12-15		General Register 15								
65	16-19	Reserved	Urigin Address								
es	20-23	Reserved	Local Limit Addr	ess							
les	24-27	Reserved	SVC or Execute A	ddress							
tes	28-31	Reserved	SVC Interruption	Address							
tes	32-35	Reserved	Program Interrupt	tion Address							
tes	36-39	Reserved	Asynchronous Int	erruption Address							

When the interruption is a specification exception due to an odd address, the ILC is unpredictable. The Last instruction field

contains the odd address.

B١

Bv

R.

Ru

LEX LIST AFTER AN ASYNCHRONOUS INTERRUPT

rtes 0-3	Program	ming Use	UNPREDICTABLE
rtes 4-7	*I C Program C C Mask	Local Instruction Address	
tes 8-11	G	ieneral Register 14	
tes 12-15	G	ieneral Register 15	
tes 16-19	Reserved	Origin Address	
tes 20-23	Reserved	Locał Limit Address	
tes 24-27	Reserved	UNPREDICTABLE	
tes 28-31	- Reserved	SVC Interruption Address	
tes 32-35	Reserved	Program Interruption Address	
rtes 35-39	Reserved	Asynchronous Interruption Addre	55

*ILC is unpredictable.

SVC INTERRUPTION EXAMPLE

	이가 있다. 김 관광 같은		Store SVC interrupt Local mode in	ion code. s reset by the interruption				
						c	urrent PSW	
Bytes	0-3	Programming Us	e Interruption Code			ILC CC	Prog Mask Address	
Bytes	4-7	L C Program C C Mask	Local Instruction Address					- C
Bytes	8-11	Ge	eneral Register 14			Gen Reg 14		LOCAL AF
Bytes	12-15	Ge	eneral Register 15			Gen Reg 15		8VC
Bytes	16-19	Reserved	Origin Address			f saat in t		
Bytes	20-23	Reserved	Local Limit Address					
Bytes	24-27	Reserved	Last Instruction Address	*****	Address of SVC			
Bytes	28-31	Reserved	SVC Interruption Address					
Bytes	32-35	Reserved	Program Interruption Address	\neg				
B	26.20	Received	Asynchronous Interruption Address	This addres	is points to the SVC handling			

Once the DOS return address has been computed, an EXL instruction is executed, local mode is set, and a return is made to an address in the DOS SV.

The DOS supervision erates in problem state. When a Start I/O instruction (privilegeo instruction) is executed in the DOS supervisor, the DOS emulator intercepts the program interrupt causing the following to be performed by the 3145 microprogram (Emulator Interrupts GHVI).

- 1. The privileged operation interruption code is set into bytes 2 and 3 of the LEX list.
- 2. The selected current PSW instruction address is loaded into bytes 4-7 of the LEX list.
- 3. The address of the Start I/O instruction is stored into bytes 25-27 of the LEX list.
- 4. The current values of general registers 14 and 15 are stored into bytes 8-15 of the LEX list.
- 5. The program interruption address, bytes 33-35 of the LEX list, is placed into the current PSW.
- 6. The CPU is removed from LEX Mode.

А

в

D

Ε

- The DOS emulator commences instruction execution at the the address specified by the program interruption address and performs the following:
- 1. The ACCW instruction is executed to adjust the CCW data addresses to their values in the emulator program.
- C 2. Control is transferred to the OS SV to handle the Start I/O.

When the OS SV has handled the Start I/O, a return is made to the DOS Emulator. The EXL instruction is then executed to restore information prior to restarting the DOS SV at the point of interruption. Refer to "EXL Example: Execution." Upon completion of the I/O operation, another ACCW instruction is executed to restore the CCW,data addresses to their local values.

START I/O INTERRUPTION EXAMPLE





MONITOR CALL

The monitor call feature allows the programmer to place a trace on his program operation. The monitor call instruction is inserted as desired, throughout the program to cause program interrupt when that point is reached. The conditions can be made selective by segregating them into up to sixteen classes. These classes can be masked to allow interrupts only for specific conditions. A code number placed in the operand field of the instruction is reported along with the monitor class number during each interruption.

Monitor Call Instruction

0	8	16	20	31
AF	12	81	D1	
MC			SI	<u> Kolesee</u>

12 Bits 8-11 must be zero.

- Bits 12-15 contains one of sixteen monitor class numbers in binary notation for the test.
- B1 General register to used in determining the monitor code along with D1.
- D1 This field, along with the register defined by B1, is added to produce the monitor code. (D1 can identify 256 codes without the use of the register.)

The monitor class number specified in the 12 field of the instruction is tested against the monitor class mask in control register 8. The sixteen bit mask identifies the monitor classes with bit 16 for class 0 and bit 31 for class 16. If the mask allows the class, a program interrupt is initiated. When a class is inhibited, the monitor call instruction passes as a No-Op.

Control Register 8

00000000	0000000	Monitor Class Msk	
0		16 31	

When the instruction initiates an interrupt, the monitor class number specified by the instruction is posted in permanent storage byte location 95. The class is posted as a binary number in the loworder with the high-order set to zero. The computed monitor code is set into permanent storage word location 156 with byte 0 set to zero.

Permanent Storage 94

0000000	ю	Mon C	.L #	PER Code	00000000	ļ
0		8	1	6	24	

Permanent Storage 9C

00000000	Monitor Code
0	8 1.4 1.4 1.4 1.4 1.4 1.4 1.4 1.4 1.4 1.4



MONITOR CALL CPU 130

EXTENDED CONTROL MODE

Introduction

The extended control (EC) feature of the 3145 initiates and reports a number of System/370 functions. The CPU can still operate in basic control (BC) mode to accommodate programs written for System/360. A new PSW format is used along with control registers and extensions to the permanent assigned storage to implement EC mode.

Bit 12 of both the old and new formats of the PSW has a common notation for identification. When the bit is set to 0, the CPU reads the PSW as BC mode. If the bit is set to 1, the PSW is read as EC mode. The change from one mode to the other can be made with any PSW interchange. An I/O operation can be started in one mode and continued to an ending in the other mode. EC mode is required in order to perform most of the System/370 features.

Feature Mask

The changes to the basic PSW are made to satisfy requirements of the EC feature. An expanded system mask controls additional features.

Mask bits for the new features replace the system (I/O) mask of the BC mode PSW. The system mask is moved to the control registers. The condition code and program mask fields have been moved to bits 18 · 23 of the EC mode PSW. The ILC and the interruption codes have been moved to an area of the permanently assigned storage.

EC mode activates the following features on the 3145. System Mask bits on 1 allow the indicated features to function.

R Program Event Recording (PER) – This feature allows the programmer to debug his programs by identifying instructions that could cause trouble in operation. These include branch conditions, instruction fetch, storage alterations, and general register alterations. The recognition of one of these conditions causes a program interruption with the instruction address and code posted in a permanent storage location.

T Dynamic Address Translation (DAT) – This feature allows conversion of programs expressed in virtual address to real address in main storage. The translation does not occur for I/O addresses or for permanently assigned addresses used by CPU. The I/O counterpart is the Indirect Data Addressing (IDA) feature that is not masked and is allowed in BC mode.

I/O Input/Output Mask — The I/O mask that allows interruptions for the I/O channels selectively is located in control register 2. The mask bit in the EC mode PSW represents a master mask.

E External Mask — The External mask that allows interruptions for the external devices selectively is located in control register 0. The mask bit in the EC mode PSW represents a master mask.

The diagram shows the related mask bit, feature and contro% egisters. Greater detail of the feature is defined under the feature name.



3145 TM CPU 131

BC and EC PSW formats

Control Registers

The 3145 has sixteen one-word control registers that provide an extension to the EC mode PSW. The control registers are located in control storage at addresses F480 through F48F. Three of these control registers also function with BC mode for external mask and machine check controls. Details of the register assignments are contained in the sections of this manual that discuss the features.

The registers are loaded to standard masking and address values during system clear. The registers can be loaded from main storage with the load control instruction (LCTL). The information in the registers can be transferred to main storage with the store control instruction (STCTL). When using the load and store instructions, the unused registers must be considered in the main storage area specified.

CONTROL REGISTER ASSIGNMENTS

0	System Control	Translate Control	External Interrupt Masks
1	Segment Table Length	Segment Ta	ble Origin Address
2	Channel Mas	ks	
3			Reserved
4			Reserved
5	가 안 없는 것		Reserved
6			Unassigned
7			Unassigned
8			Monitor Mask
9	PER Event Masks	00000000	PER General Register Alteration Mask
10	00000000	PER S	tarting Address
11	00000000	PER E	nding Address
12			Unassigned
13			Unassigned
14	Error Re Control	covery and Masks	
15	00000000	MCEL A	ddress
Ċ) 8	3 2 1	6 24 31

CONTROL REGISTER 0 Bits 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 25 26 27 28 29 30 31 21 22 22 24 01=2K 00=64K 10=4K 10=1M System Translate External Interrupt Masks Control Control Reset to 00 00 00 E0 **CONTROL REGISTER 1** Bits 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 Segment Table Length Segment Table Address 000000 Reset to 00 00 00 00 **CONTROL REGISTER 2** 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 28 27 28 29 30 31 Bits 01 I/O Masks Reset to FF FF FF FF Note: Initial Value of unassigned positions in all reis unpredictable but is assumed to be 0.

31

predictable but is assumed to be 0.

EXTENDED CONTROL MODE

CPU 132



Reset to 00 00 00 00

상품은 고양 영화 관계 전



CONTROL REGISTER 11

						<u>.</u>	÷	14.	100	<u>`</u>		1.7				- 16			1	1.1			· · ·	a 1.								
Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
		0 (000	00	00				Γ								PER	Ene	ling	Ade	ires											
Reset to 00	00	00	00						.																							
EXTENDED CONTROL MODE CPU 134

Permanent Storage Assignments

System/370 assigns the first 128 bytes of main storage as fixed operation-addresses and adds assignments to much of the area up to 256 bytes. It also reserves the main storage addresses between 256 and 512 for feature use and an option for logout starting at address 512.

Most of the new area between address 80 and 100 is used only by EC mode operation and features. BC mode operations do not develop the interruption codes and logging information that is stored here. The features added with EC mode also post their result codes and addresses in this new area.

An I/O interruption posting area located between addresses AC and CO allows greater error definition in EC mode, than in BC mode. A portion of this posting area is also posted by BC mode operations to allow for combinations of EC/BC mode. The BC mode program cannot always handle the information automatically. Details of the permanent storage assignments are in the manual sections that discuss the features.

DEC	HEX	PERMANENT STORAGE ASSIGNMENT
0 0 4	00 00 04	Initial Prog Load PSW or PSW Restart New PSW
8 12	08 0C	Initial Prog Load CCW1 or PSW Restart Old PSW
16 20	10 14	Initial Prog Load CCW2
24 28	18 1C	External Old PSW
32 36	20 24	Supervisor Call Old PSW
40 44	28 2C	Program Old PSW
48 52	30 34	Machine Check Old PSW
56 60	38 3C	Input/Output Old PSW
64 68	40 44	Channel Status Word
72	48	Channel Address Word
76	4C	Unassigned
80	50	Timer
84	54	Unassigned
88 92	58 5C	External New PSW
96 100	60 64	Supervisor Call New PSW
104 108	68 6C	Program New PSW
112 116	70 74	Machine Check New PSW
120 124	78 7C	Input/Output New PSW

DEC	HEX	PERMANENT STORAGE ASSIGNMENT				
128	80	Unassigned				
132	84	00000000000000000000000000000000000000				
136	88	0000000000000 ILC SCV Intp Code				
140	8C	0000000000000 ILC Prog Intp Code				
144	90	00000000 Translation Excpt Addr				
148	94	00000000 Monitor Class No. PER Code 0000000				
152	98	00000000 PER Prog Event Addr				
156	9C	00000000 Monitor Code				
160	A0	Unassigned				
164	A4	같은 것이 있는 것이 같은 것이다. 이 것이 것은 것은 것은 것은 것은 것은 것은 것은 것이 있다. 같은 것이 있는 것이 같은 것은 것이 있는 것이 같은 것은				
168	A8	Channel ID				
172	AC	00000000 IOEL Pointer				
176	80	Limited Channel Logout (ECSW)				
180	B4	Unit St Chan St Count				
184	B 8	Key Flag I/O Address				
188	BC	CCW Address				
192	CO	Unassigned				
196	C4					
200	C8					
204	cc					
208	D0					
216	08	CPLITimer				
220	DC					
224	EO	Clock Comparator Reserved				
228	E4					
232	E8	Machine Check Intp Code				
236	EC					
240	FO	Unassigned				
244	F4					
248	F8	00000000 Failing Storage Addr				
252	FC	Region Code				
256	100	CPU Independent Log				

....

PSW Interchange Sequence

When starting an interruption sequence, the current PSW is stored as the old PSW for that class of interruption. The new PSW for the interruption class is then read in and set as the current PSW for the interruption routine. The sequence is similar in both BC and EC mode. The differences result from the allowable features. A similar sequence is taken at the end of the interruption when the old PSW is returned from storage to fill the current PSW in order to continue the problem program.

- Before leaving the GICM routine, the EC interruption code is stored in the assigned permanent storage location, and the first word of the entered PSW is returned from control storage FF38.
- After branching to the GIPW routine, the CC and the program mask from the U0 register are set in byte-2 of the first word of the PSW.
- The ILC is not stored in the old PSW for EC mode but is stored with the interruption code when required.



- If the new PSW does not call for translate mode, the translate tables and the mode bit are reset to ensure that the mode is off.
- · The EC system mask is moved to the EPSWA register for operation.



Α



Interrupt Codes

INTERRUPT	•			
			DRESS	
CLASS	CODE	8C	EC	FUNCTION
External (Note 1)	1005 1004 0040 0080 00XX	1A	86	CPU Timer Clock Comparator External Interrupt Key Interval Timer Direct Control
Supervisor Call (Note 2)	xxxx	22	8A	
Program	01-0F 10 11 12 40 80	2A	8E	(See System/370 Principles of Operation, GA22-7000.) Segment Translation Exception Page Translation Exception Translation Specification Monitor Call PER (Note 4)
Machine Check		32	E8	See Error Handling (Machine Check Logout)
1/0 (Note 3)	xxxx	3A	BA	

Note 1 Code is bit significant by external interrupt signals.

Note 2 Code is dependent on the I-field of the supervisor call instruction.

Note 3 Code is dependent on I/O device address.

Note 4 PER interrupt concurrent with another program interrupt ORs the value 80 with another interrupt code to yield a resultant interrupt code.

Store Then Mask Instructions

The two store then mask instructions can adjust the system mask during the program sequence. The AND instruction removes mask bits from the mask. The OR instruction adds bits to the mask. In either case the current mask is stored at the address specified for later recall, when required. If control register 0 bit 1 = 1, the instructions are not executed.

Store Then AND System Mask

	1	16	2	n
AC	Mask		B1	D1
INSM		SI		

Store Then OR System Mask

STOSM		SI		11 - K
AD	Mask	B1	D1	
0	8	16 2	0	31

The store then mask instructions are performed in supervisor mode for either BC or EC mode. The first eight bits of the current PSW are stored at the specified first operand address in main storage. The current system mask bits are then logically ANDed or ORed with the immediate operand of the instruction and the results returned to the operating register. In BC mode, the I/O mask stored in the MSKB external register is stored and adjusted. For EC mode the new system mask stored in the EPSWA external register is stored and adjusted. No other changes are made to the operation. CPU 138

DYNAMIC ADDRESS RANSLATION

- DAT (dynamic address translation) is a combination of software, hardware, and microprogramming.
- No special programming conventions are required by the problem programs to utilize the DAT feature.
- The Supervisor program handles the problem programs by dividing them into unique blocks, and moving these blocks in and out of real storage as needed.

Introduction

The DAT feature enables the storage capabilities of the 3145 to be extended beyond the real storage size. This is accomplished by locating programs and data on an external direct access storage device (disk, drum, etc.). This is referred to as virtual storage. The information resides in virtual storage until the controlling program requires it. At this time, control is transferred to a supervisor program which uses I/O routines to move the needed information into main storage. Control is returned to the program requiring the information to complete the interrupted operation. A maximum of ixtee: million bytes (16M) may be used as virtual storage by the 3145.

The supervisor program controls data movements from or to virtual storage by dividing it into increments designated as segments and pages. A segment is the largest division of virtual storage and may be either 1M or 64k bytes in size. The page is a subdivision of the segment and may be either 4k or 2k bytes in size. Segment and page size are parameters set up in the supervisor program according to the customer operational needs. The movement of data between real storage and virtual storage is always in pages of 4k- or 2k-byte increments depending on which size the supervisor is using. To control this data movement, the supervisor program initially builds segment and page tables when the real and virtual storages are loaded. These tables are continuously being updated to reflect the current location of data as it is being used by problem programs and moved between storages by the supervisor.

DAT is a hardware addressing function. The software supervisor manipulates data between storages and sets up and maintains the tables and control used by the hardware. DAT uses the hardware assigned for the OS DOS Compatability feature. Both the DAT and OS DOS features can be active at the same time. Handling of the address adjustment factor for both is accomplished by the GGST micro routine.

Addresses Subject to Translation

When the System/370 microprogram handles addresses that are subject to translation, the storage control word using that address, specifies the ADJ function. The ADJ decode enables the address adjustment hardware. If address adjustment is designated by the EPSW, the translation process takes place.

The following addresses are maintained, interpreted, or stored

- as virtual addresses, and are subject to translation:
 - 1. Instruction address in PSW
 - 2. Branch addresses
- Operand addresses when that instruction uses the address to refer to a main storage location. This excludes Set Storage Key, Insert Storage Key, and Reset Reference Bit Addresses.
 - 4. Address stored in register 1 by Translate and Test and Edit and Mark.
 - Address stored at location Hex 90 on a translation exception interruption.
 - Address stored at location Hex 98 on c program event recording (PER) interruption.
 - 7. PER starting address in control register 10.
 - 8. PER ending address in control register 11.

Examples of storage words designating address translation:

DW Q ADJ, W+4 STW X ADJ, V

Addresses Not To Be Translated

Address that are not to be translated are handled by storage control words that do not specify the ADJ function. When a storage control word without ADJ is executed, the ADR ADJ trap latch cannot be set (no mis-match trap can occur), and no addressing of the translation lookaside buffers takes place.

The following addresses are not translated by the CPU or channel.

- 1. Segment table origin address.
- 2. Page table origin address.
- 3. Machine check extended log pointer in control register 15.
- 4. I/O extended log pointer at location Hex AC.
- 5. Address stored at location Hex F8 (failing storage address) on machine check interruption.
- 6. Region code stored at location Hex FC on machine check interruption.
- 7. PSW addresses.
- 8. Address used by hardware to update location Hex 50 timer.
- 9. Address for command address word for Start I/O or Start I/O Fast Release Hex 48.
- 10. Addresses for channel command words.
- 11. Addresses for fetching or storing data by the channel. 12. Address for channel status word during execution of an
- I/O instruction or during an interruption. 13. Address of PSW used during an IPL,

Examples of storage words not designating address translation:







DYNAMIC ADDRESS TRANSLATION **CPU 140**

Develop Address





- 4 Page out the real storage page if required.
- B Page in the virtual page to the real storage location. 6 Return to I-cycles routine GAAI to execute the instruction the

CPU 1

trap was initiated from.

DYNAMIC ADDRESS TRANSLATION **CPU 144**





Segment No.

Page No.

2k Page Size



DYNAMIC ADDRESS TRANSLATION

CPU 146

Segment Table Entries

- A segment size of 1M has 16 segment table entries.
- A segment size of 64k has 256 segment table entries.
- Each entry in the segment table designates the length, availability, and the origin of the corresponding page table.

Entries in the segment table have the format:

Length			Page	Table Origin	Address			Ĩ.	
0 3	4	.7	8			28	29 30	31	

Length: Bits 0-3 specify the length of the page table in increments that are equal to a sixteenth of the maximum size of the table. This code is compared against the 4 high-order bits of the page number to determine whether the page address designates an entry within the page table.

EXAMPLE

PAA			
Byte 1		Byte 2	Byte 3
01234567	012	3 4 5 6 7 0	1 2
Segment no.	Page no.		•
en de la companya de National de la companya de la company			
64k Segment size 4k Page size	0 =	one entry one six- teenth of the table size or 4k (minimu sixteen entries of 4 increments or 4k x 64k table size or ec to one segment (M mum).	m). ik 16= qual axi-
64k Segment size	0 =	2 entries	
2k Page size	F=	32 entries	tan ing tang dia karang dia karang Karang dia karang dia ka
1M Segment size	0 =	16 entries	
4k Page size	.¦.'F ≠	256 entries	a da sera da s Esta da sera da
1M Segment size	0 =	32 entries	2 AU
2k Page size	F =	512 entries	
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~	.1		
0 1 2 3 4 5 6 7	0	Ξ.	
Length			

If the address is not within the page table, a page translation error is recognized (Prog Intr code 11).

# Page Table

Origin: Bits 8-28 with three zeros, gated as bits 29-31, form a 24-bit real address that designates the origin of the page table.

# (Invalid):

Bit 31 controls the availability of the segment. When bit 31=0, address translation proceeds. When bit 31=1, a segment translation exception is recognized, and control is passed to the supervisor (Prog Intr code 10).

# PAGE TABLE ENTRIES

 Each entry in the page table designates the availability of the page and contains the high-order bits of the real page address.

The page table entries are 2 bytes in length and are divided into fields:



## Page Address:

Bits 0 to 11 or 0 to 12, depending on the page size, provide the high-order bits of the real storage address.

#### (Invalid):

Bit 12 or 13, depending on the page size, specifies the availability of the page associated with the page entry. When the bit is zero, address translation proceeds. When the bit is one, a page translation exception is recognized (Prog Intr code 11).

Bit 14 (for 2k page) and bits 13 and 14 (for 4k page) are set to zero when the page entries are formed by the supervisor program. If these bits are not 00 whenever the page entries are accessed, a translation specification exception is recognized (Prog Intr code 12).

# OS DOS Compatability and DAT Active

When both of these features are active and the local area of the , DOS emulated environment is being accessed, the adjustment factor is added to the DOS address. The result is the virtual address that is handled by the DAT mechanism. A normal DAT translation is performed with this virtual address. Byte 0 and 1 of the TLBs is the virtual address without adjustment. Bytes 2 and 3 contain the real address entry of the page accessed by the adjusted address.





# Examples:

 Byte O		Byte 1	Byte 2	Byte 3
40		64 k segmen	t 2k page	
80	: ≝'	64 k segmen	t4k page	
50	=	1M segment	2 k page	
90	= '	1M segment	4 k page	
		Ŧ		

Note: These are the only valid combinations for byte 1. Remaining combinations result in a translation specification. (Prog Intr code 12).



Bits 0 through 7 designate the length of the segment table in units of sixteen entries. This code is used to determine whether the entry specified by the segment number falls within the segment table. If not, a segment translation exception results (Prog Intr code 10).

Segment Table Lengths



- 1M = 00 in the segment table length specifies 16 entries in the segment table. This is the only valid entry for 1M segments.
- 64 k = 00 specifies 16 entries in the segment table.
- 64k = OF specifies 256 entries in the segment table.

# DYNAMIC ADDRESS TRANSLATION CF



# Reference and Change Bit Recording

- With the DAT feature installed, 2 additional bits in the storage protect key are activated.
- Bits 0-4 of the storage protect key are used in the standard manner.
- Bit 5 becomes the reference bit and is set each time the associated 2k block is accessed from real storage for a read or store operation.
- Bit 6 becomes the change bit and is set each time a 2 k block is accessed for a store operation. This indicates to the supervisor that the page has been altered and must be written back or updated in virtual storage to reflect the change.
- Reference and change recording is active in either EC or BC mode, and whether DAT is active or inactive.



Note: Reference and change recording operates on 2048-byte blocks (storage protect blocks) regardless of page size.

The paging supervisor uses the status of the reference and change bits to keep track of the areas in real storage that are avialable for use.

Whenever a page is referenced that is not located in the real storage, the paging supervisor must fetch this page from virtual storage and place it in an available page location in real storage. By interrogating the storage protect keys, the supervisor determines which page in real storage may be replaced with the page from virtual storage. If possible, a page is selected that has not been accessed. If all pages have been accessed, the paging supervisor selects a page that has been referenced only. If all pages have been modified, the paging supervisor selects a page in real storage, pages it out to the virtual storage device, and pages in the required new page to the vacated area.

DYNAMIC ADDRESS TRANSLATION CPU 150

Translation Lookaside Buffer LB) TLB REGISTERS 0-7, BYTES 0 AND 1

Bytes 0 and 1 of the TLB registers contain the virtual address portion of the page addresses that have been accessed by the current program. The virtual addresses are loaded into the TLBs during the execution of the DAT trap routine GGST.

The virtual address is gated from byte 1 and byte 2 (Bits 0.3 for 4K page; bits 0.4 for 2K page) of the PAA' by the destine table line and the active LRU line.

The destine table line is activated by the decode of the DK expanded local storage address. After the segment and page table addresses have been formed in the GGST routine, the control word RDH DK WK, NOP is executed. The WK register contains the page table entry address which in turn contains the REAL address that is associated with the virtual address causing the DAT trap. This real address is gated to bytes 2 and 3 of the TLB selected. At the same time, the virtual address from the PAA is gated to bytes 0 and 1 of the TLB selected.

Whenever bit 7 of NP2 is set, the reset table line brings up Reset Reg 0. This is done when the Purge Table instruction is executed, and when a bad page or invalid page is detected in the

TLB REGISTERS 0-7, BYTES 2 AND 3

Bytes 2 and 3 are set with the real address from the page table entry that is accessed in the translation trap routine GGST. After the segment and page table addresses are formed in the GGST routine, the control word RDH DK WK, NOP is executed. The WK register contains the page table entry address which contains the REAL address associated with the virtual address that caused the * translation trap. The REAL address is gated on the EBI to bytes 2 and 3 of the selected TLB.

The DK function brings up the destine table line. This, when ANDed with the LRU line results in Set Reset Address Adj Reg 0 which gates the real address to the proper TLB.

The output of the selected TLB is gated to the real address assembler to the M-register. The displacement value from the

execution of the GGST routine. All bits including parity are reset.

The output of the TLBs is gated to the MATCH circuits whenever a translatable address is gated to the PAA. When a match occurs between the virtual address from the TLBs and the PAA, a match line is activated. This match line gates the set and reset function for the LRU matrix.

The virtual address is also gated to the display assembler whenever the external address 2E is used in a display operation. Switch H determines the register to be displayed. The match register can be displayed through external 08.

Note: TLB registers may be valid with either odd or even parity.





MT311 - 344



MT111 - 132

3145 TM CPU 151

# TES 2 AND 3 PAA is gated to the M-register and forms the low-order portion of the full real address.

.

Bytes 2 and 3 of the TLBs are displayed in the byte 2 and 3
 position of the external display when the external address 2E is used in the display operation.

Switch H determines which register (0-7) is to be displayed.

Example



# DYNAMIC ADDRESS TBANSLATION CPU 152

The LRU output is used to gate the virtual address from the PAA and the real address from EBI into the selected TLB when the DK function is used as a destination in a control word. The X and Y lines for the DK function bring up the destine table line, which is ANDed with the output of the LRU matrix to access the proper TLB.



# LRU Matrix

• Used to address the least recently used TLB when an address is to be loaded into the TLBs.

• Set and reset by the match circuit or by the microprogram.

Whenever a storage word with address adjust specified is executed, a match of the virtual address from the PAA and the TLBs is attempted. When a match is made, the real address from the TLB causing the match is gated to the M-register along with the displacement from the PAA.

The column in the LRU corresponding to the TLB causing the match is reset. The row in the LRU corresponding to the TLB causing the match is set. This causes the LRU output to point to the next TLB that has been least recently used. The output of the LRU matrix is only used when the TLBs are being used as a destination.

Whenever the TLBs are reset, the LRU is reset. The line (reset tables) is brought up by bit 7 of NP2. This bit is set by the microprogram when the reset is necessary. The control word, normally used to set bit 7 of NP2, is a branch word— NP2, OR, K01. This word is generally followed by NP2, A-, K01. The second branch word resets bit 7 of NP2 enabling the TLBs and LRU to be used for address translation. The LRU is displayed in external 08 byte 2.



# Working Register (WK)

The working register is used as the destination of the computed segment table entry address or the computed page table entry address.

The segment table entry address is formed by the control word:  $WK\mbox{=}SN\mbox{+}SO$  .

The WK register is then used as the address source register in the storage control word that accesses the segment table entry: RDW RW WK, NOP. This word places the page table origin address into the RW register.

The page table entry address is formed by the control word: WK=PN+RW;

The WK register is then used as the address source register in the storage control word that accesses the page table entry: RDH DK WK, NOP. This word gates the real address from the page table entry to the TLB, and also gates the virtual address from the PAA to the TLB.



WK register becomes the destination.

## NPO and NP1 Registers

NPO and NP1 registers may be set from either the EBI or the PAA. When NPO or NP1 are specified as the destination by any control word except a storage word, the data setting these registers is gated from the EBI. When a storage word with ADR ADJ active is executed, the address from PAA bytes 1 and 2 is gated to NPO and NP1.

expanded local storage using address 7B.



# NP2 and NP3 Registers

NP2 is loaded from the EBI whenever NP2 is addressed as a data, • destination. The outputs of the NP2 register are used for various controls throughout the DAT hardware.

When NP2 is addressed as a source, it is gated out to byte 2 of the address adjust entry. Bit 6 (Segment Number Invalid) is sampled directly from the segment number compare circuit to bit 6 of byte 2 address adjust entry.

NP3 bits 0, 1, and 3 are set from the EBI by the gating shown in the diagram. NP3 bit 2 is set directly from the page number compare circuits. The outputs of these latches are gated to byte 3 of the address adjust entry when NP3 is addressed as a source.

NP2 and NP3 are displayed in the byte 2 and 3 positions of the external display using expanded local storage address 7B.

	NP2		
GIPW ROUTINE (ECPSW BIT 5) GCCR ROUTINE (Load Control Registers)	BIT 0 BIT 1	ADDRESS ADJUST MODE	ACTIVATES DAT HARDWARE     INVALID PAGE OR SEGMENT SIZE IN GCCR ROUTINE
( <u>Not Used</u> )	BIT 2		
GCCH ROUTINE (Control register 0 Bit 11)	BIT 3		= 1 = 1M SEGMENT 0 = 64K SEGMENT
(Control register 0 Bits 8 & 9)	BIT 4	EMULATOR FEATURE MODE	0 = 2K PAGE
SS2 LARGER THAN SEGMENT NO.		SEGMENT NO. INVALID	SEGMENT NUMBER
(SS2 Compared to four high bits in PAA) GGST AND GCCR ROUTINE	BIT 7	RESET TABLES	TOO LARGE = RESETS TLB 0 AND 1 AND 1 BU
	1.00	1	

	NP3				
INVALID BIT ON IN PAGE TABLE		PAGE INVALID	-	INTERRUPT CODE 11	
(Bit 12 or 13, depending on page size)	5110				
INVALID BIT OFF IN SEGMENT	BIT 1	SEGMENT VALID	=	INTERRUPT CODE 10	(Note 1
Length field in segment table comp. to PAA page no.	BIT 2	PAGE TOO LARGE	⁻	INTERRUPT CODE 11	
BIT 14 IN PAGE TABLE #0	-BIT 3	PAGE FORMAT INVAL	<u>0   </u> =	INTERRUPT CODE 12	
(for 2K page, and bits 13 and 14 #0 for 4K page)	Ļ	Note 1:	Results in a ir	terrupt if the bit is not on	

Bits 4-7 not used

# ent Number Compare

The SS2 latches 4.7 are set in the GCCR routine by the arithmetic word SS2=SO0, OE, 0. The SS2 latches now contain the four low-order bits of the segment length code. The high bit latch is set if any of the high-order bits of the segment

# SEGMENT NO.

#### SS2 EBI BYTE 2 BIT 4 PH 0 4 PH 1 5 Logic SEG. NO. INVALID PH 2 n۵ 6 РН 7 BIT 0-3 OR PH HI DESTINE SEG. LENGTH REG. TRANSLATE INVALID

Page Number Compare

The page number from NP0 or NP1 is compared with the page table length code from the segment table entry. For a 1M segment, bits 4-7 of NPO are compared with the page table length code. For a 64K segment, bits 0-3 of NP1 are compared with the page table length code.



3145 TM CPU 155

#### length code are on.

A comparison is made between the SS2 latches and the segment number from NP0. For a 64K segment, bits 4-7 of NP0 are used in the compare operation. For a 1M segment, bits 0-3 of NP0 are used in the comparison. SS2 is in Exp LS 7D but there are no display facilities for this location,





DYNAMIC ADDRESS TRANSLATION **CPU 15** 









CPU 160

# Load Real Address (RX)

#### LRA

	81	R1	X2	B2	D2	
0		8 1	2 16	• •	20	31

The real address of the second operand is inserted in the general register specified by the R1 field. The remaining high-order bits of the general register are set to zero.

The virtual address specified by the X2, B2, and D2 fields is translated by the address translation facility regardless of the setting of the translation bit in the PSW. The 24-bit real address is inserted in bit positions 8-31 of the general register specified by the R1 field, and bits 0-7 are set to zero. The translated address is not inspected for resolution, protection, or validity.

The condition code is set to 0 when translation can be completed: that is, the entry in each table is within the specified limits and the invalid bits are zero. When the invlalid bit in the segment table entry is on, the condition code is set to 1 and the real address of the segment table entry is placed in the general register specified by the R1 field.

When the invalid bit in the page table entry is on, the condition code is set to 2 and the real address of the page table entry is placed in the general register specified by the R1 field.

When either segment table entry or the page table entry is outside the table, the condition code is set to 3 and the register designated by the R1 field contains the address of the entry that would have been referred to if the length violation had not

ocurred.



- 0 ---- Translation available
- 1 ---- Segment table entry invalid
- 2 ---- Page table entry invalid
- 3 ---- Segment or page table length violation

# **Program Interruptions**

Translation

 $|F_{i}| \leq \frac{1}{2} |F_{i}| \leq \frac{1}{2}$ 

Specification:

Operation:	The DAT feature is not installed. The operation is suppressed.
Privileged Operation:	The CPU is in the problem state. The operation is suppressed.
Addressing:	The address of the segment table entry designates a location outside the available main storage. The op-
	eration is suppressed.

Bits 8-12 of control register 0 contain an invalid code, or the page table entry has a format error. The operation is suppressed.



# Reset Reference Bit

## RRB

I	B213		B1		D1		
I	0	16	20	) • .		31	

The reference bit in the storage key associated with the operand address is set to zero.

No other access to the key is permitted between the moment of fetching and the moment of storing the key. The remaining bits of the key are not affected by this instruction.

The operand address designates a location in real storage and is not subject to address translation. Protection does not apply to this reference.

The condition code is set to reflect the status of the reference and change bits prior to the setting of the reference bit to zero.

# **Resulting Condition Code**

0	Reference bit 0, change bit 0
1	Reference bit 0, change bit 1
1	

2	neierence bit 1, change bit u	
3	Reference bit 1, change bit 1	

# **Program Interruptions**

Operation:	The DAT feature is not installed.	
	The operation is suppressed.	ŧ.
Privileged Op-	The CPU is in the problem state.	
eration:	The operation is suppressed.	

# Purge TLB

# PTLB

-	820D		
2	0 10	6	31

All information in the translation lookaside buffers is made invalid. No changes are made to the contents of addressable storage or other registers. The contents of bits 16-31 of this instruction are ignored.

## Condition Code

The code remains unchanged.

Program In	iterruptions
------------	--------------

Operation:	The DAT feature is not installed.
	The operation is suppressed.
Privileged Op-	The CPU is in problem state.
eration:	The operation is suppressed

# Hardware Error Checking

# Adr X-Late LRU Inval (Address Translate LRU Invalid)

The LRU should have only one active line at any time. The ADR X-Late LRU Invalid error indicates that an even number of lines are active which differs from the normal odd number condition. This error indicates a failure of the LRU hardware.



# Adr X-Late Mult Match (Address Translate Multiple Match)

During DAT translation operations, the ADJ storage words cause a comparison of the PAA (virtual) address to the eight TLB registers (bytes 0 & 1). Either a no-match condition results and a trap to GGST routing is performed to load the TLB, or a successful match results in gating the selected TLB (bytes 2 & 3) to the M-register. This error indicates a failure of the LRU, TLB, or match circuit resulting in the PAA matching more than one TLB register.



# Adr X-Late No Match (Address Translate No Match)

When a TLB register is loaded (with a storage word of the form "RDH DK WK"), a match is made via the normal match hardware. The absence of this match signal during the execution of this storage word indicates a hardware failure in the LRU, TLB registers, or match circuits and allows MCKB1 bit 4 to be set.

DESTINE TLB	· · · · · · · · · · · · · · · · · · ·	
ADR ADJ SAVE PAA		SET ELIMINATE (NO MATCH)
FORCE LRU INVALID (DIA		A (MCKB BYTE 1 BIT 4)
(DIAGNOSTIC ADR ADJ MATCH	OR N	IT 012
na na nationa di disettata		

DYNAMIC ADDRESS TRANSLATION **CPU 164** 

# DAT Exercise

This exercise may be used to verify the operation of the address translation hardware and microprograms. By altering the loop that is entered and the microprogram, scoping procedures may be formed for detailed checks of hardware operations. Assume that the DAT feature is installed and that the System/370 microprogram is loaded.

	영화 동안 같아요. 전화 2000년 1월 2000년 1월 2010년 1 1월 2010년 1월 2
PR-KB ENTRY	DESCRIPTION
Step 1 AM 600 50605100	Load an RX Store instruction with an R1 field of 6, B2 field of 5, and a displacement of 100, at location 600. RX Store OP R1 X2 B2 Displacement 5 0 6 0 5 1 0 0
Step 2 AG 5 00104000	Load general register 5 with the base number to be used in calculating the effective address of the second operand for the Store instruction. During I-cycles, the second operand effective address is formed by adding the contents of general register 5 with the displacement value. The result is the virtual address 104100. This a dress is beyond the real storage range of any Model 145 system. Therefore, this address must be translated to form a real address. The real address is loaded into a TLB by the address translated to form a real address. The real address is loaded into a TLB by the address translated to form a real address. The real address is loaded into a TLB by the address translated to form a real address. The real address is loaded into a TLB by the address is a known value and is loaded into the page table later in this procedure.
Step 3 AG 6 C0C0C0C0	Load general register 6 with the value to be stored at the second operand real address
Step 4 AM 604 47F00600	Load an RX Branch on condition instruction, with a mask field of F (branch unconditionally), and a displacement of 600, at location 604. This instruction branches to the Store instr tion to form a two-instruction loop. RX Branch on Condition 0P M1 X2 B2 Displacement 4 7 F 0 0 6 0 0
Step 5 AC 0 00900000	Load control register 0 with bits 11, 12 = 10 to specify a 1M segment, and bits 8, 9 = 10 to specify a 4K page.
Step 6 AC 1 00000500	Load control register 1 with a segment length code of 0 (byte 0 = 00), and a segment table origin address of 500.
Step 7 AM 500 00000800 00000700 00000001 00000001 00000001 00000001 000000	Load the segment table, located at address 500, with 16 entries as specified by the segment length code of 0. The first enty contains the page table origin address 800. The page table la cated at 800 is associated with the instruction addresses, which are located in the same 4K page of real storage. The second entry contains the page table origin address 700 which is associated with the second operand address of the store instruction. The remaining 14 entries are loaded with the invalid bits set. The page table length codes in both the first and second entries of the segment table, are 0, indicating a page table length of 16 entries.
Step 8 AM 800 00000008 00080008 00080008 00080008	Load the first entry of the page table located at address 800 with 0000. This is the high-order portion of the real address for both instruction addresses. Load the remaining 15 entries this page table with the invalid bits set.
Step 9 AM 700 00030008 00080008 00200008 00080008 00080008	Load the fifth page table entry of the page table located at address 700, with 0020. This is the high-order portion of the real address of the second operand of the store instruction. Lo the remaining entries with the invalid bits set.
Step 10 AM 0 04080000 00000600	Load the initial program PSW with bit 5 (translate mode), and bit 12 (EC mode) set. Place the address 600 in the instruction address field.
Step 11 AM 68 040A0000_00001234	Load the Program new PSW with bits 5, 12, and 14 (wait state) on. Place the address 1234 in the instruction address field. This address is used to identify any program interrupts that may occur.
Step 12 DV 104100 = 002100 xxxxxxx xxxxxxxx xxxxxxxx xxxxxxx xx	Perform the display virtual storage function. This operation causes the real address (right of the equal sign) to be calculated by the address translation routine GGST. The GGST routine is branched from the alter display routine. 104100 is the effective virtual address of the second operand of the Store instruction.
Step 13 DV 000600 = 000600 Ex60511 470600	Displaying the virtual address of the Store instruction, reveals that the virtual and real address for the instruction are equal.

Press the restart key. The address 600 from the PSW is gated to the I-register and the I-Cycles routine is entered. EC and translate mode are set and the system should run with the EC mode and ADR X-LATE indicators on. If the WAIT indicator is on and the address 1234 is displayed continually in the A-Register display indicators, interrogate the interruption code in the PSW located at address 8C.



Invalid Bit

CPU 166



#### **Check Execution**

The following procedure verifies execution and DAT register loading.

- Place address 600 in the address switches, Address Compare switch in I-COUNTER LOGICAL, Address Compare Control switch in STOP, Storage Select switch to MAIN STORAGE, System should stop.
- Place address 104100 in the address switches, Address Compare switch to ANY LOGICAL, Storage Select switch to MAIN STORAGE, Address Compare Control switch in STOP. System should stop.
- Place address 600 in the address switches, Address Compare switch in I-COUNTER REAL, Storage Select switch to MAIN STORAGE, Address Compare Control switch in STOP. System should stop.
- Place address 2100 in the address switches, Address Compare switch in ANY or DATA STORE, Address Compare Control switch in STOP, Storage Se lect Switch to MAIN STORAGE. System should stop.

If a program check occurs, display Program Interrupt Codes at location 8C.

Main Storage Byte 3

00000000 00010000 10 – Segment Translation Exception Caused by: Segment invalid bit on, or Segment length code error.

12 - Translation Specification

Caused by: Page table bits 13.

zero.

14 not equal to zero, or Segment table bits 29, 30 not equal to

Control reg 0 Bit 10 not=0. Invalid combination of page and segment bits in control reg 0 bits 8, 9 and 11, 12.

00000000 00010001 11 - Page Translation Exception Caused by: Page invalid bit on, or Page length code error,

The virtual address (bytes 0, 1) and the real address (bytes 2,3) are zero with the parity bits on. Parity bits on indicate that

this TLB was loaded by the microprogram.

0000000 00010010

Display DAT Registers 1. Display TLB 0 (External) Word address 2E in switches F, G Switch H to position 0 Storage Select switch to EXT REGS 2. Display TLB 1 (External) Word address 2E in switches F, G Switch H to position 1 Storage Select switch to EXT REGS

Byte 0 = 10 Virtual address Byte 1 = 40 Byte 2 = 00 Real address Byte 3 = 20

Place switch H in any position 2-7, Bytes 0 and 1 have not been accessed and are blank with no parity bits. TLBs 2-7 could contain residual data.

- 3. Display the LRU (External) Word address 08 in switches F, G Storage Select switch to EXT REGS
- The LRU, located in byte 2 of the display, should have bit 2 on. This indicates that TLB 2 is the least recently used and is to be loaded next.
- 4. Display the NP register (Expanded Local Storage) Word address 7B in switches F, G Storage Select switch to EXP LS

```
Byte 0 = 10
Byte 1 = 40
Byte 2 = 98
Byte 3 = 40
```

```
5. Control register 0 is located at control storage address F480.
Control register 0 = 00 90 00 00
Control register 1 is located at control storage address F484.
Control register 1 = 00 00 05 00
```

6. To modify the exercise for a scope loop, modify as follows:

AM 604 B20D0000 AM 60A 47F00600 DYNAMIC ADDRESS TRANSLATION

**CPU 16** 

# CHANNEL INDIRECT DATA ADDRESSING CPU 170

#### CHANNEL INDIRECT DATA ADDRESS (CIDA)

# Introduction

The CIDA feature can extend the address adjustment of the dynamic address translation (DAT) feature to the I/O channels. A contiguous set of virtual addresses can be mapped into a noncontiguous set of pages in real storage. Because only a single data address need be in effect at a time for a channel, using the DAT hardware to handle the adjustment with each access is not necessary. The adjustment factor is applied by the program, and the real address is stored for the CIDA controls. These adjusted addresses are stored as an IDA list (IDAL) for each CCW. Each word of the list is referred to as an IDA list word (IDALW).

The operation of the CIDA feature is selective and can be used in either BC or EC mode. An CIDA flag in the flag byte of the channel CCW functions is the indirect addressing switch. When the CIDA flag = 1, the normal data address of the CCW is the address of an IDAL in main storage. This list contains one or more addresses of main storage on page boundaries that can be used in sequence for the CCW operation. The first address is not required to be on a page (2K) boundary to permit filling a partial page. All remaining addresses in the list must be for the starting address of a page in main storage. The operation need not fill the last page used.

In operation the hardware must recognize the end of a page and enter the next address (IDALW) from the IDAL. For all operations except read backward, the end of the page is recognized by the change of the low-order eleven address bits from ones to zeros. For the read-backward operation, the change is from zeros to ones. The low-order eleven bits of an address must be zeros (2K boundry) for a forward operation.

The change is detected by testing for an inversion of byte 2 bit 4 of the data address. Any address in the IDAL other than the first that is not on page boundary, or that contains information in byte 0, causes a program check indication and the operation is terminated.

#### **Byte Multiplexer Channel**

For the byte-multiplexer channel, the CIDA feature requires a fourth word in each assigned UCW. This word holds the IDAL address for the assigned device. During the entry of the CCW, the CIDA flag bit is tested to determine the use of the data address field. A CIDA flag causes the address to be stored in the fourth word of the UCW, and then uses that address to enter the first IDALW to store as the data address in the UCW.

With each use of the data address for data transfer, a test is made for the updated address crossing a 2K boundary. This test is made by comparing byte 2 bit 4 of the address before and after the update. Any change represents the cross of a 2K boundary. When a change is detected, the IDAL address is entered from the UCW to fetch the next IDALW for the new data address. In CIDA command chaining, each CCW is tested for the 1DA flag: either the normal data address or the CIDA addressing routine.

# COMMAND CODE = WRITE IDA = 1

The data address in the CCW provides the address of which in turn provides the data address.



The data addressing for console printers is the same as for the multiplexer channel. The CIDA feature applies only to operations initiated by the Start I/O instruction. Alter and display operation do not translate addresses.

# Selector Block-Multiplexer Channels and IFA

With the CIDA feature, the selector and block-multiplexer channels operate in the same manner. A block-multiplexer channel can only disconnect at the completion of a command and, therefore, does not require storing the CIDA information in the UCW. Upon entry of a CCW either on initial selection or subsequent command chaining, the CIDA flag is tested. When the flag is present, the address in the CCW is not entered into the GDRL register. Instead, the address is that of the IDAL to be used for entering an IDALW into the GDRL register for the data address. The next IDALW is immediately read into the GDRL register to back up the GDRL register.

When the data address in the GDRL register has been either incremented or decremented until it crosses a 2K boundary, the hardware transfers the contents of the GDBRL register into the GDRL register and requests a CIDA data trap (D10C). When the

trap is honored, the IDAL address is used to enter the next IDALW into the GDBRL register. The addresses entering the GDBRL register contain only bytes 1 and 2 because they contain no key and because the low-order byte must be either 00 or FF. The data transfer stops in the normal manner at the end of the record or in the case of an error.

The IFA functions in the same manner as the selector channel for CIDA operations that use share cycles to transfer data to or from main storage. For control, sense, and operations that use microprogram to transfer data, the hardware forces the addressing to use the FDRL register when the FD register is addressed. The CIDA data trap request uses the IFA gated-attention trap (D120) and branches to the IDALW entry routine when no gatedattention condition is found. CIDA Data Address Conu



Page-End Detection



#### **Basic Selector Channel Addressing**

The selector channel data addresses supplied by the CCW define the starting address in main storage to be used for the operation. The data address is initially set into the GD register in local storage. This same data address is also set into the expanded local storage GDRL register for use in addressing by the share cycle. The share cycle hardware actually addresses the GD register but raises the B-register addressing gate to enter the GDRL register. During the second cycle count update, the address gate is dropped so that the GD address 1 enters the GC register. The address in the GD register is not used for the data movement.

## **Basic IFA Addressing**

The IFA addressing is similar to the selector channel addressing but involves some exceptions because of the hardware integration of the control unit. The share cycle transfers between the file and main storage are identical but use the FD and FDRL registers Share cycle transfers between the file and control storage involve the FA register for data address and the B-register address gate is not raised. The transfers of control, sense, and data information between control or local storage and main storage do not use the share cycle cortos. The main storage address is set into the FDRL register and the B-register addressing gate is forced by the hardware condition.

The data address from the CCW is entered into both the FD local storage register and the GDRL expanded local storage register. The address for any control storage area involved is entered into the local storage FA register by the microprogram. The share cycle controls address either the FD or the FA register for the data address depending on whether main storage or control storage is affected. When the FD register is used, the B-register addressing gate is raised to enter the FDRL expanded local storage register.

The IFA tlata transfers that do not use the share cycle controls force a three-step clocking sequence to gate the two-cycle storage word. During the second cycle court update, the gate is dropped and the count is entered from the data register and incremented by 1 for either the FC or the FB register. When transferring information between main and control storage, both of these addressing systems are used but without the share cycle controls. The microprogram controls the addressing with the gate being forced through hardware.

#### **CIDA Backup Control**

When the CIDA flag is set, byte 2 bit 4 of the address sets a polarity-hold latch. After the address has been updated as the result of the transfer, the same bit is compared with the polarity-hold latch to determine whether the level has been changed. A change represents the cross of a 2K boundary. The address backup register (GDBRL) bytes 1 and 2 are transferred into the GDRL register to continue the operation. Byte 3 of the GDRL register now stands at either all zeros or all ones that represent the 2K boundary for forward or backward transfers. Byte 0 of the GDRL register contains the protect key and does not change for the new address.

When the backup register is transferred, a request is made for a CIDA data trap to enter the next IDALW. When this trap is honored, the IDAL address in the GD register is used to enter the next IDALW into the GDBRL register. All IDAL words except the first are tested for 2K boundary. A part of the zero-count detection circuits is used to test the 11 low-order bits. These are all zeros for a forward transfer and all ones for a backward transfer. The read backward line reverses the bit levels through OE logic circuits to test for ones in the zero test circuit.

If this hardware detects a program violation in the CIDA list, the hardware forces on the highest-order address bit when the bad list entry is moved into the backup data address register (GDBRL). If data transfer continues until this entry of the list is needed, an address check occurs which results in a channel program check through the normal address check mechanism. If the data transfer is concluded before the bad entry is needed, no check is indicated.

If other activity on the system prevents honoring the CIDA trap before the data transfer exhausts the contents of both GDRL and GDBRL, a line is sent to the channel hardware. This prevents further share cycles from that channel until the CIDA trap has been honored. Note that this is a highly unlikely event because it requires that a one megabyte device has its trap request locked out for two milliseconds.

# CHANNEL INDIRECT DATA ADDRESSING CPU 172

# PROGRAM EVENT CORDING (PER)

The program event recording feature provides a means for debugging new programs or revisions. PER can alert the programmer when these events occur.

- Successful execution of a branch instruction.
- Alterations of the contents of designated main-storage locations.
- Alteration of the contents of a specified general register.
- Fetching of an instruction from designated main-storage locations.

# Introduction

The program has control over the conditions that are considered events in the program sequence. These events can be selectively monitored to aid in program analysis. When an event occurs, a program interruption is initiated if the masking conditions allow. An interruption for an event does not remain pending: if the interruption is masked, the information is lost. Information concerning the events is reported through the interruption codes. The PER feature operates only in EC mode.

## **Program Event Operations**

PER operations are initiated by setting one or more PER control bits to a 1 in control register 9 and setting the PER mask (ECPSW bit 1). The program to be scanned is performed in the normal manner.

Addresses defined in control registers 10 and 11 apply to both the 'instruction fetch' and the 'storage alteration' events. When the starting address is smaller than the ending address, the event area is from the starting address through the ending address. If the ending address is smaller than the starting address, the event area is from the starting address to the end of storage and from address zero through the ending address. A single storage address is defined when the starting address are the same.

With the PER feature allowed (ECPSW bit 1 = 1) and when an event is recognized, the interrupt is taken with the appropriate indications stored in the PER interrupt information in main storage words 94 and 98. If the PER mask bit is not set, the event conditions are lost.

# **Control Register Allocation**

Control registers 9, 10, and 11 are used to define the events to be monitored and the limitations imposed for the PER operation.

# **PER Control Bits**

Bits 0-7 of control register 9 specify the events to be monitored. When a bit = 1 the event is monitored.



# CONTROL REGISTER 10

Bits	0	1 2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20 Z	1 2	2 23	24	25	26	27	28	29	30	31
			000	000	000	)		5									PE	ERS	itartii	ng Ad	Idres	s								
	-		аны 2010	. 1. 4																										'

# CONTROL REGISTER 11

Bits	0 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
		0	00	00	000	)			S.								P	ER	Endi	ng A	ddr	ess									٦

# **Extended Interrupt Code**

A detected event condition is reported in the extended interruption code during the effected program interrupt for EC mode. The program interruption code of 80 defines PER as the cause. The PER conditions are reported in main storage words 94 and 98.

# PER Code

Bits 16-23 of main storage word 94 specify event or events causing the interruption. These bits are arranged in the same relation as the control bits in control register 9. When a bit = 1, the event was detected.

- Bit 0 Successful Branch Instruction
- Bit 1 Instruction Fetching
- Bit 2 Storage Alteration
- Bit 3 General Register Alteration
- Bit 4 Unassigned Bit 5 Unassigned
- Bit 6 Unassigned
- Bit 7 Unassigned

# Program Event Address

Bits 8-31 of main storage word 98 specify the address associated with the recognized event. This is normally the address of the instruction causing the event except an execute initiated instruction when the execute instruction address is stored.

# Word 94

[	· · ·	T	PER CODE	00000000
0	8	16		24 31
Word 98				

00000	000	PROGRAM E	VENT ADDRESS		
0	8	16	24	31	

Bit 0 Successful Branch Instruction

- Bit 1 Instruction Fetching Bit 2 Storage Alteration
- Bit 3 General Register Alteration
- Bit 4 Unassigned
- Bit 5 Unassigned
- Bit 6 Unassigned
- Bit 7 Unassigned

# PER GR Alteration Masks

Bits 16-31 of control register 9 specify which general registers are to be monitored for alteration of their contents. The 16 bits are assigned in the order of their ascending bit numbers to the 16 registers. When a bit = 1, the register is monitored.

# PER Starting Address

Bits 8-31 of control register 10 define the first address of the main storage area to be monitored.

# PER Ending Address

Bits 8-31 of control register 11 defines the last address of the main storage area to be monitored.

# PROGRAM EVENT RECORDING CPU 174

# Successful Branch Instruction

When the PER control bit 0 = 1, the operation sequence tests for a successful branch event following the execution of each branch instruction. A successful branch results from branching the instruction flow to the branch address of the instruction. The branch at the end of the load-PSW instruction is not considered a successful branch because that is the normal instruction flow.

When bit 0 of the PER code is set to 1, the program event address is set to the address of the branch instruction unless it was initiated by an 'execute' instruction that stores the instruction address. The address to which the instruction branched is stored as the instruction address in the old PSW.

## Instruction Fetching

When the PER control bit = 1, the operation sequence tests that the initial byte of the instruction falls within the monitored area of main storage. When an instruction is performed as the result of the 'execute' instruction, the initial bytes of both instructions are tested. The event is recognized if either byte falls within the monitored area.

When bit 1 of the PER code is set to 1, the program event address is set to that of the fetched instruction. If the fetched instruction is the object of an 'execute' instruction, that instruction address is stored. The next instruction address is stored as the instruction address of the old PSW.

## **Storage Alteration**

When the PER control bit 2 = 1, the operation sequence tests that the information is destined to an address within the monitored area of main storage. This does not include addresses initiated by the CPU for permanent storage and logout assignments. The storage is considered altered when the instruction could change the information, even if the value is modified by zero.

When bit 2 of the PER code is set to 1, the program event address is set to that of the instruction causing the reference. If the 'execute' instruction initiated the reference instruction, the address of the execute instruction is stored. The next instruction address of the old PSW.

# **General Register Alteration**

When the PER control bit 3 = 1, the operation sequence tests that the information is destined to a general purpose register defined by the alter mask in control register 9. The register is considered to be altered, even when the value is unchanged, if the instruction could have changed the value. The register could have been transferred to itself or the value modified by zero. If the instruction involves multiple registers, the event is reported when any one of the group is defined by the alter mask.

n ar an sia subrigan si 👘

When bit 3 of the PER code is set to 1, the program event address is set to the address of the reference instruction except when it was initiated by the execute instruction when that address is stored. The next instruction address is stored as the instruction address of the old PSW.

# PER Operations Introduction

At the end of each instruction, if the PER control is active, the operation is branched to the entry of the PER routine (GQER). The normal RTN LNK statement performs the branch because the link address has been altered during the setup to the B5 module. Of interest in this routine is the successful branch ending because this is one of the events being monitored. When the entry is through the branch leg, a test is made for the load-PSW instruction that is not included in the event test. If the PER control bit 0 is set, the routine posts the PER code bit 0 as the indicator in the PM register byte 1. One or more of the remaining PER code bits may have been posted by the routine before the instruction was performed. Any bits in this byte indicated that a PER interrupt is pending and the operation branches to the GICM routine to post the interrupt conditions. If no PER interrupt is pending, a test is made for other interrupts pending. In which case, the information for the one of highest priority is posted.

After posting the interrupt information for an interrupt, the operation branches to the GIPW routine to store the current PSW as the old PSW and enter the appropriate new PSW to process the interrupt. The exchange of PSWs normally results in masking the PER feature (PSW bit 1) for the interrupt sequence. The exchange may require additional setup operations for a change in the control mode. Before returning to the I-cycle hardware to start the first instruction of the interrupt program, the tests for PER events must be made on that addres if PER is masked on in the new PSW.

When no interrupt conditions are involved upon entry into the GQER routine or with the return from the GIPW routine, the three events involving addressing are tested. The address of the next instruction in the I-register is stored in control storage FFD0 for use in posting an interrupt after the instruction operation. The PER starting address and the PER ending address are entered to define the monitored main storage area. The two addresses are compared to determine whether the monitored area wraps storage which is considered acceptable. After determining that the instruction fetch event is to be monitored, the instruction addresses to determine whether it falls within the monitored area. When it does, the PER code bit 1 is posted to indicate.

Further event testing requires the entry of the instruction opcode. This op-code is masked to determine whether it is the execute instruction. In which case, the object instruction address must also be tested for possible location within the monitored area. The operation sequence enters the object op-code because it is this operation that involves the operand address. A test is also made for a possible execute-execute sequence that is invalid.

Following the instruction-fetch event test, the op-code digits are used to address the F3 module of control storage. The byte thus entered contains a PER control mask and four control bits that define unique conditions for the op-code. The PER control mask is ANDed with the PER control bits to remove those events that cannot occur. A bit remaining in the modified PER control byte indicates that the operand is considered changed. The four low-order bits are defined as follows:

- Bit 4 Special Instructions (B2 and EA)
- Bit 5 Insert Char under Mask (BF) Bit 6 Loag Multiple (98)
- Bit 7 Long Operation (2 GPRs).
The modified PER conservery to be considered. To make the test, the operand address information of the instruction must be entered and the address developed. This address is compared with the previously entered PER starting and ending addresses defining the monitored area of main storage. If the operand address falls within this area, the PER code bit 2 is set.

When the modified PER control byte contains bit 3, a test is made with the GPR-alter mask to determine whether the altered register is monitored. Bits 4, 5, 6, and 7 of the modified control byte represent special conditions to be considered in testing. Bit 4 identifies the B2 and EA instructions that are not tested. Bit 5 indicates an insert character under mask instruction that does not represent an alter when the mask is zero. Bit 6 indicates a load-multiple instruction in which a series of GPRs are altered. Bit 7 identifies a long operation instruction in which two GPRs are altered.

The address for a GPR is in binary notation in the instruction and must be encoded into a bit identification to compare with the GPR alter mask in control register 9. For a load-multiple operation, the instruction delines the first and last register involved and the bit identification for each register must be encoded. For a long operation the two consecutive registers are encoded. The identification bits are assembled into bytes 2 and 3 of a working register to be ANDed with bytes 2 and 3 of control register 9. Any matching bit condition results in a non-zero result to indicate that the event has taken place. The operation defines that a monitored register has been altered but does not indicate which one or whether more than one was altered. The PER code bit 3 is set to indicate the GPR-alter event.

Following the tests for PER for the next instruction, the operation returns to hardware I-cycles to process the next instruction. This instruction is the first instruction of the interrupt routine if the PSWs were exchanged.







#### DOCUMENTATION PLAN **VOL 32 VOL 34 VOL 30** Provides descriptions of keys, lights, and switches Describes motor-generator concepts and CPU power on the system operator console. Included are system. Includes sequencing, distribution, service Alphabetical subject index of documentation conconsole operating procedures and applications. checks, removal and replacement, and preventive System Co NSo Lo tents, and index of cross references among all PoWeR maintenance, volumes, INDEX Documentation plan defines the organization of Presents high level description of Phase 21 tech-Describes hardware and programs involved in the manual and the contents of major sections. nology. Describes the storage areas and functions detecting and handling of machine malfunctions. Includes legend pages describing manual symbology. which include: addressing, BSM data flow, and STOR age PLAN **RECovery Features** timings. Explains the use of Error Checking and Correction (ECC) in detecting and correcting data errors. Presents high level description of the CPU. Shows data and control flow, plus a brief description of major functional area. Describes the diagnostic hardware and programs **INTRoduction** used in the Model 145. **DIAG** nostic Functions Describes the working relationship of the functional units that make up the CPU. Uses unit data flows, **VOL 33** operational diagrams, text, and second level diagrams CPU Hardware where necessary. Provides a detailed description of the Model 145 Contains a summary of data that is useful to the I/O Channels: Byte-Multiplexer, Selector, and service representative when troubleshooting. Block-Multiplexer. Included are operational **VOL 31 REF** erence **CHaNneLs** diagrams and flow diagrams with related text. Defines the standard interface with expansions, Contains microprogram information needed to read the microlisting (microprogram source document). Describes the control words that make up the micro-**MIC** roprogram programs, and contains microprogram instruction examples. Covers the details of the adapter data flow, file operations, checking facilities, func-Presents a high level description of the Console File tional units, and interrupt handling. Integrated File Adapter and a detailed description of the Console-File attachment circuits. Console-File Adapter Covers optional features that are not covered in Presents a detailed description of the 3210 and 3215 related sections, included are descriptions of Console Printer-Keyboards and associated attachment .... IBM 1401/1440/1460 and 1410/7010 Compati-

Optional FEATures

bility, Direct Control, and Channel-to-Channel

Adapter.

circuitry. Includes microprogram operation and pro-

gramming information.

Console Printer-Keyboard

3145 TH PLAN 1

## ABBREVIATIONS

# А

A	and function
AAR	A-address register
ABM	advanced bi-polar monolithic (storage)
ABRTY	A and B retry register
ac	alternating current
ACB	address check boundary
ACBR	address check boundary register
ACR	automatic carrier return
A/D	alter/display
ADDR	address
adj	adjust
Adr-I	address-in
Adr-O	address-out
adv	advance
ALD	automated logic diagram
A-LS	A-local storage
alt in in	alter
Alt/Disp	Alter/Displa /
ALU	arithmetic logic unit
AM	address mark
amp 🦾	amplifier, ampere
ANUM	add numeric
appndg	appendage
APR	alternate path retry
arith	Arithmetic
ASCII	american standard code information
	interchange
ASCP	automatic system checkout program
asm	assembler
AT	attention (file)
ATTN	attention
avl	available

В	
BAL	branch and link
BAR	B-address register
88E	branch on bit equal
BC	basic control
BCA	bit count appendage
BCA	basic channel adapter
BCAI	basic channel adapter interface
BCD	binary coded decimal
BCE	branch on character equal
BDIL	branch and do interpretive loop
bfr	buffer
BI FLAG	branch on invalid flag
bin	binary
bik	block
BLS	B-local storage
BMF	block multiplexer feature
BR	bit ring
BR	branch

brd	board
BS	byte source, bit select
BSM	basic storage module
bwd	backward
BWF	branch if wordmark or zone equal
BYTDST	byte destination

### C C count CA control address CAR cylinder address register CAW СВ

\$*i

	eynneer eest tegister
CAW	channel address word
СВ	circuit breaker
CC	condition coch, chain command,
CCC	channel control check
ССН	channel check handler
CCW	channel command word
CD	chain data time of day clock
CDA	chained data
CDC	channel data check
CE	customer engineer
CF	console file
CFC	console-file checking
CFDA	console-file disk address
CFDR	console-file data register
chan	channel
char	character
chk	check
chng	change
chnl	channel
CIDA	channel indirect data address
ck	check
CKD	count, key, and data
clk	clock
CM	current module
cmd	command
Cmd-O	command-out
cmnd	command
cncl	cancel
cnd	condition
cnsi	console
cnt	count
cntr	counter
coax	coaxial cable
CO	convenience outlet
comp	compare
con-con	contingent connection
cond	condition
cons	console

CORV	convenience
corr	correction
CP	circuit protector
СРК	console printer-keyboard
comt	complement
CPU	central processing unit
CPURTY	central processing unit retry register
CR	control register
CS	control storage
CSW	channel status word
CTCA	channel to channel adapter
CTCAY	channel-to-channel adapter X system
CTCAV	channel-to-channel adapter X system
otr	counter
ctrl	
CUA	control unit address
COB	control unit busy
CUE	control unit end
cyc	cycle
cyl	cylinder
D	
<b>D</b>	data
DA	data address
DAT	dynamic address translation
	direct access torage device
450	deuble
	double divisit control
da:	direct current
DCBI	direct control but in
DCBI	direct control bus-in
DCBU	direct control bus out
DCU	disconnect command chaining
	Drect control noid-in
DUFL	De-coupler
DUR	dynamic device reconsiguration
DE	device-end
Cec .	decode, decimal, decrement
DED	double error detect
Del	delay
dest	destination
DE	
	CISK THE
CIAG	Ciagnostic
	aitterence
	do interpretive loop
Dir-In	direct control bus-in
UIT-UUT	direct control bus-out

Disc-I

dist

div

disconnect-in

distribution

division

DL data length dly delay D-Mod D-modifier Doc documentary console DOS disk operating system dply display disabled dsbld duplicate dup

# E EBC

EBCDIC	extended binary coded decimal
	interchange code
EBI	external bus-in
EBO	external bus-out
EC	external control, engineering change
	extended control
ECC	error checking and correction
ECCL	error checking and correction logic
ECNT	error count register
ECSW	extended channel status word
ED	external damage
EDBI	external data bus-in
EDBO	external data bus-out
EM	external damage report mask
env	envelope
EOF	end of file
EOL	end of line
EP	emergency pull (switch)
EPO	emergency power off
EPSW	extended PSW
eq	equal
ERDS	environmental recording data set
EREP	environment recording edit and
	print program
ERP	error recovery procedure
err	error
ev	even
exc	exception
EXCA	external control assembler
EXE CPL	T execute complete
exp	expanded
EXPLS	expanded local storage
ext	external
ext asm	external assembler
ext dst	external register destination
evtint	external interrupt

1	· · · · ·
FBAK	1 backup external word
FBO	file bus-out
FCH	file count register high
FCL	file count register low
FCND	file conditions external word
fdbck	feed back
FDR	file data register
FERR	file error external word
FM	file mask
FF	flip flop
FLS	feature local storage
FM	file mask
FOP	file operation register
F STAT	file status external word
FTAG	file tags external word
FTC	flush through checking
fwd	forward

G	
gen	generate, generator
GM	group mark
gnd 🐰	ground
60	concercil registers

GSTAT	selector channel status external word
grp	group
GR	general registers

# Н

73

HA	home address
HÓV	halt device
hdwr	hardware
hex	hexadecimal
hi	high
HIO	halt input/output
HMRTY	H and M retry registers
HS	hard stop
hwd	hardware
Hy	hertz

# 1

IAR	instruction address register
18	interrupt buffer
IBU	I-register backup
IC	instruction counter
ICC	interface control check
icplt	incomplete
I-cy	instruction cycle
id	identifier
IDA	indirect data address
IDAL	indirect data address list
IDALW	indirect data address list word

if	interface
IFA	integrated file adapter
IFCC	interface control check
IFCU	integrated file control unit
IL .	incorrect length
ILC	instruction length code
IM	input/output extended logout mask
IMPL	initial microprogram program loading
INB	in backward
inc	increment
ind	indicator
INF	in forward
inh	inhibit
inst	instruction, instruct
instr	instruction
intf	interface
intlk	interlock
intr	interrupt
intv	interval
intvn	intervention
invld	invalid
invrt	invert
1/0	input or output
IOCA	input/output communications area
IOEL	input/output extended logout
IPL	initial program load
ISC	integrated storage control
ISK	insert storage key

## job control language

## kilo, relay, key

J JCL

K K KD

keybd

KL

key and data keyboard key length

L	
L 1	length
LC	lower case
LCTL	load control
LD	line driver
LEX	local execute mode
LO	low
Logi	logical
LH	L-register high half
LHM	left-hand margin
LL	L-register low
Ink	link
LR	line receiver
LRA	load real address
LRU	least recently used

1.5	local storage
LSCA	local storage control assembles
LOCA	local storage control storage
Laca	local storage control storage
LSDST	local store destination
ماما	latah

# М

lth

mach	machine
MB	M-register back up
MBO	multiplexer bus-out
MC	machine check
MCAR	machine-check analysis and recording
MCEL	machine-check extended logout
MCH	machine-check handler
MCK	machine-check register
MCKB	machine-check B-register
MCIC	machine-check interruption code
MCKA	machine-check A-register
MCPU	move data in CPU
MCRR	machine-check recovery recorder
MFE	magnetic feedback emitter
MFT	multiple fixed tasks
MG	motor generator
Mid-Pac	middle power package regulator
MIO	move data for I/O
misc	miscellaneous
MLC	machine level control
mod	module
mono	monolithic
MOP	mini operation register
MP	matrix printer
MPX	multiplexer
MRTY	M-retry register
ms	millisecond
us.	microsecond
MS	main storage
MSF	main-storage frame
MST	monolithic systems technology
MTO	multiplexer tags-out
MTI	multiplexer tags in
MUA	multiple unit address
MVT	multiple variable task

### N NA N/L n/o NOP

norm

NPL

ns

next address
new line
normally open
no operation
normal
new product langua
nanoseconds

inverter

## O OBR ОС OE OP

OP-1

OPL

OS

OSC ov

exclusive OR operation operational-in operational OP-O operational-out operating system oscillator over voltage

outboard recorder

overcurrent

# Р РАА

R

PAA	pre-address assembler
P-Bit	parity bit
PCI	program-controlled interrupt
PD	instruction processing damage
PDAR	program damage assessment and repa
PE	print emitter
PER	program event recording
PF	power frame
PG	power gate
PGA	power gate A
PGB	power gate B
PH	polarity hold
PIR	priority interrupt register
PIRM	priority interrupt register mask
POH	power-on hours
POR	power-on reset
pos -	position
prev	previous
PR-KB	printer-keyboard
proc	process
prog	program
prot	protect
PSW	program status word
pt	point
PTLB	page table lookaside buffer
ptr	printer
pty	parity
pwr	power

R	
r	register
RO	record zero
R1	record 1
RAC	remote analysis center
RAS	reliability availability serviceability
RCNT	retry count register
RCS	reloadable control storage
RD	read
RDK	reset diagnostic key

rdy ready ' rec recovery recalibrate recal ref reference regulator, register reg request req required reqd Req-I request-in reverse rev RHM right-hand margin RM. record mark RMS recovery management system rms root mean square RR record ready recovery report RRB reset reference bit Rst reset, restart rtn return rty retry rty fig retry flag

### S SAR

storage address register SCAMPART storage console approaches, manual procedures, and reference timings SCF storage and control frame sch search SCP system control panel SCR silicon controlled rectifier sct sector SD system damage SDBI storage data bus-in SDBO storage data bus-out SDC suppress data check SDK set diagnostic key SDR storage data register, statistical data recorder secondary sec sect sector sei select seld selected Sel-I select-in Sel-O select-out selr selector seq sequence serdes serializer/deserializer SEREP system error record editing program Serv-I service-in Serv-O service-out sht short SI system incidents SIO start input/output SIOF start I/O fast release S/L sense preamplifier and data latch SLI suppress length indication SLT solid logic technology SM synchronous mask sng single

U segment origin special UC upper case S, P, T and L-registers UCW SPTL back-up word for SPTL U/L registers microsecond us. system recovery set or reset set storage key start V status-in STCTL store control volts standard ٧ vac storage val validate status-in vdc STIDC store identification channel VFO store identification processor VMA stacked VOM storage stop single unit address Ŵ suppress suppress-out WB system unavailable time hw word supervisor call WK: service-in WLR switch WΜ word mark selector/block-multiplexer channel Wr write synchronize WS syndrome WTay word-top system WTC Х XFer * unsfer

terminal block test channel timer damage, time delay temporary, temperature terminator transformer T-register high thermal threshold through transfer in channel test input/output translation lookaside buffer time of day time-of-day high word time-of-day low word timing pulse transformer-rectifier, trap

tilt or rotate

timing signal bus-out

trigger

so

spec

SR

S/R

SSK

Stat-I

st

std

stg

STI

STOP

stkd

stor

stp

SUA

supp

SUT

SVC

SVI

SW

sx

sync

synd

sys

T

TB

TCH

TD

temp

term

tfmr

TH

therm

thid

thru

TIC

TIO

TLB

TOD

TODH

TODL

ΤР

TR

T/R

tor

**TSBO** 

Sup-O

SPTL

SPTLB

4 *

unit control word upper or lower volts alternating current volts direct current variable frequency oscillator virtual machine assist feature volt/ohm meter

> word bottom working register wrong length record word separator, word source World Trade Corp.

basic micro-diagnostic group micro microprogram microsecond phase phase control regulator

translate

X-Late

*BAS

μ

μpm

μs

ф

ФCR



PLAN 6

### TIMING CHARTS





GENERAL

0

0

() A3

0

Diag 1-2

e 122

<u>___</u>

1.6 12.

TANK AND AND A TANK



### **On-Page Connector**

1.

÷.

Indicates connection between two parts of the same diagram. Arrow leaving symbol points (line-of-sight) to correspondingly numbered symbol. 4 .5

#### ١, 1.2 **On-Page Connector**

Indicates connection between two parts of the same diagram. Alphameric grid coordinate of complementary connector shown beneath,

### **Off-Page Connector**

Indicates connection between diagrams located on separate pages. Location of correspondingly - lettered symbol shown adjacent, Alphameric grid coordinate may be included,

Function or Process Detailed Elsowhere Indicate in blockwhere the detailed, flow chart of the process is located.

Process

Major Functions or Events

Flow chart block where action is indicatable. (Use and placement of indicator is optional)

Yes Flow chart block where condition branched on is indicatable. (Use and placement of indicator is optional.) Terminal

> Indicates beginning or end of flow peth. *....*

THE PROPERTY OF THE OWNER OF THE PROPERTY OF THE

......

Annotation, commant blogt. . norv note. 17945 THE OF LITTLE OF

### NºC MO

Ξ

No

Ξ

3145 TM PLAN7

## INDEX

A A-byte assembler CPU 89 A-gate layouts STOR 5-6 A-H rotary switch functions CNSL 30 A-local storage data flow CPU 12 A-register description CPU 88 introduction INTR 10 use STOR 3 A-register, B-register and ALU CPU 88 A-register display data flow CNSL 16 indicators (lower roller) CNSL 14 AB backup REC 11 AB retry (ABRTY) REC 11 ABM array card, 16 and 32k STOR 26.1 ABM storage decodes STOR 32.1 ac (automatic configuration) REC 18 ac cable, IFA/2319 IFA 98 ac/dc module, removal and replacement Models FED-1 PWR 58 Models H2-K2 PWR 127 ac (voltage) blowers Models FED-1 PWR 8 Models H2-K2 PWR 100 convenience outlets Models FED-1 PWR 8 Models H2-K2 PWR 104 distribution Models FED-1 PWR 8 Models H2-K2 PWR 104 ACB control circuits CPU 37 examples CPU 36 register description CPU 35 settings CPU 35 accelerate latch CPK 45 accessing (storage) STOR 3 active cap adjustment PWR 52 bias voltage check PWR 54 check PWR 52 ripple check PWR 54 addition of external storage STOR 107 address adjust (see OS/DOS compatibility) adjust, introduction INTR 11 buffer card STOR 3 check boundary (see ACB) checking (SAR) STOR 22 compare control, switch CNSL 32 compare, switch CNSL 18 compare switch, logic diagram CNSL 18 formation, next control word ' MIC 16 mark AM bytes' IFA 17 detect logic IFA 17 of int detect timing IFA 18 detection IFA 16 3410.54 address, mark (continued) restart IFA 17 sync byte IFA 17 address-in CHNL 4 address interface (storage) advanced bi-polar STOR 29 phase 21 STOR 28 address-out CHNL 4 address translation CPU 139 addresses trap (retry) REC 15 addressing BSM STOR 22 local storage scoping procedure CPU 17 adjustments active-cap PWR 52 data separator IFA 15 delay line IFA 15 error detector IFA 15 gate generator IFA 15 MG output voltage Models FED-I PWR 56 Models H2-K2 (see 3047 Power Unit TM) MG regulator overvoltage Models FED-1 PWR 55 Models H2-K2 (see 3047 Power Unit TM) regulators, power service checks Models FED-1 PWR 50 Models H2-K2 PWR 124 reset pulse PWR 53 singleshot VFO IFA 15 storage STOR 73-75 zero detector IFA 15 adr comp match, indicator CNSL 6 adr X-late LRU invalid, indicator CNSL 5 multi-match, indicator CNSL 5 mode, indicator CNSL 5 no match, indicator CNSL 5 advance head trap IFA 77 advanced bi-polar address and data interface (illustration) STOR 18 advanced bi-polar BSM-ECCL-CPU interface (illustration) STOR 18 advanced bi-polar control lines BSM reset STOR 26 BSM select STOR 26 machine reset STOR 26 write (U/L) STOR 26 advanced bi-polar storage addressing configuration STOR 82 boards STOR 106 board pins STOR 102 SDR card STOR 105 write timing STOR 110 advanced checkpoint/restart REC 26 aids (see service aids) alarm audible CPK 61 indicator CPK 6. reset switch CPK 6

ALD references (power) Models FED-I PWR 6 Models H2-K2 PWR 102 allow CE mode, switch IFA 86 alter/display mode indicator CPK 6 operations CPK 54 switch CPK 6 ALD references (power) AM (address mark) bytes IFA 17 AM (asynchronous machine check logout mask) REC 17 analysis, machine check REC 24 arithmetic and logic unit (ALU) description CPU 93 gating (A/B) CPU 91 introduction INTR 10 K-assembler CPU 92 shift gating CPU 92 arithmetic control word byte operation MIC 76 description MIC 68 fullword bit format MIC 70 indirect byte operation MIC 78 operation MIC 80 shift operation MIC 82 type 10 bit format MIC 70 type 11 bit format MIC 70 arithmetic word types (ABCK) MIC 70 array card population chart (PF) STOR 9 and 67 array card storage elements (illustration) STOR 10 array card structure STOR 26.1 array cards (PF) STOR 68 array module STOR 15 array module and chip selection (phase 2I) STOR 15 assembler instructions MIC 70 attachment, 3210 PR-KB CPK 14 attachment, 3215 PR-KB CPK 33 attention (status bit 0) CPK 61 autotransformer (WTC) Models FED-1 PWR 8 Models H2-K2 PWR 104

B (backup) REC 18 14 B A B-byte assembler CPU 90 1 B-gate layouts STOR 4 B-local storage, data flow CPU 14 B-LS source direct address CPU 15 **B**-register description CPU 98.201 11 introduction INTR 10 (111) (1611 4) use STOR 3 backup and retry registers and the second second introduction INTR 10 use REC 10 82812 backward seek test IFA 94 basic C. C. Halling Com STAT control command execution IFA 40 data command execution IFA 41

1 in

basic (continued) IFA operations IFA 35 logic symbology REF4 basic BSMs STOR 7 basic diagnostic group (storage) STOR 70 bias voltage amplitude PWR 15 check PWR 67 ripple amplitude PWR 15 ripple frequency PWR 15 waveform (waveshape) PWR 15 bias voltage supply description PWR 15 bit assignments, IFA externals IFA 11 bit cell (illustration) STOR 16 operation (phase 21) STOR 16 selection (description-phase 21) STOR 23 bit count appendage (BCA) IFA 24 bit ring logic IFA 19 bit ring timing IFA 19 bit select (BS) STOR 16 block multiplexer channel charts, objectives CHNL 98 operations CHNL 89 selector channel INTR 17 semidynamic UCW address assignment CHNL 92 shared UCW assignment CHNL 91 store/load UCW traps CHNL 95 UCW address pointer tables CHNL 92 assignment CHNL 91 assignment registers CHNL 92 pool CHNL 93 blower troubleshooting aid PWR 64 blowers ac Models FED-1 PWR 8 Models H2-K2 PWR 104 off, switch CE3 location PWR 28 description PWR 29 off, switch (Models H2-K2) PWR 110 removal and replacement Models FED-1 PWR 59 Models H2-K2 PWR 128 board decoupling capacitors PWR 22 board select STOR 15 branch and link control word example MIC 34 branch and link or return word bit format MIC 32 description MIC 30 branch and module switch bit format MIC 19 description MIC 18 example MIC 20 branch word . bit format MIC 24 Sam description MIC 22 example MIC 26

3145 TM X-2

branch word, special module switch function MIC 28 branch symbols, control word MIC 89 branching circuits (microprogram) CPU 104 BSM addressing (advanced bi-polar) address and control interface (text and illustration) STOR 26 address control lines, chart STOR 26 24 11 address flow example STOR 27 addressing circuits STOR 25 addressing illustration STOR 25 BSM decode STOR 25 BSM decode chart (PF storage range) STOR 25 BSM clock STOR 19 BSM configuration (addressing and data) advanced bi-polar storage STOR 29 phase 21 storage STOR 28 BSM data flow advanced bi-polar description STOR 42 illustration STOR 42 advanced bi-polar data-bit location chart STOR 43 data and check-bit cabling advanced bi-polar STOR 42 phase 21 STOR 40 phase 21 data-bit location chart STOR 41 description STOR 40 illustration STOR 40 BSM-ECCL CPU interface (phase 21) STOR 17 BSM-ECCL data flow (description) STOR 19 BSM functions STOR 19 BSM interface timing, charts advanced bi-polar STOR 36 phase 21 STOR 36 BSM internal timing, charts advanced bi-polar STOR 39 phase 21 STOR 38 BSM terminators STOR 3 BSMs addressing STOR 22 addressing ranges STOR 4-6 and 6.1 advanced bi-polar STOR 10 array cards phase 21 STOR 8 population chart (advanced bi-polar) STOR 9 bit capacity STOR 7 board layout STOR 10 board select STOR 22 cabling (data and check bits) STOR 40-42 comparison (phase 2I to advanced bi-polar) STOR 7 conceptual STOR 11-12 configurations STOR 28-29 CPU STOR 8 data bit location chart STOR 41 data flow STOR 40-42 interface STOR 28-29 interface timing STOR 36-37 internal timing STOR 38-39 locations STOR 4-6 and 6.1 phase 2I STOR 2 power frame (PF) STOR 9 SIZES advanced bi-polar STOR 9 phase 21 STOR 8 terminator STOR 11-12

buffer register (MB) CPU.103 bulbs (see indicators) bulk voltage amplitude PWR 14 check PWR 66 Second FINE Y description PWR 14 ripple amplitude PWR 14 . 19 frequency PWR 14 table PWR 14 21012 troubleshooting PWR.66 waveform (waveshape) PWR 14 bulk voltage supply description. PWR 14 burst mode (see channel) bus-in CHNL 4 bus-out CHNL 4 bus-cet check (PR-KB) CPK 62 buses (power) flat-wire Models FED-I PWR 3 Models H2-K2 PWR 108 **U** laminar Models FED-1 F R 3 Mode's H2-K2 PWR 108 voltage measurement charts Models FED-I PWR 46 buses (storage) SDBI STOR 3 SDBO STOR 3 busy circuit (phase 21, ABM) STOR 94 indicator (ECC check) CNSL 5 status bit (PR-KB) CPK 62 BYTDST REC 11 byte assembler A CPU 89 B CPU 90 byte counter console file CFA 15 indicator CNSL 7 use IFA 22 byte count logic, IFA IFA 22 byte-mark register description STOR 48 second-level diagram STOR 48 byte-mark register (storage) STOR 48 byte marks STOR 48 byte multiplexer channel (MPX) catalog numbers CHNL 37 channel log REC 20 channel machine dependent log REC 20 command chaining CHNL 43 CSW store CHNL 34 data chaining CHNL 43 data handling CHNL 21 ending procedure CHNL 31 ending status and interruption handling CHNL 30 error handling and logout CHNL 37 error logout CHNL 38 external facilities CHNL 15 functional units CHNL 13 halt device CHNL 22 halt I/O CHNL 22

byte multiplexer channel (MPX) (continued) + 1/O device and UCW addressing CHNL 12 initial selection routine CHNL 26 interface control check CHNL 26 interruptions SHAR !! normal primary ending status CHNL 32 normal secondary ending status CHNL 33 program-controlled interrupt (PCI) CHNL 35 IPL (Initial program load) CHNL 45 local storage CHNL 18 13.132 logout CHNL 36 MBI (multiplexer bus-in) CHNL/15 MBO (multiplexer bus-out) CHNL 15 MTI (multiplexer tags-in) CHNL 15 MTO (multiplexer tags-out) CHNL 15 operations CHNL 19 second-level diagram CHNL 16 share trap data handling CHNL 21 request CHNL 28 start I/O (SIO) description CHNL 22 detailed flowcharts CHNL 24 status interrupt CHNL 31 stop after log CHNL 36 store and load PSW CHNL 44 test channel CHNL 42 test I/O (TIO) CHNL 22 unit control word format CHNL 13 byte-multiplexer channel interruption CHNL 33 byte selection (SDBO) CPU 8 С C-register description CPU 98 force IFA 30 introduction INTR 9 use STOR 3 cable ac IFA 98 mixer board dc IFA 98 R/W coax and selected IFA 98 simplex IFA 98 cancel key (PR-KB) CPK 6 latch (PR-KB) CPK 15

CB trip, indicator

CBs PWR 6-11

sequence

CE panel

CE (customer engineer)

error light IFA 86

IFA/2319 IFA 86

Models FED-I PWR 28

CCW entry and decode IFA 45

mode indicator IFA 86

Models H2-K2 PWR 110

CD (time-of-day clock damage) REC 18

Models FED-I PWR 28

Models H2-K2 PWR 110

and appendicyd Here HAV HE

### address word (CAW) CHNL 3 available interruption CHNL 89 block multiplexer CHNL 89 burst mode defined CHNL 2 operation CHNL 9 bus-in CHNL 4 bus-out CHNL 4 byte-mode operation CHNL 9 byte-multiplexer data flow CHNL 10 check handler routine REC 23 clear I/O CHNL 3 command word (CCW), defined CHNL 3 commands for IFA IFA 8 commands for PR-KB CPK 60 configurations CHNL 2 control check (PR-KB) CPK 61 control words CHNL 3 data check (PR-KB) CPK 61 data rates INTR 16 dependent logout REC 20 end (CE, status bit 4, PR-KB) CPK 61 general description block-multiplexer feature INTR 17 byte-multiplexer INTR 16 data rates INTR 16 integrated file adapter INTR 17 selector INTR 16 halt device (HDV) multiplexer CHNL 22 selector CHNL 68 halt I/O (HIO) multiplexer CHNL 22 selector CHNL 68 ID IFA IFA4 multiplexer CHNL 36 selector CHNL 77 identification REC 19 indirect data address block-multiplexer CPU 170 byte-multiplexer CPU 170 CIDA backup control CPU 172 controls CPU 171 IFA addressing CPU 172 page end detection CPU 171 selector CPU 170 selector channel addressing CPU 172 initial selection sequence CHNL 8 instructions CHNL 3 interruption, available CHNL 89 introduction CHNL 2 logout multiplexer CHNL 36 selector CHNL 77

CF commands CFA /11 39

description CFA 6

use CPU 34

channel

CF pwr on, indicator CNSL 6

CFDAR register

chaining check (PR-KB) CPK 61

channel (continued) mark 0-in CHNL 4 metering controls CHNL 6 multiplexer mode CHNL 2 retry description REC 10 introduction INTR 14 selection controls CHNL 4 sequence codes multiplexer CHNL 36 selector CHNL 37 short CSW CHNL 26 standard interface CHNL 4 status byte (PR-KB) CPK 61 status bytes (IFA) IFA 74 status word (CSW), defined CHNL 3 tags-in CHNL 4 tags-out CHNL 4 test I/O (TIO) multiplexer CHNL 22 selector CHNL 65 unit control word multiplexer CHNL 13 selector CHNL 50 channel checks CHNL 71 channel indirect data address CPU 170 channel logout multiplexer CHNL 36 selector CHNL 77 channel-to-channel adapter feature (CTCA) address compare and data flow X FEAT 28 control command FEAT 20 controls CNSL 36 data flow FEAT 20 description FEAT 20 disable and compatibility FEAT 29 functional units FEAT 28 halt I/O FEAT 27 I/O interface isolation FEAT 26 no operation FEAT 23 on-line/off-line modes FEAT 26 operational characteristics FEAT 20 programming notes FEAT 27 read or read backward FEAT 23 select priority, mode selection and bypass FEAT 29 selection and reset X FEAT 28 sense adapter state FEAT 23 sense command byte FEAT 23 sequence and control X FEAT 28 status, sense, and input B bus-in X FEAT 29 system or selective reset FEAT 24 test I/O FEAT 23 write FEAT 23 write end of file FEAT 23 charts address control lines advanced bi-polar STOR 26 phase 2I STOR 23 array card population advanced bi-polar STOR 67 BSM decode chart advanced bi-polar STOR 25 data-bit location advanced bi-polar STOR 43 check bit and error correction (at ECCL board) STOR 50 charts (continued) phase 21 STOR 41 segment addressing chart phase 21 STOR 22 syndrome decoder STOR 52 system storage capacities STOR 2 check chaining channels CHNL 71 PR-KB CPK 61 channel control channels CHNL 71 IFA IFA 74 PR-KB CPK 61 channel data channels CHNL 71 IFA IFA 74 PR-KB CPK 61 console-file data even-odd CFA 14 extra bit CFA 14 out-of-sync CFA 14 missing bit CFA 14 shift register CFA 14 data parity (PR-KB) CPK 43 flush through external CPU 10 local storage CPU 29 GB parity CHNL 85 hardware (storage) address STOR 72 byte STOR 72 data STOR 72 hardware check STOR 71 store control line parity STOR 72 interface control channels CHNL 71 IFA IFA 75 PR-KB CPK 61 machine REC 8 multiplex CHNL 37 power (see power checks) print emitter sync CPK 43 program channels CHNL 71 IFA IFA 74 PR-KB CPK 61 protection channels CHNL 71 IFA IFA 74 PR-KB CPK 61 retry REC 10 service CPU (clock) CPU 100 console file CFA 18 delay line adjustments (storage) STOR 73-75 PR-KB (see PR-KB Theory-Maintenance Manual) timing (storage) STOR 73-75 unit IFA 74 check bit and error correction chart STOR 50 check-bit correction, description STOR 54 check bit examples STOR 83 checking circuits first sequence Models FED-I stage 1 PWR 40

checking circuits, first sequence (continued) stage 2 PWR 42 first sequence Models H2-K2 stage 1 PWR 118 stage 2 PWR 120 second sequence Models FED-I stage 1 PWR 41 stage 2 PWR 43 second sequence Models H2-K2 stage 1 PWR 119 stage 2 PWR 121 checkpoint/restart REC 26 check reset (CK reset, CE2) switch description Models FED-I PWR 29 Models H2-K2 PWR 112 location Models FED-I PWR 28 Models H2-K2 PWR 110 chip (selection) STOR 15 clock comparator (see clock comparator and CPU timer) CPU (see CPU clock) damage REC 18 IFA logic IFA 16 share cycle logic IFA 31 timing IFA 18 logic diagram, storage STOR 30-31 stop, indicator CNSL 6 storage STOR 22 sync, indicator CNSL 4 time-of-day (see TOD clock) clock comparator and CPU timer instructions FEAT 31 manual mode conditions FEAT 36 set and manual operations FEAT 35 store operations FEAT 34 timer hardware FEAT 32 timer operation FEAT 31 update operation FEAT 33 clock-out CHNL 5 CM (configuration report mask) **REC 17** cmnd reg indicator CNSL 7 cntr match, indicator CNSL 7 codes printer tilt/rotate (T/R) CPK 8 T/R and keyboard, table CPK 9 translation, 3210 PR-KB CPK 10 3210 keyboard CPK 7 3210 transmission CPK 9 colon printing the punctuation character 3210 CPK 11 3215 CPK 31 command chart, console file CFA 12 command execution IFA control cmd IFA 40 IFA data cmd IFA 41 command-out CHNL 4 command reject IFA IFA 46 PR-KB CPK 62

command retry I/O interface sequence CHNL 80 introduction INTR 14 operation REC 10 command retry feature CHNL 80 common region, 1400 compatibility FEAT 4 comparator, clock (see clock comparator) compare circuits IFA 26 compatibility exceptions INTR 14 compatibility feature, 1400/1410 ANUM instruction FEAT 11 BIFLAG instruction FEAT 12 common region FEAT 4 COMP instruction FEAT 11 control byte charts FEAT 10 control storage assignment FEAT 2 DIL and BDIL instruction FEAT 10 EA instruction FEAT 10 local storage assignment FEAT 2 main storage assignment FEAT 2 MCPU instruction FEAT 11 MIO instruction FEAT 11 op-code chart 1401/1460 FEAT 6 op-code chart 1410/7010 FEAT 7 op-codes, 1440 FEAT 8 program debugging inforamtion FEAT 12 compatibility feature 1401/1440/1460 FEAT 2 compatibility feature 1410/7010 FEAT 2 component labeling (component Id) Models FED-I PWR 3 Models H2-K2 PWR 100 locations Models FED-1 PWR 5 Models H2-K2 PWR 102 removal Models FED-I PWR 57 Models H2-K2 PWR 126 replacement Models FED-1 PWR 57 Models H2-K2 PWR 126 component locations Models FED-I PWR 6 Models H2-K2 PWR 102 condensed data flow, 3215 CPK 33 condition code validity REC 18 configurations advanced bi-polar STOR 82 BSM address and control interface STOR 28-29 basic STOR 7 PR-KB CPK 3 3145 STOR 80 3345 STOR 80 console file adapter CFA 2. byte control CFA 6 byte counter description CFA 6 byte counter indicator CFA 15 byte format CFA 4 CFDA track/selector table CFA 4 clock CFA 8

console file (continued) command byte, CFA.4. command chart CFA 12 38.96.5 command register. description CFA 6 indicator CFA 15 commands CFA 11 compare circuits CFA 6 control commands CFA 11 CPU clock start CFA 15 data and clock bit timing CFA 4 data byte CFA 4 data checks CFA 14 data flow CFA 1 data register (CFDR) CFA 6 disk address byte CFA 4 disk-address register CFA 15 disk address register (CFDAR) CFA 6 disk format CFA 4 error checks CFA 14 even-odd check CFA 14 extra bit check CFA 14 head control and track accessing CFA 6 IMPL example CFA 16 indicators CNSL 7 introduction CFA 2 operation commands CFA 11 out-of-sync/missing bit check CFA 14 pause CFA 15 register display switch CNSL 7 registers, indicators CNSL 7 removal and replacement CFA 18 sector format CFA 4 sector ready latch CFA 10 shift register description CFA 6 parity error CFA 14 timing CFA 9 timing charts, shift register CFA 9 voltages stage 1 PWR 47 stage 2 PWR 49 console-file register display CNSL 31 console layout CNSL 2 continuing scan commands IFA 70 control address set. key CNSL 32 control and data flow (PR-KB) 3210 CPK 15 3215 CPK 34 control command execution IFA 49 control commands IFA 50 control. data transfer (PR-KB) CPK 49 control keys and indicators CPK 6 control mode (see extended control mode) control. new line, 3210 CPK 23 control panel (see CE sequence panel) control. read/write cycle, 3210 CPK 23 control register assignments CPU 106 decodes. I-cycles CPU 64 description CPU 106 machine check REC 17 valid REC 18

control register (continued) 0 CPU 107 CPU 107 2 8 bit assignment CPU 107 uses CPU 130 14 (CR 14) REC 17 15 (CR 15) REC 17 3210 CPK 15 3215 CPK 33 control, stepper motor CPK 44 control storage IFA assignments IFA 12 physical description STOR 19 control-unit end (status bit 2, PR-KB) CPK 61 control word access MIC 4 address generation MIC 12 bit definition arithmetic indirect byte MIC 73 arithmetic type 10 byte MIC 70 arithmetic type 10 fullword MIC 71 arithmetic type 11 MIC 72 branch and link or return word MIC 32 branch and module switch MIC 19 branch word MIC 24 indirect byte MIC 73 storage word (K-addressable) MIC 50 storage word (non K-addressable) MIC 48 word-move word MIC 40 branch MIC 24 branch and link or return word MIC 30 branch and module switch MIC 19 branch symbols MIC 12 descriptions MIC 15 examples arithmetic, byte operation MIC 76 arithmetic fullword MIC 80 arithmetic indirect byte MIC 75 arithmetic, indirect byte operation MIC 78 arithmetic, shift operation MIC 82 branch and link MIC 34 branch and module switch MIC 20 branch word, set/reset function MIC 26 branch word, module switch function MIC 28 return word MIC 36 storage read halfword MIC 52 storage word, direct control MIC 60 storage word, store inder mask and dec ement count MIC 64 storage word, TB function MIC 56 word-move word MIC 42 functions MIC 15 next address generation MIC 12 stat set symbols MIC 84 storage word MIC 44 storage word forms MIC 46 types MIC 15 word-move word MIC 38 controls and indicators power Models FED-I PWR 28 Models H2-K2 PWR 110

controls and indicators (continued) PR-KB keys (switches) CPK 6 system control panel (see CNSL section) convenience outlet voltages Models FED-1 PWR 8 Models H2-K2 PWR 104 conversion emitter pulse CPK 40 power PWR 8 cooling Models FED-I PWR 3 Models H2-K2 PWR 100 core (see storage) correction check-bit STOR 54 count controls IFA 22 count externals IFA 22 count field IFA 22 counter decode IFA 22 counter logic IFA 22 counts PE timing pulse CPK 41 print emitter timing CPK 41 retry REC 12 single-bit STOR 52 CPU A and B gate voltages stage | PWR 46 stage 2 PWR 48 ac distribution PWR 8 ac-to-dc conversion Models FED-I PWR 10 Models H2-K2, stage 1 PWR 105 Models H2-K2, stage 2 PWR 106 BSM (interface) STOR 17-18 configuration INTR 13 cycle times (see CPU clock) dependent logout REC 19 external word CPU 34 frame electrical components Models FED-I PWR 6 Models H2-K2 PWR 102 gate labeling Models FED-1 PWR 10 Models H2-K2 PWR 102 hardware CPU 103 high priority CPU 42 identification REC 19 independent logout REC 18 indicator CNSL 4 indicator (power check) Models FED-I PWR 28 Models H2--K2 PWR 110 status, indicators CNSL 6 thermal checks PWR 69 CPU ac-to-dc conversion Models H2-K2, stage 1 PWR 105 Models H2-K2, stage 2 PWR 106 stage 1 PWR 10 stage 2 PWR 11 CPU clock card locations CPU 100 checks and adjustments CPU 100

### 3145 TM X4

CPU clock (continued) delay card plugging CPU 101 start indicator CNSL 7 CPU dc outputs for A- and B-gates, stage 1 Models FED-1 PWR 46 Models H2-K2 PWR 124 CPU dc outputs for A-gate, stage 2 flat-wire bus version Models FED-I PWR 48 Models H2-K2 PWR 124 CPU dc outputs for A-gate, stage 2 U laminar bus version Models FED-1 PWR 48 Models H2-K2 PWR 124 CPU dc outputs for B-gate main storage Models H2-K2 PWR 124 stage 1 PWR 46 stage 2 PWR 48 CPU diagnostic hardware block A local storage DIAG 20 enable generate address DIAG 28 enable generate controls DIAG 28 invert Z-reg parity bits DIAG 20 stop on machine check DIAG 20 CPU select pulse adjustment STOR 73 description STOR 33 ECCL board To relationship STOR 33 CPU storage STOR 2 CPU to ECCL interface (advanced bi-polar) STOR 18 CPU timer (see clock comparator) CPURTY REC 11 CR (control registers valid) REC 18 CS adr, indicator CNSL 6 CSW CHNL 3 cycle control, 3210 read/write CPK 22 cyclic check IFA 25 controls IFA 24 logic IFA 24 cyclic-code hardware register IFA 25 register IFA 24 cylinder address register test IFA 29 concept IFA 4 select switch IFA 86

## D

D-register description CPU 95 introduction INTR 10 damage external REC 18 instruction processing REC 18 interval timer REC 16 report REC 8 report REC 8 report mask REC 17 system machine check interrupt REC 16 subclass REC 18 damage (continued) time-of-day clock mach:: e check interrupt REC 16 subclass REC 18 timer REC 18. DAT (see dynamic address translation) data bit location chart STOR 41 check STOR 72 command execution IFA 41 field IFA 6 indicator CNSL 7 late CPK 47 length (DL) IFA 6 parity check (PR-KB) CPK 43 parity control IFA 23 rates channels INTR 16 records IFA 7 registers, 3215 CPK 33 request controls IFA 28 storage IFA 4 transfer clock IFA 26 data and check bit cabling STOR 40-42 data and control registers, 3210 CPK 15 data chaining (illustration) STOR 18 data flow basic IFA IFA 4 condensed, 3215 CPK 33 ECCL data flow STOR 21 fetch operation STOR 59 storage fetch operation STOR 59 main storage-to-ECC STOR 21 read description STOR 46 store operation STOR 61 write description STOR 46 store operation STOR 61 3145 INTR 7 3210 CPK 15 3215 CPK 34 data handling routine CHNL 31 data-in (interface line) description CHNL 4 introduction INTR 15 data interface (storage) advanced bi-polar STOR 29 phase 21 STOR 28 data-out (interface line) description CHNL 4 introduction INTR 15 data-transfer control IFA IFA 23 PR-KB CPK 49 dc profile PWR 65 dc voltage adjustments Models H2-K2 PWR 124 stage 1 PWR 46 stage 2 PWR 48 checks (see power checks)

dc voltage (continued) conversion CPU PWR 10 CPU (Models H2-K2), stage 1 PWR 105 CPU (Models H2-K2), stage 2 PWR 106 PF PWR 12 PF (Models H2-K2) PWR 107 distribution CPU PWR 10 CPU (Models H2--K2) stage 1 PWR 105 CPU (Models H2-K2), stage 2 PWR 106 PF PWR 12 PF (Models H2-K2) PWR 107 indicators (regulator) Models H2-K2 PWR 110 stage 1 PWR 28 stage 2 PWR 30 measurements Models H2--K2 PWR 124 stage 1 PWR 46 stage 2 PWR 48 profile PWR 65 ripple check PWR 66 service checks Models H2-K2 PWR 124 stage 1 PWR 46 stage 2 PWR 48 decode CCW entry IFA 45 delay line adjustments and checks STOR 73-75 description STOR 19 delay line cards card No. 1 STOR 73-75 card No. 2 STOR 73-75 TD jumpering examples STOR 73-75 delay lines 1 and 2 card layouts STOR 73-75 delta 200 ns index IFA 32 dependent logout REC 19 destination addressing (LS) CPU 11 destination control diagram EXPLS CPU 28 destination look ahead (L/S) CPU 11 device-end (DE status bit 5 PR-KB) CPK 62 diag (external word) CPU 34 diag stop, indicator CNSL 4 diag0 register DIAG 12 diag1 register DIAG 15 diagnose instruction control storage full-recording mode REF 7 control storage threshold mode REF 7 control storage quiet mode REF 7 load patch words REF 7 main storage full-recording mode REF 7 quiet mode REF 7 PSW restart REF 7 diagnostic addresses 0-6 IFA 11 diagnostic control SAR DIAG 21 diagnostic/console-file controls rotary switch CNSL 28 diagnostic hardware ABRTY group 1 DIAG 15 ABRTY group 2 DIAG 22 ABRTY group 3 DIAG 26 diagnostic parity DIAG 18

diagnostic hardware (continued) diagnostic ripple DIAG 19 early delay function DIAG 26 general information DIAG 2 inhibit print DIAG 38 matrix printer lines to TI DIAG 37 MPX diagnostic control DIAG 17 PE envelope diagnostic indicators IFA 11 diagnostic key DIAG 12 diagnostic local storage assignment DIAG 35 diagnostic procedure (MST regulators) PWR 66 diagnostic registers DIAG 12 diagnostic service signal (GA, A-, KOD) DIAG 32 diagnostic strobe CPK 18 diagnostic tests BAB5 DIAG 5 BEA6 DIAG 6 EGE7 DIAG 8 EJD5 DIAG 33 IFA inline indicators IFA 87 test decode IFA 88 trap IFA 74 unsafe condition test IFA 91 diagram legend IFA 34 direct byte addressing definition MIC 16 direct control feature data flow and controls FEAT 17 flow chart, write and read direct FEAT 18 general information FEAT 14 interface lines FEAT 14 read-direct instruction FEAT 14 signals originating outside the CPU FEAT 16 signals originating within the CPU FEAT 15 timings FEAT 19 write direct instruction FEAT 14 direct word addressing definition MIC 16 disconnect in (interface line) description CHNL 4 introduction INTR 5 disk adr reg, indicator CNSL 7 disk drive, 23 FD CFA 2 disk pack, IFA/2319 IFA 4 disk speed diag test IFA 90 disk storage select logic IFA 33 display assembler out indicator CNSL 12 display, key CNSL 32 display operations CPK 54 DK-register (EXPLS-7C) CPU 22 DOS error-recovery procedures REC 25 recovery management support REC 24 restart facilities REC 26 RMS REC 24 DOS-checkpoint/restart REC 26 DOS-emulator (see OS/DOS compatibility) double-bit error isolation STOR 99 double-bit errors STOR 52 downshift, (stepper) motor CPK 46 dual-level MST regulator operation PWR 21 MST regulator purpose PWR 21

dual-level (continued) phase-controlled regulator adjustment and scoping procedure PWR 54 phase-controlled regulator scoping procedure PWR 57 dual-level supply removal and replacement PWR 32 dynamic address translation (DAT) CPU 139 address match basic CPU 140 match-no-match CPU 157 address paging CPU 142 basic operation (flowchart) CPU 156 change bits CPU 150 control registers 0, 1 CPU 148 develop address CPU 141 errors CPU 162 exercise CPU 164 GGST routine CPU 158 introduction CPU 139 load DAT control reg (flowchart) CPU 149 load real address CPU 161 LRU matrix CPU 152 NP regs CPU 154 OS/DOS compatibility and DAT active CPU 147 page number compare CPU 155 page table entry CPU 146 real address formation CPU 159 reference bits CPU 150 reset reference bit CPU 162 segment and page size CPU 144 segment number compare CPU 155 segment table entry CPU 146 translation lookaside buffer (TLB) CPU 151 working reg CPU 153

# E

for 3210 graphics CPK 10 matrix code translation tilt/rotate location table CPK 11 ECC (error check and correction) busy, indicator CNSL 5 check bit and error correction chart STOR 50 DBL bit, indicator CNSL 5 hdw, indicator CNSL 5 indicators (see CNSL section) introduction STOR 3 logic diagram STOR 49 operation STOR 48 ECCL (see error check and correction) ECCL and BSM timing, description STOR 33 ECCL board layouts 01B-A3 (CPU) STOR 56 03A-A3 (PF) STOR 57 ECCL board external (3145/3345) STOR 108 ECCL check bit and error correction chart STOR 50 ECCL compare, error detect, and decode (second level) STOR 49 ECCL data flow STOR 21 ECCL delay lines advanced bi-polar description STOR 35 ECCL board timing chart STOR 35 · * hardware service aid (FL and AR) STOR 35

ECCL delay lines (continued) phase 21 description STOR 34 ECCL board timing chart STOR 34 second level STOR 34 ECCL to BSM interface (advanced bi-polar) STOR 18 ECCL-to-ECCL interface STOR 18 ECCL to storage interface (phase 21) STOR 17 ED (external damage) REC 18 EM (external damage report mask) REC 17 emergency power off emergency pull switch CNSL 35 EPO sequence PWR 38 function Models H2-K2 PWR 113 stage 1 PWR 29 stage 2 PWR 31 mechanical reset PWR 38 operation CNSL 35 3047 (see 3047 Power Unit Theory-Maintenance Manual) emitter pulse conversion CPK 40 enable system clear, key CNSL 32 end switch (PR-KB) CPK 6 ending an alter/display operation CPK 54 ending operation IFA 74 ending sequence IFA 76 environment recording edit and print program REC 2 EPO PWR 38 EPSWA external CPU 34 equipment check (PR-KB) CPK 62 erase command IFA 57 erase gate IFA 27 EREP REC 26 detector, IFA VFO IFA 14 disable, switch IFA 86 display, IFA 2319 IFA 86 handling channel retry INTR 14 command retry INTR 14 detection STOR 20 error checking and correction (ECC) INTR 14 handling STOR 20 introduction REC 2 logging STOR 20 microprogram instruction retry description REC 9 introduction INTR 14 error check and correction comparison and error correction, description STOR 48 data flow STOR 21 data flow concept STOR 48 description STOR 48 ECCL check bit and error correction chart STOR 50 functional areas, description STOR 48 read and write generators, description STOR 48 second level STOR 49 syndrome decode, description STOR 48 syndrome decoder chart STOR 52 error checking and correction (ECC) board layout STOR 56-57 check bit and error correction chart STOR 50 check-bit correction STOR 54

error checking and correction (ECC) (continued) data flow concepts STOR 48 double-bit errors STOR 52 error-type decode STOR 53 functional areas STOR 48 introduction INTR 14 parity correction STOR 52 read and write generators STOR 48 single-bit error correction STOR 52 single-bit errors STOR 52 syndrome decoder chart STOR 52 error correction, description check-bit STOR 54 double-bit STOR 52 parity STOR 52 single-bit STOR 52 error detection, description STOR 52 double-bit STOR 52 single-bit STOR 52 error handling (description) STOR 20 error isolation, double-bit STOR 99 error logging (description) STOR 20 error messages (PR-KB) CPK 54 error override switch (CE2) location Models H2-K2 PWR 110 stage 1 PWR 28 stage 2 PWR 30 description Models H2-K2 PWR 112 stage 1 PWR 29 stage 2 PWR 31 error-recovery procedures REC 25 error retry description REC 8 error retry timing REC 15 error trap sequence IFA 78 test IFA 96 error type decode STOR 35 error type decoder description STOR 19 second level STOR 53-54 exe cplt, indicator CNSL 6 execute phase (I-cycles) CPU 82 expanded external assembler diagram CPU 31 expanded local storage (EXPLS) CPU 19 destination control diagram CPU 28 destinations CPU 27 gating CPU 20 general information INTR 11 map CPU 21 source gating CPU 23 source gating diagram CPU 24 source gating examples CPU 25 EXTDST REC 11 extended control mode control registers CPU 131 EC PSW CPU 131 feature mask CPU 131 interrupt codes CPU 138 permanent storage assignments CPU 134 PSW interchange CPU 136 store then mask instructions CPU 138

extended diagnostic group (storage) STOR 70 extended interruption information REC 19 extended logout IFA IFA 75 length REC 18 mask REC 17 pointer REC 17 external address formation MIC 17 assembler diagram CPU 30 facilities assignment and index CPU 32 description CPU 29 general information INTR 13 external storage considerations STOR 104 external words (see registers) externals IFA IFA 11 map CPU 32 F F-gate (mixer board) PWR 59 FA (failing storage address valid) REC 18 FCH-register IFA 22 FCL-register IFA 22 FDR-register IFA 23 features 3145 Models GE-1 INTR 3 3145 Models H2-K2 INTR 4 fetch operation data flow STOR 59 description STOR 19 fetch sequence CPU 69 field-count controls IFA 22 file data coax IFA 33 file-data register IFA 23 file-mask algorithm IFA 50 bit assignment IFA 12 filter installation precaution PWR 64 filters location Models FED-I PWR 59 Models H2-K2 PWR 128 removal Models FED-1 PWR 59 Models H2-K2 PWR 128 replacement Models FED-I PWR 59 Models H2-K2 PWR 128 firing circuits, 3215 print magnet CPK 42 fix, microprogram temporary MIC 85 flat-wire bus Models FED-I PWR 26 Models H2-K2 PWR 108 floating ground PWR 73 floating-point registers valid REC 18 flow charts, IFA CCW entry IFA 45 cylinder address reg test IFA 89 diag error trap IFA 96

flow charts, IFA (continued) diag test decode IFA 88 diag unsafe test IFA 91 disk speed test IFA 90 ending sequence IFA 76 error trap IFA 78 halt I/O IFA 84 index trap IFA 77 initial selection IFA 42 interrupt IFA 80 load IPL IFA 66 micromonitor control DIAG 35 read data IFA 65 read HA IFA 64 read/write diagnostic IFA 92 recalibrate IFA 54 scan IFA 73 search ID IFA 71 search KD IFA 73 search key IFA 72 seek commands IFA 52 seek tests IFA 94 sense IFA 48 set file mask IFA 51 space count IFA 55 storage (see data flow) test I/O IFA 82 write CKD IFA 60 write data IFA 59 write HA IFA 58 flush-through check (FTC) INTR 10 FM (asynchronous fixed logout mask) REC 17 FMOD-register IFA 33 FOP-register IFA 21 force count (odd) register L/S CPU 15 forward operation multiple characters CPK 52 single character CPK 51 forward seek test IFA 94 FP (floating point registers valid) REC 18 frame ground PWR 72 FTC register CPU 96 flow charts (see data flow) functional areas (storage) address control line chart (phase 2I) STOR 23 addressing circuits STOR 22 array card logical addressing STOR 22 array module chip selects STOR 22 bit-cell selection (phase 2I) STOR 23 board select STOR 22 BSM addressing (phase 2I) STOR 22 description STOR 22 SAR address checking STOR 22 storage address register (SAR) STOR 22 functional operations, description fetch control storage STOR 58 data flow STOR 59 main storage STOR 58 store data flow STOR 61

functional operations, description (continued) operation STOR 60 functional units A- and B-registers CPU 88 arithmetic and logic unit CPU 88 backup and retry registers CPU 103 C-register CPU 98 console file (see CFA section) console printer-keyboard (see CPK section) D-register CPU 95 expanded local storage CPU 19 external facilities CPU 29 H-register CPU 41 I-cycle hardware CPU 47 IFA (see IFA section) introduction INTR 6 local storage CPU 10 M-register CPU 102 MB-register CPU 103 N-register CPU 103 SDBO assembler CPU 8 SPTL-registers CPU 4 storage (see STOR section) Z-register CPU 94

## G

GA functions, branch word MIC 25 GA set/reset functions IFA 13 gap sensor, VFO IFA 14 gate generator, VFO IFA 14 gate layouts-storage STOR 4 gated attention trap IFA 74 gating, T1 register input 3215 CPK 35 GCL register CHNL 88 GDRL (EXPLS) CPU 22 GDRL register CHNL 88 general description IFA 4 3145 INTR 2 general information power Models FED-1 PWR 3 Models H2-K2 PWR 101 storage control storage (size) STOR 2 CPU models STOR 2 storage gates STOR 2 storage sizes STOR 2 general-purpose registers CPU 16 general registers (GR) valid REC 18 generators (storage logic) parity-out (description) STOR 19 read (description) STOR 19 syndrome (description) STOR 19 write (description) STOR 19 ground loop check PWR 74 grounding principles PWR 74

# H

H-register bit functions CPU 41 detailed description CPU 38 general information INTR 10 halt device IFA 84 IFA IFA 84 multiplexer CHNL 22 selector CHNL 68 halt I/O operation IFA 84 IFA IFA 84 multiplexer CHNL 22 selector CHNL 22 halt word selection (SDBO) CPU 8 hard machine checks interrupts REC 16 softward recovery REC 25 hardware checks-storage byte check STOR 72 data check STOR 72 hardward check STOR 71 storage-control-line parity check STOR 72 hardware functions, I-cycle CPU 49 hardware recovery logic REC 3 hardware test, 3215 CPK 32 hdw indicator (ECC check) CNSL 5 head control and track accessing CFA 6 heads extended control line IFA 97 heads extended indicator PWR 28 head-select switch IFA 86 high trap IFA 74 HM backup REC 11 HM retry (HMRTY) REC 11 hold-out CHNL 5 home-address field IFA 6 HS (hard stop) REC 17

## I

I-buffers, introduction INTR 11 I-cycles adder CPU 62 adder carry logic CPU 62 address generation CPU 63 address generation and control decode CPU 65 ALD references CPU 56 align routine entry CPU 73 exit CPU 75 alignment CPU 54 alignment (purpose) CPU 74 branch loop timing chart CPU 80 calculate operand addresses CPU 54 control register bits CPU 64 control register decodes CPU 64 control-word DFOC (hardware functions) CPU 78 control-word forced functions CPU 64 CPU low request (during I-cycles) CPU 66 data flow chart CPU 58 entry CPU 53

I-cycles (continued) error conditions CPU 66 fetch sequence CPU 69 functions CPU 48 further fetch (example) CPU 69 further fetch sequence CPU 69 general description CPU 68 hardware control line generation CPU 64 description CPU 57 functions CPU 49 indicator CNSL 5 locations CPU 56 I-buffers CPU 59 IBU-register CPU 61 IMM byte register CPU 60 indicator CNSL 6 initial addresses CPU 53 instruction RR-ADD instruction CPU 81 RX-add with double indexing and alignment CPU 83 RX-execute instruction CPU 86 SS-MVC instruction CPU 84 introduction CPU 47 I-phase/E-phase objectives CPU 52 I-register CPU 62 key register CPU 61 microcode control hardware loading of I-buffers CPU 53 flow charts CPU 53 hardware functions CPU 50 micro module assignment CPU 52 op-register CPU 60 prefetching (example) CPU 69 perform prefetches CPU 54 program modification CPU 76 RR instruction sequence CPU 70 RX instruction sequence CPU 71 RX, RS, SI instruction sequence CPU 71 share cycle (during I-cycles) CPU 66 SS instruction sequence CPU 72 storage correction cycle (during I-cycles) CPU 66 storage fetch operations (examples) CPU 69 timing chart (I, IBU, TR, updating) CPU 62 timings CPU 67 TR-register CPU 62 U-register CPU 60 V-register CPU 61 W-register CPU 61 I-register (EXPLS 50) description CPU 21 use CPU 62 IA (instruction address validity) REC 18 IBU register (EXPLS-54) description CPU 21 use CPU 61 ICS-register (EXPLS-56) CPU 21 IFA abbreviation list IFA 3 adjustments (IFA) stage 1 PWR 47 stage 2 PWR 49 basic operations IFA 35

IFA (continued) cables ac IFA 98 mixer board dc IFA 98 R/W coax and selected IFA 98 simplex IFA 98 CE panel IFA 86 channel ID IFA 4 checks (see check) clock and bit ring IFA 18 commands IFA 8 configurations IFA 2 control-storage assignments IFA 12 data flow IFA 4 data-flow index IFA 10 description basic IFA 4 power IFA 97 diagnostic functions approach DIAG 44 CE panel tests DIAG 54 FAT byte assembler DIAG 46 GA microword decode DIAG 44 hardware DIAG 45 lines and latches gate to diagnostic addresses DIAG 46 record format DIAG 51-52 VFO adjustable delay cards (2319-A01) DIAG 53 wrap tests DIAG 54 diagnostic indications IFA 11 extended log REC 21 externals diagnostic address (FAT) IFA 11 diagnostic bit assignment IFA 11 diagnostic FAT assembler DIAG 46 GA set/reset functions controls IFA 13 GA function charts DIAG 44 halt I/O operation IFA 84 heads extended IFA 97 high priority CPU 43 interface, 2319 IFA 98 interrupt routine IFA 80 introduction IFA 1 local-storage assignments IFA 12 low trap CPU 43 mini-op control IFA 8 power supplies IFA 97 sense bytes bus-out parity check IFA 46 command reject IFA 46 cyclic code check IFA 46 data check IFA 46 data check-count field IFA 46 end-of-cylinder IFA 46 equipment check IFA 46 file protected . IFA 46 format IFA 46 intervention required IFA 46 invalid sequence IFA 46 missing address mark IFA 46 multi-module select IFA 47 no record found IFA 46

3145 TM X-8

IFA, sense bytes (continued) online IFA 47 overflow incomplete IFA 46 overrun IFA 46 pack change IFA 47 ready IFA 47 seek check IFA 46 seek incomplete IFA 47 selected status IFA 46 sense byte 0 IFA 46 sense byte 1 IFA 46 sense byte 2 IFA 46 sense byte 3 IFA 47 sense byte 4 IFA 47 sense byte 5 IFA 47 SERDES check IFA 46 track condition IFA 46 track overrun IFA 46 unsafe IFA 46 unselected status IFA 46 write current sense IFA 47 share-cycle controls IFA 30 start I O operation IFA 42 status indications IFA 9 test I O operation IFA 82 transfer in channel (TIC) IFA 8 voltages. stage 1 PWR 47 stage 2 PWR 49 ILC CPU 47 IM (input output extended logout mask) REC 17 IMM byte register CPU 60 IMPL. check indicator (ISC) CNSL 37 control storage load STOR 2 example CFA 16 operation CNSL 34 370 microprogram load REF 15 IMPL req'd indicator CNSL 6 IN (interrupt) external word CPU 34 incorrect length (IL PR-KB) CPK 61 independent logout REC 18 index control logic IFA 32 controls IFA 32 trap IFA 74 trap to advance head IFA 77 indicatable keys power-on key CNSL 34 start console-file key CNSL 32 indicator byte IFA 24 indicators A-register display CNSL 14 adr comp match CNSL 6 adr x-late mode CNSL 6 alarm CPK 6 alter display mode CPK 6 busy (ECC check) CNSL 5 byte cntr CNSL 7 cancel CPK 6 CB trip Models H2-K2 PWR 110

indicators, CB trip (continued) stage | PWR 28 stage 2 PWR 30 CE error (IFA) IFA 86 CE mode (IFA) IFA 86 CF power on CNSL 6 clock stop CNSL 6 clock sync CNSL 4 cmnd reg CNSL 7 entr match CNSL 7 console-file registers (0-7) CNSL 7 corr cycle CNSL 6 CPU power check PWR 28 power check (Models H2-J2) PWR 110 system check CNSL 4 CPU clock start CNSL 7 CS adr CNSL 6 data CNSL 7 dbl-bit (ECC check) CNSL 5 diag stop CNSL 4 disk adr reg CNSL 7 dply asmblr out CNSL 10 error display (0-7) IFA 86 exe cplt CNSL 6 heads extended (PF) PWR 28 hdw (ECC check) CNSL 5 I-cvcle CNSL 6 I-cycle hdw (system check) CNSL 5 I/O infc dsbld CNSL 36 IMPL req'd CNSL 6 intvn read CPK 6 load CNSL 8 log pres CNSL 6 LRU inval (adr x-late check) CNSL 5 M-reg lamp CNSL 4 man CNSL 8 MG check Models H2-K2 PWR 110 stage 1 PWR 28 stage 2 PWR 30 MS: (storage and control frame indicator) stage 1 PWR 28 stage 2 PWR 20 multi-match (adr x-late check) CNSL 5 no match (adr x-late check) CNSL 5 pause CNSL 7 ΡF Models if2-K2 PWR 110 stage 1 PWR 28 stage 2 PWR 30 power check Models H2-K2 PWR 110 stage 1 PWR 28 stage 2 PWR 30 power-on complete Models H2-K2 PWR 110 stage 1 PWR 28 stage 2 PWR 30 power on (PR-KB) CPK 6 power on start

indicators, power on start (continued) Models H2-K2 PWR 110 stage 1 PWR 28 stage 2 PWR 30 power (system check) CNSL 5 probe(IFA) IFA 86 proceed CPK 86 regulators CPU PWR 28 CPU (Models H2-K2) PWR 110 MSF PWR 28 PF PWR 28 PF (Models H2-K2) PWR 110 request pending CPK 6 retry CNSL 4 SAR (parity check) CNSL 4 SDBI (parity check) CNSL 4 SDBO (parity check) CNSL 4 set chan (system check) CNSL 4 share cycle CNSL 6 sng ECC CNSL 6 sng ECC thld CNSL 6 stor byte marks (parity check) CNSL 4 stor control lines (parity check) CNSL 4 stor ctrl lines (parity check) CNSL 4 stor prot (parity check) CNSL 4 stor I cycle CNSL 6 sys CNSL 8 test CNSL 8 therm check PWR 28 therm (system check) CNSL 5 TOD clock invld CNSL 31 trap 1 cycle CNSL 6 trap 2 cycle CNSL 6 wait CNSL 8 indirect addressing (LS) CPU 13 indirect byte addressing definition MIC 16 indirect word addressing definition MIC 16 initial microprogram program load (impl) CFA 2 initial power-on sequence description Models FED-I PWR 34 Models H2-K2 PWR 114 operation Models FED-1 PWR 34 Models H2-K2 PWR 114 initial regulator voltage adjustment PWR 40 initial selection IFA 42 inline diagnostics IFA 87 inline test sequence IFA 87 input/output logout mask REC 17 RMS REC 26 insert storage key STOR 63 instruction address validity REC 18 cycles (see I-cycles) diagnose REF 7 formats CPU 47 length codes CPU 47 monitor call CPU 130

instruction (continued) processing damage REC 16 store channel ID REC 19 store CPU ID REC 19 instructions channel CHNL 3 load control REC 17 retry, microprogram REC 8 store control REC 17 INTA-register CPU 34 INTB-register CPU 34 interface A and B switches (ISC) CNSL 37 cabling ground wires PWR 72 control check (PR-KB) CPK 6 CTCA CNSL 36 IFA/2319 IFA 98 ISC CNSL 37 operation summary CHNL 8 standard INTR 15 integrated file adapter (see IFA) integrated storage control features FEAT 37 functional details FEAT 37 switches and indicators CNSL 3 interlocking (cycle) CPK 49 interrupt code machine check REC 18 summary REC 22 table CPU 138 key CNSL 34 routine IFA 80 interruptions cause REC 16 machine check REC 16 occurrence REC 18 interval timer damage REC 16 description CPU 114 interrupt CPU 114 switch description CNSL 31 use CPU114 intvn req'd CPK 6 invalid address (PR-KB) CPK 54 invalid character (PR-KB) CPK 54 1/0 communications area CHNL 36 device and UCW addressing CHNL 12 extended logout mask REC 17 extended logout pointer multiplexer CHNL 36 selector CHNL 77 hold, switch (CE5) description PWR 29 description (Models H2-K2) PWR 112 location PWR 28 location (Models H2-K2) PWR 110 INFC DSBLD, indicator (CTCA) CNSL 36

I/O (continued) interface A and B switches (ISC) CNSL 37 CTCA CNSL 36 ISC CNSL 36 INTFS DSBLD indicator (ISC) CNSL 37 ISC feature switches and indicators CNSL 37 ISK (insert storage key) STOR 63 off, switch (CES) description PWR 29 description (Models H2-K2) PWR 112 location (Models H2-K2) PWR 110

## J

J-K latch operation CPK 48

## K

K-assembler, ALU CPU 92 KE (key in storage error uncorrected) REC 18 key (see storage protect) key field IFA 6 key in storage error uncorrected REC 18 key length (KL) IFA 6 key register CPU 61 key switch (CE key switch) CNSL 35 keyboard EBCDIC location table CPK 11 data to T1 DIAG 36 read operation, 3215 CPK 38 test (T) mode CPK 54 3210 CPK 7 3215 CPK 27 keys (see switches) keys-store and display CNSL 32

# Ľ

L-register CPU 7 lamp test, switch console CNSL 31 IFA (CE panel) IFA 86 PF (CE panel) Models H2-K2 PWR 110 stage 1 PWR 28 stage 2 PWR 30 lamps (see indicators) latch accelerate CPK 45 J-K operation CPK 48 shift CPK 45 stop CPK 46 stop latch buffered CPK 46 late data CPK 47 legend, IFA diagrams IFA 34 LG (logout valid) REC 18 lights (see indicators) limited channel logout multiplexer CHNL 36 selector CHNL 37 line generation, PE 1-7 CPK 41

control instruction REC 17 indicator CNSL 8 IPL operation IFA 66 key CNSL 34 loading I-buffers CPU 53 local storage address formation MIC 7 data checking CPU 10 data flow INTR 7 destination addressing CPU 11 destination look ahead CPU 11 introduction INTR 11 maps CPU 16 operation CPU 10 scoping procedure CPU 17 timing chart CPU 11 local storage A A-addressing from console CPU 13 data flow CPU 12 indirect addressing CPU 13 LSCS mode (addressing) CPU 13 source direct address CPU 13 local storage address formation chart MIC 17 local storage assignments IFA 12 local storage B addressing from console CPU 15 addressing from console file CPU 15 data flow CPU 14 force count address CPU 15 indirect addressing CPU 15 source direct address CPU 15 local storage/expanded LS source gating diagram CPU 24 local storage/external address formation MIC 16 local-storage map CPU 16 location(s) character, F9XX translate CPK 30 CPU electrical components Models FED-I PWR 6 Models H2-K2 PWR 102 current PSW CPU 16 I-cycle hardware CPU 56 power frame Models FED-I PWR 6 Models H2-K2 PWR 102 storage STOR 66 log pres, indicator CNSL 6 logic and storage timings charts fast control access STOR 112 read control storage STOR 113 read main storage STOR 114 read operation STOR 116 store main storage STOR 115 store operation STOR 116 logic, basic symbology REF 4 logout areas REC 18 channel dependent REC 20 CPU dependent REC 19 CPU independent REC 18 IFA extended IFA 75 length REC 18 machine check REC 17

load

logout (continued) mask REC 17 multiplexer channel CHNL 36 pointer REC 17 selector channel CHAN 77 valid REC 18 low trap IFA IFA 74 low trap request IFA 40 lower roller, A-register display CNSL 14-15 LRTY REC 11 LRU inval indicator (adr x-late check) CNSL 5 LSCS mode (addressing) CPU 13 LSDST REC 11 lubrication (MG) Models FED-I PWR 60 Models H2-K2 (see 3047 Power Unit 11)

## Μ

M-register bit decode advanced bi-polar decode PWR 27 bit decode phase 2I decode PWR 24 comp, indicator CNSL 4 description CPU 102 gating (traps) CPU 45 introduction INTR 9 use STOR 3 machine check analysis and recording REC 24 code validity bits REC 18 control registers REC 17 description REC 8 extended interruption information REC 19 extended logout (MCEL) handler REC 23 handling REC 22 hard, description REC 18 hard, software recovery REC 23 interruption code REC 18 interruption codes (old PSW) REC 22 interruptions REC 16 length REC 18 logout REC 17 logout mask REC 17 mask REC 17 pointer REC 17 priority CPU 42 registers REC 8 soft, description REC 16 soft, software recovery REC 23 trap CPU 42 main and control storage areas STOR 19 main storage access MIC 5 description STOR 19 sizes INTR 8 maintenance aids blocking prefetch CNSL 19 channel REF 9 console-file byte counter CNSL 29 console printer-keyboard REF 10 control word address trap CNSL 19 CPU REF 8

maintenance aids (continued) data compare trap CNSL 18 I-cycle branch loop CPU 80 IFA REF 12 motor generator Models FED-1 PWR 62 Models H2-K2 (see 3047 Power Unit TM) major functional unit, card assignments REF 16 MAN (manual), indicator CNSL 8 manual storage procedure STOR 78 manual storing procedure STOR 79 maps control storage MIC 13 expanded local storage CPU 21 externals CPU 32, 33 local storage CPU 16 mark 0-in (interface line) description CHNL 4 introduction INTR 15 MAS assembler instructions MIC 10 mask asynchronous fixed logout REC 17 asynchronous machine-check logout REC 17 configuration report REC 17 damage report REC 17 external damage report REC 17 I/O logout REC 17 input/output extended logout REC 17 recovery report REC 17 synchronous machine-check extended logout REC 17 warning REC 17 masks and key validity REC 18 MB-registers description CPU 103 MB-registers purpose INTR 9 MCKA REC 8 MCKB REC 8 memory (see storage) meter, usage CNSL 35 metering-in CHNL 5 metering-out CHNL 5 MG (see motor generator) microcode-hardware functions (I-cycles) CPU 50 microdiagnostics basic tests listings DIAG 55 microprogram example BGA8 DIAG 19 sample listing DIAG 5 test BAB5 (console-file mode) DIAG 4 test BEA6 (local-store control store mode) DIAG 6 troubleshooting DIAG 4 extended tests I-cycle microprogram example (EDBO) DIAG 28 test EGE7 DIAG 8 test listings DIAG 56-60 testing philosophy DIAG 2 troubleshooting DIAG 8 micromonitor control DIAG 35 microprogram assembler instructions MIC 10

. 6

3145 TM X-9

3145 TM X-10

output MIC 7 branch symbols MIC 84 branching circuits CPU 104 control word MIC 15 control-word functions INTR 13 description MIC 12 index (370) REF 23 instruction retry REC 10 listing, fields defined MIC 8 listing (370 sample) MIC 6 microroutines (sample) MIC 3 operation CPK 57 patch procedure MIC 85 principles description MIC 2 introduction INTR 12 references material MIC 84 retry REC 13 retry intro INTR 14 temporary fix MIC 85 temporary fix procedure MIC 85 temporary fix routine MIC 85 370 MIC 3 mid-pac regulator removal and replacement PWR 32 mini-op control IFA 9 mini-ops IFA 9 codes IFA 21 control IFA 9 modifiers IFA 21 no-op IFA 36 read data IFA 37 trap IFA 74 write data IFA 38 write gap IFA 29 mixer board blower and filter PWR 59 mnemonics CPK 54 mode extended control CPU 131 quiet REC 5 recording REC 5 register description CPU 34 use REC 5 threshold REC 5 module selection IFA 33 storage STOR 10 switch, IFA IFA 33 monitor call CPU 130 MOP-register IFA 21 motor control logic CPK 45 motor downshift CPK 46 motor-generator (MG) adjustments (overvoltage and output) PWR 55 check indicator (PF) description PWR 29 location PWR 28 control (see motor-generator regulator) drive motor PWR 4 enclosure (housing) PWR 4 exciter (ac) PWR 4

microprogram. assembler (continued)

4 1

motor-generator (MG) (continued) faults PWR 70 general information PWR 3 hold, switch (CE3) description PWR 29 location PWR 28 lubrication PWR 60 output faults PWR 70 output voltage adjustment PWR 56 overvoltage adjustment PWR 55 overvoltage (OV) check PWR 71 power-off controlled, switch (CE3) description PWR 29 location PWR 28 preventive maintenance charts PWR 60 regulator general information PWR 3 overvoltage adjustment PWR-55 removal and replacement PWR 57 service aid charts PWR 62 thermal PWR 45 thermal check PWR 70 troubleshooting mide PWR 62 3047 (see 3047 Power Unit Theory-Maintenance Manual) motor-generator output voltage adjustment Models FED-I PWR 56 Models H2-K2 (see 3047 Power Unit Theory-Maintenance Manual) motor-generator regulator overvoltage adjustment Models FED-I PWR 55 Models H2-K2 (see 3047 Power Unit Theory-Maintenance Manual) motor-generator troubleshooting guide PWR 62 MPX (see byte-multiplexer channel) MRTY REC 11 MS (PSW masks and key validity) REC 18 MSF indicator stage 1 PWR 28 stage 2 PWR 30 MSKA external CPU 34 MSKB external CPU 34 MST regulators bias voltage PWR 15 bulk voltage PWR 14 component location PWR 77 diagnostic procedure PWR 66 dual-level PWR 20 sequencing first sequence PWR 17 second sequence PWR 17 single-level PWR 20 start line PWR 20 tri-level PWR 22 types PWR 20 multi-match indicator (adr x-late check) CNSL 5 multi-track operations, IFA read IFA 63 search IFA 69 multi-unit addressing CHNL 12 multiplex cable, IFA/2319 IFA 98 multiplexer channel INTR 16

multiplexer channel logout CHNL 36 multiplexer trap CPU 43 MULTITAG switch CNSL 37

# N

N-register description CPU 103 introduction INTR 9 new line control, 3210 CPK 23 next control word address formation MIC 16 no match, indicator (adr x-late check) CNSL 5 no-op command IFA 50 mini-op IFA 36 (no operation) CPK 60 NOREG (external word) CPU 34 NORM test flow, common routine common setup and entry to common routines DIAG 48 diagnostic "common routine" description DIAG 48 not ready switch CPK 6 NP-register (EXPLS-7B) CPU 22

# 0

OBR REC 25 off/keyboard, switch (3215) CPK 32 op-register CPU 60 op-register logic IFA 21 open flat-wire bus solution PWR 74 operating mode (CTCA) offline CNSL 36 online CNSL 36 operation MST regulator PWR 20 phase-controlled regulator PWR 18 operation registers IFA 21 operational-in CHNL 5 operational-out CHNL 5 operations basic IFA 35 priority CPU 41 operator's console, PR-KB description CPK 5 optimized character arrangement, 3210 print element CPK 8 optional features INTR 4 OS. checkpoint/restart REC 26 error-recovery procedures REC 25 1/O recovery management support REC 26 recovery management support REC 23 restart facilities REC 26 RMS REC 23 warm start REC 26 **OS/DOS** compatibility address adjustment example CPU 122 adjust CCW instruction CPU 123 example CPU 125 adjust CCW list CPU 124 execute local instruction example CPU 121 functional units CPU 117 general description CPU 116

OS/DOS compatibility (continued) instruction, execute local (exl) CPU 119 interrupt handling CPU 127 local list (lex list) CPU 120 LRU (least recently used) CPU 117 operational example CPU 118 microprogram flow chart (LEX, ACCW instructions) CPU 126 SIO example CPU 128 SVC interrupt example CPU 128 table buffer registers CPU 117 OS/DOS functional units CPU 117 outboard recorder REC 25 outlets (convenience) Models H2-K2 PWR 104 stage 1 PWR 8 stage 2 PWR 9 output voltage adjustment (MG) Models FED-I PWR 56 Models H2-K2 (see 3047 Power Unit Theory-Maintenance Manual) overall data flow INTR 6 overvoltage adjustment (MG) Models FED-I PWR 55 Models H2-K2 (see 3047 Power Unit Theory-Maintenance Manual) overvoltage/overcurrent detection Models FED-1 PWR 44 Models H2-K2 PWR 122 overvoltage (OV) trip circuit basic operation PWR 68 troubleshooting overvoltage protection disabled PWR 68 overvoltage protection operational PWR 68

# Ρ

P-register CPU 6 panel (see CE panel) parity control, data IFA 23 parity correction STOR 52 parity generator STOR 19 parity generator (conceptual) STOR 52 patch procedure, microprogram MIC 85 pause, indicator CNSL 7 PD (instruction processing damage) REC 18 PE timing counter CPK 41 PE 1-7 line generation CPK 41 PER CPU 173 PF indicator (PF) PWR 28 PGA, PGB, CHIP and control bits, selection STOR 32 phase rotation detection circuit Models FED-I PWR 34 Models H2-K2 PWR 114 phase-controlled regulator active cap PWR 18 active cap adjustment PWR 52 component location PWR 77 components PWR 18 control section operation PWR 18

phase-control regulator (continued) input PWR 16 operation PWR 18 regulator removal and replacement PWR 57 start line condition PWR 77 phase 21 address and control interface STOR 23 address and data interface (illustration) STOR 17 address flow, example STOR 24 BSM-ECCL-CPU interface (illustration) STOR 17 control lines board active STOR 23 board select STOR 23 machine reset STOR 23 reset STOR 23 write STOR 23 photographs (see troubleshooting aids) physical configuration general INTR 3 storage gates STOR 2 plug (jumper) cards block-multiplexer channel feature plug cards REF 18 memory select plug card REF 20 microprogram temporary fix plug card REF 20 selector channel feature plug card REF 20 serial number, EC and feature plug card REF 21 standard clock cards REF 20 PM (program mask and condition code validity) REC 18 PN-register CPU 22 pointer, machine-check extended logout REC 17 power CE panel Models H2-K2 PWR 110 stage 1 PWR 28 stage 2 PWR 30 check indicators CE panel (Models H2-K2) PWR 110 CE panel (stage 1) PWR 28 CE panel (stage 2) PWR 30 operator console CNSL 5 checking circuits Models FED-I PWR 40 Models H2-K2 PWR 118 component locations Models FED-I PWR 6 Models H2-K2 PWR 102 console file stage 1 PWR 46 stage 2 PWR 48 control switches Models H2-K2 PWR 112 stage 1 PWR 29 stage 2 PWR 31 conversion (ac-to-dc) CPU, Models H2-K2 stage 1 PWR 105 CPU, Models H2-K2 stage 2 PWR 106 CPU, stage 1 PWR 110 CPU, stage 2 PWR 111 PF. Models H2-K2 PWR 108 PF, stage 1 PWR 12 PF, stage 2 PWR 13

power (continued) CPU A-gate Models H2-K2 PWR 124 stage 1 PWR 46 stage 2 PWR 48 CPU B-gate Models H2-K2 PWR 124 stage 1 PWR 46 stage 2 PWR 48 dc distribution Models H2-K2, stage 1 PWR 105 Models H2-K2, stage 2 PWR 106 stage 1 PWR 8 stage 2 PWR 9 distribution (gate) CPU, stage 1 PWR 24 CPU, stage 2 PWR 25 PF A- and B-gate PWR 108 frame locations Models FED-I PWR 6 Models H2-K2 PWR 102 general information Models FED-I PWR 6 Models H2-K2 PWR 100 IFA stage 1 PWR 46 stage 2 PWR 48 motor generator Models FED-I PWR 4 Models H2-K2 (see 3047 Power Unit Theory-Maintenance Manual) off switch (CE4) description PWR 29 description (Models H2-K2) PWR 112 location PWR 28 location (Models H2-K2) PWR 110 off switch console CNSL 34 PR-KB CPK 6 protection circuits Models FED-I PWR 40 Models H2-K2 PWR 118 **PF** regulators Models H2-K2 PWR 107 stage 1 PWR 46 stage 2 PWR 48 preventive maintenance Models FED-I PWR 60 Models H2-K2 PWR 125 printer-keyboard stage 1 PWR 46 stage 2 PWR 48 safety Models FED-I PWR 3 Models H2-K2 PWR 100 service aids motor generator troubleshooting guide PWR 62 supplies (see regulators) thermals Models H2-K2 PWR 123 stage 1 PWR 45

power, thermals (continued) stage 2 PWR 46 visual index (ALD references) Models FED-I PWR 6 Models H2-K2 PWR 102 voltage adjustments stage 1 PWR 46 stage 2 PWR 48 power checks CE panel indicators Models H2-K2 PWR 110 stage 1 PWR 28 stage 2 PWR 30 protection and checking circuits Models FED-I PWR 40 Models H2-K2 PWR 118 power fault detection block diagram PWR 80 power frame ac-to-dc conversion Models H2-K2 PWR 107 stage 1 PWR 12 stage 2 PWR 13 power frame regulators Models H2-K2 PWR 107 stage 1 PWR 46 stage 2 PWR 48 power frame, storage STOR 2 power gate A (PGA) STOR 20 power gate B (PGB) STOR 20 power off CE panel (CE4 switch) Models H2-K2 PWR 112 stage | PWR 29 stage 2 PWR 31 console, key CNSL 34 emergency, key PWR 38 sequence CPU PWR 38 CTCA CNSL 36 power on CE panel (CE1 switch) description PWR 29 description (Models H2-K2) PWR 112 location PWR 28 location (Models H2-K2) PWR 110 complete, indicator Models H2-K2 PWR 110, 111 stage 1 PWR 28 stage 2 PWR 30 console, key CNSL 34 indicator (PR-KB) CPK 6 sequence (CPU) Models FED-1 PWR 35 Models H2-K2 PWR 115 sequence (CTCA) CNSL 36 start, indicator Models FED-I PWR 28 Models H2-K2 PWR 110 switch console CNSL 34 PR-KB CPK 6

power on (continued) switch (CE1) description PWR 29 description (Models H2-K2) PWR 112 location PWR 28 location (Models H2-K2) PWR 110 power-on sequencing Models FED-I PWR 34 Models H2-K2 PWR 114 power panel, CE Models FED-I PWR 28 Models H2-K2 PWR 112 power sequence and control circuitry Models FED-I PWR 36 Models H2-K2 PWR 116 power supplies IFA 97 preventive maintenance (PM) general information Models FED-I PWR 60 Models H2-K2 PWR 125 motor-generator Models FED-1 PWR 60 Models H2-K2 (see 3047 Power Unit Theory-Maintenance) motor-generator chart PWR 60 regulators Models FED-1 PWR 60 Models H2-K2 PWR 125 previous op algorithm IFA 12 primary power box, components Models FED-1 PWR 6 Models H2-K2 PWR 102 print element (type head) character layout CPK 8 print emitter sync check CPK 43 timing counter CPK 41 print magnet firing circuits, 3215 CPK 42 print operation, 3215 CPK 27 printed circuit board PWR 73 printer 3210 CPK 8 3215 CPK 27 printer tilt/rotate codes CPK 8 printing the punctuation character, colon (:) 3210 CPK 11 3215 CPK 31 priority controls diagram CPU 44 operation cycles CPU 40 operations CPU 41 operations (H-register) CPU 39 PR-KB code translation CPK 10 probe light IFA 86 proceed, indicator CPK 6 processing damage REC 18 program check (PR-KB) CPK 61 controlled interrupt (PCI PR-KB) CPK 61 mask validity REC 18 status word (PSW), defined CHNL 3 program event recording (PER) CPU 173 programming information (PR-KB) CPK 60 protected dual-level regulator (MST) PWR 21

PR-KB alter display information REF 8 protection check (PR-KB) CPK 61 PSW location CPU 16 PSWCTL, external word CPU 34 pushbutton switches (see switches)

# 0

quiet mode REC 5

### R

rate, rotary switch CNSL 24 RC (region code valid) REC 18 read fetch STOR 58 generator STOR 48 IFA commands IFA 63 data command IFA 65 data delay, VFO IFA 14 data mini-op IFA 37 diagnostic tests IFA 92 full cylinder test IFA 92 gate · IFA 27 HA command IFA 64 IPL command IFA 66 share request IFA 29 single track test IFA 29 read commands IFA 57 read. keyboard. operation 3210: CPK 25 3215 CPK 38 read, main storage operation STOR 58 read operation timing STOR 116 read write controls IFA 27 gate logic IFA 27 cycle control, 3210 CPK 22 readout control word MIC 4 readout main storage MIC 5 ready switch CPK 6 real address computation example CPU 122 recalibrate command IFA 54 record fields count IFA6 data IFA 7 home address IFA 6 identifier IFA 69 key IFA 6 mode REC 5 zero (RO) IFA 6 recording machine check REC 24 modes REC 5 recoverv hardware logic REC 3 management support (RMS) REC 23 procedures REC 25 report REC 8 software REC 2 description REC 23

recovery, software (continued) introduction REC 4 system REC 18 reference manuals REF 6 references, IFA latches REF 13 region code (RC) REC 19 region code valid REC 18 register Α description CPU 88 use STOR 3 ABRTY REC 11 B CPU 88 backup REC 11 description CPU 88 use STOR 3 CFDAR CPU 34 CPURTY REC 11 D CPU 95 DK (EXPLS-7C) CPU 22 EPSWA (external) CPU 34 EPSWB (external) CPU 34 FA assignment table IFA 12 initial selection IFA 43 seek command IFA 52 SIO-CC IFA 52 FAT assignment table IFA 11 use IFA 33 FB IFA 12 FBAK assignment table IFA 11 initial selection IFA 43 FBO assignment table IFA 11 use IFA 27 FC assignment table IFA 12 initial selection IFA 43 assignment table IFA 11 no-op IFA 36 read data IFA 37 write data IFA 38 write gap IFA 39 FCL assignment table IFA 11 no-op IFA 36 read data IFA 37 write data IFA 38 write gap IFA 39 FCS assignment table IFA 11 error trap sequence IFA 96 halt I/O or halt device IFA 85 initial selection IFA 43 FCND IFA 11 FD IFA 12 FDR

register, FDR (continued) assignment table IFA 11 read commands IFA 63 read data IFA 37 read operation IFA 29 search commands IFA 68 seek operation IFA 28 write commands IFA 57 write data IFA 38 write operation IFA 28 FDS assignment table IFA 11 diagnostic chaining test IFA 88 error trap sequence IFA 96 sense command IFA 48 FED assignment table IFA 11 diagnostic chaining test IFA 88 diagnostic unsafe condition test IFA 59 FERR IFA 11 FFL IFA 11 FGL IFA 11 FGT assignment table IFA 11 error trap sequence IFA 96 FHC IFA 11 FM assignment table IFA 12 cylinder address register test IFA 89 diagnostic unsafe condition test IFA 91 FMOD assignment table IFA 11 sense command IFA 48 use IFA 33 FOP assignment table IFA 11 cylinder address register test IFA 89 diagnostic chaining test IFA 88 diagnostic unsafe condition test IFA 91 no-op IFA 36 read data IFA 37 read/write diagnostic test IFA 92 write data _IFA 38 write gap IFA 39 FS IFA 12 FSB assignment table IFA 11 error trap sequence IFA 96 FST assignment table IFA 11 use IFA 33 FSTAT IFA 11 FTAG IFA 11 FTC CPU 96 FTI IFA 11 FTO IFA 11 FTS IFA 11 FW assignment table IFA 12 initial selection IFA 43 GBD CHNL 51

register (continued) GBF CHNL 51 GBUF CHNL 51 GB1 CHNL 51 GB2 CHNL 51 GB3 CHNL 51 GCL CHNL 88 GCT CHNL 51 GDRL description CPU 22 use CHNL 88 GE CHNL 51 GF CHNL 51 GL CHNL 51 GO CHNL 51 GR CHNL 51 GS CHNL 51 GSP CHNL 51 GTI CHNL 51 GTO CHNL 51 H (priority) CPU 38 HMRTY REC 11 I (EXPLS-50) description CPU 21 use CPU 62 IBU (EXPLS-54) description CPU 21 use CPU 61 ICS (EXPLS-56) CPU 21 IMM byte CPU 60 INTA (interrupt A) CPU 34 INTB (interrupt B) CPU 34 key CPU 61 L CPU 7 M description CPU 102 use STOR 3 MB CPU 103 MBI CHNL 15 MBO CHNL 16 mode REC 34 MOP no-op IFA 36 read data IFA 37 write data IFA 38 write gap IFA 39 MSKA (external) CPU 34 MSKB (external) CPU 34 MTI CHNL 17 MTO CHNL 17 N CPU 103 NP (EXPLS-7B) CPU 22 OP CPU 60 P. CPU 6 PN (EXPLS-79) CPU 22 S CPU 4 SAR STOR 3 save area REC 19 SDR STOR 3 SN (EXPLS-78) CPU 22 SPTL description INTR 10

3145 TM X-12

register, SPTL (continued) detail CPU 3 introduction INTR 7 use CPU 59 SPTLB REC 11 storage byte mark STOR 48 SYS (system register) CPU 38 T CPU6 TA 3210 CPK 16 3215 CPK 33 table buffer CPU 17 TF 3210 CPK 117 3215 CPK 33 TI, 3210 CPK 18 TODH CPU 109 TODL CPU 109 TR (EXPLS-55) description CPU 21 use CPU 62 TT. 3215 CPK 33 U (EXPLS-53) description CPU 21 use CPU 61 V (EXPLS-51) description CPU 21 use CPU 61 W(EXPLS-52) description CPU 21 use CPU 61 WK (EXPLS-7A) CPU 22 word 1 CPK 33 word 1 and word 2 CPK 33 Z CPU 94 regulator initial voltage adjustment procedure PWR 50 overvoltage adjustment (MG) Models FED-1 PWR 56 Models H2-K2 (see 3047 Power Unit Theory-Maintenance Manual) symptom-fix table PWR 82 regulator test switch (CE6) description Models FED-I PWR 29 Models H2-K2 PWR 112 location Models FED-1 PWR 28 Models H2-K2 PWR 110 regulators adjustments Models H2-K2 PWR 124 stage 1 PWR 46 stage 2 PWR 48 blowers and filters PWR 59 CE panel, indicators Models H2-K2 PWR 100 stage 1 PWR 28 stage 2 PWR 30 cooling Models FED-I PWR 3 Models H2-K2 PWR 100

regulators (continued) IFA adjustments (stage 1) PWR 47 adjustments (stage 2) PWR 49 locations IFA 97 indicators (seq 1 and 2) Models H2-K2 PWR 110 stage 1 PWR 28 stage 2 PWR 30 located in PF Models FED-1 PWR 47 Models H2-K2 PWR 103 maintenance approach PWR 60 mid-pac PWR 3 MST bias voltage PWR 15 bulk voltage PWR 14 component location PWR 77 diagnostic procedure PWR 66 dual-level PWR 20 information - PWR 3 sequencing PWR 17 single-level PWR 20 start line PWR 16 tri-level PWR 22 types PWR 20 overvoltage/overcurrent detection Models FED-I PWR 44 Models H2-K2 PWR 122 phase-controlled PWR 3 active cap PWR 18 component location PWR 77 components PWR 18 control section operation PWR 18 information PWR 3 input PWR 16 operation PWR 18 start line condition PWR 17 protection and checking circuits Models FED-1 PWR 40 Models H2-K2 PWR 118 removal and replacment. Models FED-1 PWR 57 Models H2-K2 PWR 126 seq 1 Models H2-K2 PWR 107 stage 1 PWR 28 stage 2 PWR 30 seq 2 Models H2-K2 PWR 107 stage 1 PWR 28 stage 2 PWR 30 series operation PWR 20 shunt operation PWR 20 service aid PWR 62 thermals (locations) Models FED-1 PWR 45 Models H2-K2 PWR 123 undervoltage sensing system Models H2-K2, stage 1 PWR 118 Models H2-K2, stage 2 PWR 120 stage 1 PWR 40 stage 2 PWR 42

voltage measurements Models H2-K2 PWR 124 stage 1 PWR 46 stage 2 PWR 48 remote sense leads (lines) **PWR 20** removal and replacement console file CFA 18 functional unit assignments **REF 16** power ac/dc modules PWR 58 ac/dc modules (Models H2-K2) PWR 126 blowers and filters (regulators) PWR 59 blowers and filters (Models H2-K2) PWR 128 dual-level supplies PWR 32 dual-level phase controlled PWR 57 motor generator (MG) PWR 58 MST series regulators PWR 57 request-in CHNL 5 request pending indicator CPK 6 request switch CPK 6 reset offline switch (3215) CPK 32 reset pulse adjustment PWR 52 reset reference bit (RRB) STOR 63 resistance table (K20, time delay) PWR 78 restart address mark IFA 17 key CNSL 33 procedures for PR-KB CPK 62 restore command IFA 50 restore seek test IFA 94 retry channel REC 11 command REC 11 counter REC 12 error REC 8 errors REC 15 hardware REC 11 indicator CNSL 4 instruction REC 11 introduction channel INTR 16 command INTR 16 error checking and correction (ECC) INTR 14 microprogram instruction INTR 14 microprogram REC 11 microroutine REC 12 operation REC 11 priority CPU 42 registers REC 11 timings REC 14 traps REC 15 return word example MIC 36 reverse operation, multiple characters CPK 53 ripple check PWR 66 RM (recovery report mask) REC 17 RMS REC 23 rotary switches A - H CNSL 28 RRB, reset reference bit STOR 63 RTYFLG REC 11 R/W coax-cable, IFA/2319 IFA 98

regulators (continued)

# S

S-register bit description CPU 4 safety Models FED-I PWR 3 Models H2-K2 PWR 100 SAR STOR 22 SAR and storage clock (second-level diagrams) advanced bi-polar storage STOR 31 phase 2I storage STOR 30 SAR bits advanced bi-polar decoding STOR 27 phase 21 decoding STOR 24 SAR parity check, indicator CNSL 4 save area REC 19 SC (storage error corrected) REC 18 SCAMPART STOR 78 SCR PWR 44 scan commands IFA 70 scan compare IFA 26 scan equal command IFA 73 schematic diagram (power sequencing) Models FED-1 PWR 36 Models H2-K2 PWR 116 SD (system damage) REC 18 SDBI STOR 3 SDBI parity check, indicator CNSL 4 SDBO array card waveshapes (phase 21 only) STOR 97 assembler INTR 9 data flow description CPU 9 overall INTR 6 use STOR 3 parity check indicator CNSL 4 parity check (phase 2I, ABM) STOR 96 pre asm CPU 8 SDR (statistical data recorder) REC 25 SDR, (see storage data register) SE (storage error uncorrected) REC 18 search, IFA commands IFA 69 ID command IFA 71 KD command IFA 73 key command IFA 72 share request IFA 28 second levels, IFA address-mark detection IFA 17 bit ring IFA 19 byte counter and decode IFA 22 C-register force IFA 30 clock IFA 16 compare circuits IFA 26 cyclic check IFA 24 data request IFA 28 data transfer clock IFA 28 index IFA 32 module select IFA 33 op register and decode IFA 21 read/write gates IFA 27 SERDES and write data IFA 23 share cycle IFA 30 share cycle clock IFA 31 share request gates IFA 29

second levels. (IFA) (continued) variable frequency osc IFA 14 write data IFA 20 zero detection IFA 16 sector ready CFA 10 seek address IFA 50 seek command IFA 52 seek diagnostic tests IFA 94 segment CPU 139 select block address, line STOR 35 selection controls CHNL 5 selector channel block-multiplexer channel IPL CHNL 81 error logout CHNL 78 trap CPU 43 buffer forward/backward assembler CHNL 85 buffer share request generation CHNL 86 dependent log REC 21 diagnostic controls DIAG 31 'GA' diagnostic functions DIAG 31 general information INTR 16 indicator CNSL 4 log REC 21 logout CHNL 77 word buffer CHNL 83 selector channels address miscompare latch CHNL 55 address-out latch CHNL 55 addressing principles CHNL 49 basic data flow CHNL 46 branch word GA function CHNL 48 buffer byte counter CHNL 85 buffer shift controls CHNL 84 catalog numbers CHNL 78 channel ID CHNL 77 channel-loaded latch CHNL 54 channel logout CHNL 77 channel-primed latch CHNL 55 check facilities CHNL 71 command-out latch CHNL 55, 56 control storage map CHNL 47 count-ready latch CHNL 55 device-initiated polling controls CHNL 56 diagnostic controls DIAG 31 error routine CHNL 79 exceptional status trap CHNL 73 external word address and bit assignments CHNL 51 functional units CHNL 48 GA diagnostic function DIAG 31 GCL register CHNL 88 GDRL register CHNL 88 halt device (HDV) CHNL 68 halt I O (HIO) CHNL 68 hold-go-for-compare latch CHNL 55 1'O communications area CHNL 77 initial selection polling controls CHNL 54 initialization CHNL 62 interruption handling CHNL 75 local-storage assignments CHNL 50 locations CHNL 47 logout CHNL 77

selector channels (continued) microprogram example (EJD5) DIAG 33 memory flag register CHNL 88 operational overview CHNL 61 operations CHNL 60 poll-control latch CHNL 54 polling clock CHNL 54 recirculate latch CHNL 55 retry-holdup latch CHNL 55 select-out latch description CHNL 55 timing CHNL 61 service-out latch CHNL 55 share cycle clock CHNL 59 controls CHNL 57 input operation with buffer CHNL 87 share request priority controls CHNL 58 standard interface CHNL 4 start I/O (SIO) CHNL 62 status handling CHNL 73 status latch CHNL 55 status store CHNL 76 test I/O CHNL 65 typical buffer position CHNL 84 word buffer CHNL 83 sense bit error cond IFA 47 sense byte for PR-KB CPK 62 sense bytes (see IFA sense bytes) sense command IFA 46 sense indications IFA 9 sense operation IFA 48 sense point caps PWR 25 separated clock/data IFA 14 sequencing (power) circuitry schematic Models FED-I PWR 36 Models H2-K2 PWR 116 description Models FED-1 PWR 34 Models H2-K2 PWR 114 EPO PWR 38 operation Models FED-1 PWR 34 Models H2-K2 PWR 114 power check PWR 38 power off PWR 38 power on PWR 35 sequential seek test IFA 94 SERDES logic IFA 23 SERDES output timing IFA 23 serial/deserial register (SERDES) IFA 23 series regulator operation PWR 20 service aid blocking prefetch CNSL 19 CE panel (switches and indicators) Models H2-K2 PWR 110 stage 1 PWR 28 stage 2 PWR 30 console-file byte counter CNSL 29 control word address trap CNSL 19 data compare traps CNSL 19

service aid (continued) diagnostics STOR 48 hardware checks STOR 71 I-cycle branch loop CPU 80 local storage addressing scope procedure CPU 17 locations (CPU) STOR 66 MG-fault isolation charts Models FED-I PWR 62 Models H2-K2 (see 3047 Theory-Maintenance Manual) service checks Models H2-K2 PWR 124 stage 1 PWR 46 stage 2 PWR 48 time delay card jumpering example STOR 73 timing adjustments and checks STOR 73-75 service aids, storage (see servicing information) service-in CHNL 4 service-out CHNL 4 service tips PWR 76 set clock (SCK) CPU 111 set file-mask command IFA 51 set IC key CNSL 32 set/reset controls IFA 13 setting the word registers CPK 43 share cycle, IFA clock logic IFA 31 gate logic IFA 30 indicator CNSL 6 priority CPU 42 timings IFA 31 share request controls IFA 28 share request gate logic IFA 29 shift cycle operation, 3210 CPK 26 shift gating, ALU CPU 92 shift latch CPK 45 shunt regulator bias and ripple check PWR 54 operation PWR 20 silicon-controlled rectifier (SCR) PWR 44 simplex cables, IFA/2319 IFA 98 single-level regulator (MST) inputs PWR 20 operation PWR 20 single-level supply removal and replacement Models FED-I PWR 57 Models H2-K2 PWR 127 singleshot chopped forward or reverse CPK 45 SS2 adjustment CPK 46 single unit addressing CHNL 12 six-pac installation REF 22 removal REF 22 SM (synchronous machine-check extended logout mask) REC 17 SN-register (expls-78) CPU 22 SNG ECC CHK indicator CNSL 6 thld, indicator CNSL 6 soft machine checks description REC 16 recovery REC 23

3145 TM X-14

software recovery description REC 23 introduction REC 4 source gating (expls) CPU 23 space count command IFA 55 special voltage distribution chart REF 3 SPTL backup (SPTLB) REC 11 register description INTR 10 detail CPU 3 introduction INTR 7 use CPU 59 SR (system recovery) REC 18 SRTY REC 11 SSK, set storage key STOR 63 ST (storage logic validity) REC 18 standard features INTR 4 standard index IFA 32 standard interface description CHNL 4 introduction INTR 15 start console-file key CNSL 32 start I/O operation IFA IFA 42 multiplexer CHNL 22 selector CHNL 62 start, key CNSL 33 start line check PWR 67 start lines definition PWR 16 equivalent circuit PWR 16 first sequence PWR 17 first sequence diagram PWR 16 purpose PWR 16 second sequence PWR 17 troubleshooting _ PWR 67 stat bits S-register CPU 4 stat set symbols MIC 84 statistical data recorder REC 25 status byte IFA 74 status byte for PR-KB CPK 61 status CE indications IFA 87 status-in CHNL 4 status indicators IFA 87 status modifier (status bit PR-KB) CPK 61 stepper motor control CPK 44 stepping switch operation PWR 32 sequencing PWR 32 stop after log multiplexer CHNL 36 recovery procedures REC 22 selector CHNL 77 stop key CNSL 33 stop latch CPK 46 stop latch buffered CPK 46 stop latch set conditions CPK 47 stopping (3215) CPK 46 storage A and B gates STOR 4-6 ABM address buffer card (Part No. 8231509 or 8238201) STOR 103

storage (continued) ABM address buffer card to storage board interface STOR 104 ABM address buffer (03A-B3, B4, C3, C4) location A2, component side STOR 102 ABM error hold circuit STOR 118 ABM, 03A-B3, B4, C3, C4 board wiring address and controls STOR 105 ABM, 03A-B3, B4, C3, C4 board data wiring (vertical) STOR 106 accessing STOR 3 address assignment (ABM) STOR 100 address check STOR 47 address check circuit STOR 117 address interface STOR 28-29 addressing advanced bi-polar STOR 14 phase 21 STOR 13 adjustments and checks STOR 73-75 allocations, compatibility features FEAT 2 any error retain latch STOR 118 array card breakdown STOR 10 array, module and chip selection by storage size STOR 32 simplified logic example STOR 15 bit-cell STOR 16 board (ABM) STOR 101, 110 board capacitor layout chart STOR 118 board selection STOR 22 BOM (see BSM) BSM address and control interface connections STOR 28-29 BSM address range STOR 4-6 BSM interface timing diagram STOR 36-37 BSM internal timing diagrams STOR 38-39 BSM reset line checks STOR 117 BSM-to-ECCL interface STOR 17-18 BSMs STOR 7 byte check STOR 72 byte mark register STOR 48 byte marks parity checks, indicator CNSL 4 cabling, data and check bits STOR 40-42 capacities (phase 21 and advanced bi-polar) STOR 2 clock STOR 22 clock logic STOR 30-31 configurations STOR 4-6 and 6.1 controls STOR 13 control line generation (PGA, PGB, chip, add 12, CTRL) STOR 32 control word (see control word) CPU select pulse STOR 33 CPU-to-ECCL interface lines STOR 17-18 ctrl lines parity check, indicator CNSL 4 data bit location chart STOR 41 data-bus-out-pre-assembler CPU 8 data check STOR 72 data flow fetch STOR 48 ECCL STOR 21 store STOR 48 data interface STOR 28-29 data register (SDR) description STOR 46

storage, data register (continued) use STOR 3 diagnostics STOR 70 ECC STOR 48 ECC-board-to-CPU timing diagram STOR 33 ECC decoder chart STOR 50 ECC logic diagram STOR 49 ECCL board (ABM) STOR 109 ECCL board layout STOR 56-57 ECCL-to-BSM interface lines STOR 17-18 ECCL troubleshooting circuit aids (ABM) STOR 118 error corrected REC 18 error detection and correction STOR 52 error handling STOR 20 error type REC 18 error-type decode logic diagram STOR 53 external, addition of STOR 107 fetch operation data flow STOR 46 functional operations STOR 58 functional units STOR 19-20 gate layout (by addresses) STOR 4-6 and 6.1 general information STOR 2 hardware check logic diagram STOR 71 interface address and control STOR 28-29 interface BSM-to-CPU STOR 17-18 introduction concepts INTR 8 introduction, hardware description STOR 2 jumpering and trilead swapping requirements (01BA3) for upgrading storage above 256k STOR 107.1 logging (error) STOR 20 main/control selection STOR 13 miscellaneous maintenance information STOR 117 module and chip selection STOR 15 MST board layout STOR 11-12 operations read control operation STOR 58 PGA, PGB, chip and control bits selection STOR 32 protect description INTR 9 feature hardware description STOR 62 functional units STOR 63 insert storage key (ISK) STOR 63 introduction INTR 9 indicator (parity check) CNSL 4 logic diagram STOR 64-65 overview STOR 20 set storage key (SSK) STOR 63 protection STOR 20 read control STOR 113 read half word example MIC 52 read main storage STOR 58 SAR (storage address register) STOR 22 SAR bit transposition checks STOR 117 SAR line checking STOR 117 SDR description STOR 44-46 SDR use STOR 3 select, rotary switch CNSL 22 sizes STOR 2 store operation STOR 60 syndrome decoder chart STOR 52 tie-up or tie-down application STOR 117 timing STOR 33 timing and adjustments STOR 73-75

storage (continued) timing chart, CPU select pulse STOR 33 timing checks of storage related timing cards (CPU, PF) STOR 117 trilead checks STOR 117 upgrading STOR 107.1 virtual CPU 139 word example (see control word) word forms (see control word) write/store operation STOR 60 24K BSM configuration STOR 8 48K BSM configuration STOR 8 storage address register (SAR) STOR 19 storage boards (see BSM) storage checking procedures manual storage STOR 78 word from switches STOR 86 storage configurations STOR 4-6 and 6.1 storage console approaches, manual procedures and reference timings (SCAMPART) STOR 78 storage data bus-in STOR 3 storage data bus-out STOR 3 storage data register advanced bi-polar storage description STOR 46 second-level diagram STOR 47 data flow, read STOR 46 data flow, write STOR 46 general description STOR 19 phase 21 storage, description STOR 44 storage interface advanced bi-polar STOR 12 phase 21 STOR 11 storage timing and adjustments STOR 73-75 storage protect condition codes STOR 63 description STOR 20 functional description STOR 63 hardware location STOR 63 high level diagrams storage capacities, 112-512k bytes STOR 64 storage capacities, 768 or 1024k bytes STUR 65 instruction format (ISK, SSK, RRB) STOR 63 key, description STOR 63 overview STOR 62 servicing information STOR 69 stock, description STOR 63 storage instructions insert storage key (ISK) STOR 63 reset reference bit (RRB) STOR 63 set storage key (SSK) STOR 63 storage selection STOR 13 storage servicing information card swapping CPU STOR 66 power frame STOR 67 diagnostics basic gloup (*BAS) STOR 70 extended group (*EXT) STOR 70 introduction STOR 70 MBA0 run-time chart STOR 70 hardware checks

storage servicing information, hardware checks (continued) address check STOR 72 byte check STOR 72 data check STOR 72 hardware check STOR 71 second-level diagrams STOR 71-72 storage-control-line parity check STOR 72 locations CPU STOR 66 power frame STOR 67 storage protect (03AA4) ALD page references STOR 69 card function chart STOR 69 card socket diagram STOR 69 timing chart STOR 69 supplementary packaging information STOR 100 storage timings (see logic and storage timing charts, and timing chart, storage) storage to ECCL (phase 21) STOR 17 store channel ID instruction REC 19 clock (STCK) CPU 111 control instruction REC 17 CPUID instruction REC 19 key CNSL 32 status CPK 54 store operation STOR 20 store operation timings STOR 116 store (write) STOR 60 store 1 cycle indicator CNSL 6 strobe diagnostic CPK 17 keyboard, 3210 description CPK 7 keyboard, 3210 use CPK 15 subchannels CHNL 3 subclass REC 18 supplies (see regulators) supply response to overvoltage/overcurrent conditions PWR 23 suppress-out CHNL 4 SW external word CPU 34 switch address compare CNSL 18 address compare control CNSL 31 alarm reset CPK 6 allow CE mode IFA 86 alter/display CPK 6 blower off (CE3) Models FED-I PWR 28 Models H2-K2 PWR 110 check control CNSL 26 check reset CNSL 32 CK reset (CE2) Models FED-I PWR 29 Models H2-K2 PWR 112 console-file register display CNSL 31 control address set CNSL 32 cylinder select/head select (IFA) IFA 86 diagnostic/console-file control CNSL 28 display CNSL 32 emergency pull CNSL 35 enable system clear CNSL 32 end CPK 6 error disable IFA 86

### 3145 TM X-16

switch (continued) error override (CE2) Models FED-I PWR 29 Models H2-K2 PWR 112 I/O hold (CE5) Models FED-I PWR 29 Models H2-K2 PWR 112 I/O interface CNSL 36 I/O off Models FED-I PWR 29 Models H2-K2 PWR 112 interrupt CNSL 34 interval timer CNSL 31 lamp test CE6 PWR 29 CE6 (Models H2-K2) PWR 112 console CNSL 31 IFA IFA 86 load CNSL 34 MG hold (CE3) Models FED-I' PWR 29 Models H2-K2 PWR 112 MG pwr off controlled (CE3) Models FED-1 PWR 29 Models H2-K2 PWR 112 not ready CPK 6 off/keyboard (3215) CPK 32 power offconsole CNSL 34 CE4 PWR 29 CE4 (Models H2-K2) PWR 112 PR-KB CPK 6 power on console CNSL 34 CE1 PWR 29 CE1 (Models H2-K2) PWR 112 PR-KB CPK 6 rate CNSL-24 ready CPK 6 reg test (CE6) Models FED-1 PWR 29 Models H2-K2 PWR 112 request CPK 6 reset offline (3215) CPK 32 restart CNSL 33 rotary, A-H CNSL 30 set IC CNSL 32 start CNSL 33 start console file CNSL 32 stepping PWR 32 stop CNSL 33 storage select CNSL 22 store CNSL 32 system reset CNSL 33 test on/test off (3215) CPK 32 test select IFA 86 TOD clk CNSL 31 symbolic microprogram MIC 6 sync byte IFA 17 sync jack IFA 86 syndrome decoder chart STOR 52 syndrome latches, variable set to (EC140239 only) STOR 92 sys indicator CNSL 8 'system indicator CNSL 8 check, indicators CNSL 8 control panel CNSL 2 damage REC 16 indicators CNSL 8 recovery REC 16 reset, key CNSL 33 System/360, System/370 compatibility exceptions INTR 14 symptom-fix information PWR 82 table (regulators) PWR 82

### Т

T-mode (keyboard test) CPK 54 T-register CPU 6 **TA-register** 3210 CPK 16 3215 CPK 33 table EBCDIC for 3210 graphics CPK 10 EBCDIC to tilt/rotate locations CPK 11 F9XX CPK 30 F9XX character locations CPK 30 keyboard-tilt/rotate codes CPK 8 keyboard to EBCDIC storage locations CPK 11 3215 7 x 7 matrix codes CPK 28 tags-in CHNL 4 tags-out CHNL 4 TD (timer damage) REC 18 TE-register 3210 CPK 17 3215 CPK 33 temperature limits (thermals) Models FED-1 PWR 45 Models H2-K2 PWR 123 temperature sensing Models FED-I PWR 45 Models H2-K2 PWR 123 temporary fix, microprogram MIC 85 test I/O operation IFA IFA 82 multiplexer CHNL 22 selector CHNL 65 test, indicator CNSL 8 test select switch IFA 86 testing basic tests DIAG 2 data flow DIAG 3 extended tests DIAG 2 philosophy DIAG 2 thermal checks PWR 69 indicator CNSL 5 operation PWR 69 sensing and locations Models FED-I PWR 45 Models H2-K2 PWR 123

therm check indicator Models FED-I PWR 28 Models H2-K2 PWR 110 threshold mode REC 5 **TI-register** input gating, 3215 CPK 35 3210 CPK 18 tilt/rotate codes CPK 8 time-of-day clock (see TOD clock) time-of-interruption occurrence REC 18 timeout circuit CPK 20 timing charts, IFA address-mark detection IFA 17 bit ring IFA 19 clock IFA 18 index IFA 32 read share IFA 28 search share IFA 28 share cycle IFA 31 variable frequency osc IFA 14 write share IFA 28 zeros clocking IFA 16 counter, PE pulse CPK 41 CPU clock (see CPU clock) I-cycle CPU 67 I-cycle branch loop CPU 28 retry REC 14 timing, chart (storage) advanced bi-polar, main storage STOR 33 phase 21 control storage STOR 33 main storage STOR 33 timing (ECCL and BSM) BSM interface STOR 36-37 BSM internal STOR 38-39 charts STOR 33 CPU to ECCL STOR 33 ECCL delay lines STOR 34-35 TOD clock ALD references CPU 113 bit definition CPU 109 card locations CPU 113 clock damage REC 16 clock-setting sequence CPU 110 clock-update sequence CPU 110 enable switch CPU 109 error detection CPU 110 H-register CPU 109 instructions CPU 110 inval-indicator CPU 109 L-register CPU 109 manual set CPU 110 operation CPU 109 output assembler CPU 112 physical description CPU 109 security switch CPU 109 set clock instruction CPU 111 store clock instruction CPU 111 switch CNSL 31 validity indicator CPU 109

TR-register (EXPLS-55) description CPU 21 use CPU 62 TR special dc power requirements stage | PWR 46 stage 2 PWR 48 TRs CPU stage 1 PWR 10 stage 1 (Models H2-K2) PWR 105 stage 2 PWR 11 stage 2 (Models H2-K2) PWR 106 power frame Models H2-K2 PWR 107 stage 1 PWR 12 stage 2 PWR 13 track format IFA 6 transfer in channel IFA 8 transfer in channel (TIC) CPK 60 transformers CPU Models H2-K2, stage 1 PWR 105 Models H2-K2, stage 2 PWR 106 stage 1 PWR 8 stage 2 PWR 9 stage 1 (400-Hz) PWR 10 stage 2 (400-Hz) PWR 11 **PF PWR 107** translating from EBCDIC to matrix code CPK 29 translation 3210 PR-KB code CPK 10 3215 PR-KB code CPK 28 trap address descriptions CPU 41 address list CPU 41 addresses (retry) REC 15 controls diagram CPU 44 CPU high CPU 43 CPU low CPU 43 cycles CPU 40 IFA high CPU 43 IFA low CPU 43 machine check CPU 42 operation CPU 39 retry CPU 42 routines IFA 74 scan/clear CPU 43 store display CPU 43 1-cycle, indicator CNSL 6 2-cycle, indicator CNSL 6 traps M-register gating diagram CPU 45 retry REC 15 TRTY REC 11 tri-level regulator (MST), stage 1 components PWR 22 inputs PWR 22 operation PWR 22 tri-level regulator, stage 2 components PWR 23 sequencing PWR 23

bias voltage check PWR 67 blowers PWR 64 bulk voltage check PWR 66 CPU thermals PWR 69 dc profile PWR 65 dc voltage profile PWR 65 dc ripple check PWR 66 filters PWR 64 ground loop check PWR 74 grounding principles PWR 72 Models H2-K2 (see troubleshooting PWR) MG checks PWR 70 faults PWR 70 overvoltage check PWR 71 MST regulators PWR 66 open flat-wire bus PWR 74 overvoltage trip PWR 68 photographs (location) AC/DC module PWR 77 bias transformer PWR 77 bulk transformer PWR 77 MST regulators PWR 76 phase-controlled dual regulator PWR 77 transformer (TR108/TR408) PWR 77 service tips PWR 76 start line check PWR 67 TT-register 3210 CPK 20 3215 CPK 33 two volt phase-controlled regulator adjustment and scoping procedure PWR 50 two-volt phase-controlled regulator scoping procedure PWR 50 two-volt regulator adjustment PWR 50 two-volt regulator scoping procedure PWR 50 T₀ STOR 33

# U

troubleshooting aids

U-laminar bus A-gate PWR 24 B-gate PWR 24 **PF PWR 108** U-register (EXPLS 53) description CPU 21 use CPU 66 undervoltage sensing stage 1 PWR 40 stage 2 PWR 42 Models H2-K2 PWR 118 unut check (status bit 6 PR-KB) CPK 62 unit control words (UCWs) CHNL 4 unit data flow A-local storage CPU 12 A-register display CNSL 16 address compare switch CNSL 18 ALU CPU 93 B-local storage CPU 14 display assembler out CNSL 10 I-cycle address generation CPU 63

unit data flow (continued) I-cycle address generation and control decode CPU 65 I-cycles CPU 58 local storage CPU 12 SDBQ pre-asm CPU 9 storage functional units STOR 22 unit exception (status bit 7 PR-KB) CPK 62 unit status byte IFA 74 unsafe indications IFA 87 upper roller, display assembler out CNSL 10 usage meters CNSL 35

### V

V-register (EXPLS-51) data flow CPU 61 storage location CPU 21 validate function DIAG 19 validity bits, machine-check code REC 18 variable frequency oscillator (see VFO) VFO adjustments IFA 14 logic IFA 14 timing IFA 14 trigger IFA 14 zeros clocking IFA 16 virtual storage CPU 139 visual index (power) ALD reference pages Models FED-1 PWR 6 Models H2-K2 PWR 102 voltage console file stage 1 PWR 47 stage 2 PWR 49 convenience outlet Models FED-I PWR 8 Models H2-K2 PWR 104 CPU A and B gates stage 1 PWR 46 stage 2 PWR 48 CPU B-gate main storage stage 1 PWR 46 stage 2 PWR 48 distribution flat-wire bus PWR 26 gate (dc) PWR 24 general information PWR 8 Models H2-K2 PWR 104 U-laminar bus PWR 24 flat-wire bus chart PWR 48 IFA stage 1 PWR 47 stage 2 PWR 49 laminar bus chart PWR 48 measurements and adjustments Models H2-K2 PWR 124 stage 1 PWR 46 stage 2 PWR 48 PF regulators stage 1 PWR 47

voltage, PF regulators (continued) stage 2 PWR 49 printer-keyboard stage 1 PWR 47 stage 2 PWR 49 TR special dc power stage 2 PWR 49 voltage distribution CPU Models FED-1 PWR 8 Models FED-1 PWR 8

Models H2-K2 PWR 104

# W

w (warning) REC 18 W-register (EXPLS-52) description CPU 21 use CPU 61 wait indicator CNSL 8 warm start facility for OS REC 26 wire color codes (grounding) PWR 73 wire fire data to T1 DIAG 37 WK-register (EXPLS-7A) CPU 22 WM (warning mask) REC 17 word buffer, selector CHNL 83 move word bit format MIC 40 description MIC 38 example MIC 42 registers, setting the CPK 43 selection (SDBO) CPU 8 1 and 2 register (3215) CPK 33 word top, word bottom (storage) STOR 16 WP (PSW MWP validity) REC 18 write ACR CPK 60 data (word 1 and word 2) registers CPK 31 IFA CKD command IFA 60 clock/data generation IFA 20 clock gate IFA 27 commands IFA 57 data command IFA 59 data mini-op IFA 38 diagnostic tests IFA 92 full cylinder test IFA 92 gap mini-op IFA 39 gate IFA 27 gate logic IFA 27 HA command IFA 56 oscillator IFA 15 share request IFA 28 single-track test IFA 92 special command IFA 57

write (continued) operation 3210 CPK 23 3215 CPK 36 store operation, storage STOR 60 WTC autotransformer Models FED-I PWR 8 Models H2-K2 PWR 104

### Ζ

Z-register introduction INTR 10 description CPU 14 zeros clocking chart IFA 16 clocking IFA 16 detect logic IFA 14 detection, IFA VFO IFA 14 NUMERIC INDEX 23 FD (see console file) 370 microprogram index REF 23 microporgram list MIC 16 microporgram listing destinations MIC 8 samples MIC 6 1400/1440 compatibility feature instruction EA FEAT 10 2312 disk drive IFA 4 2313 disk drive IFA 4 2316 disk drive IFA 4 2318 disk drive IFA 4 2319 disk facility IFA 4 3046 power interlock description PWR 34 3145 addressing configuration, example STOR 80 3145 reference manual REF 6 3210 console printer-keyboards, description CPK 4 control and data flow CPK 15 data and control registers CPK 15 keyboard CPK 7 keyboard codes CPK 7 keyboard (read) operation CPK 25 power distribution PWR 8 PR-KB code translation CPK 10 PR-KB integrated attachment CPK 14 printer CPK 8 shift-cycle operation CPK 26 voltage chart stage | PWR 47 stage 2 PWR 49 write operation CPK 24 3215 attachment data flow DIAG 36 condensed data flow CPK 33 console printer-keyboard CPK 27 - console printer-keyboard, description CPK 5 control and data flow CPK 34 data and control registers CPK 33 diagnostic functions DIAG 36

3215 (continued) diagnostic hardware DIAG 36 keyboard CPK 27 keyboard operation (read) CPK 38 power distribution PWR 8 PR-KB code translation CPK 28 PR-KB integrated attachment CPK 33 print magnet firing circuits CPK 42 print operation CPK 27 printer CPK 27 TE-register CPK 33 tests-description DIAG 39-42 TT-register CPK 33 voltage chart, stage 1 PWR 47 voltage chart, stage 1 PWR 47 write operation CPK 36 7 x 7 matrix codes, cable CPK 28 3345 addressing configuation, example STOR 80 3345 storage and control frame PWR 6