IBM **Maintenance Library**

3125 Processing Unit
Input/Output Processor

SY33-1063-1

Second Edition (November, 1973)

This manual is a major revision of, and makes obsolete SY33-1063-0. The information in the manual has been completely revised, and the manual must be read in its entirety. Changes are continually made to the information; any such changes will be reported in subsequent revisions or Technical Newsletters.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

Forms for readers' comments are provided at the back of the manual. If the forms have been removed, comments may be addressed to IBM Laboratories, Product Publications, Dept. 3179, 703 Boeblingen/Wuertt, P.O. Box 210, Germany. Comments become the property of IBM.

Preface

This manual describes the theory of operation of the input/output processor (IOP) and provides maintenance information for the IOP. Readers of the manual should have a basic understanding of IBM system concepts. The manual supplements the System/370 Model 125 CE course and serves also as a recall aid; it is not intended for self-education.

The manual is divided into seven chapters.

Chapter 1 contains general information, and an overall data flow of the IBM System/370 Model 125.

Chapter 2 contains operational principles, and IOP-MSC communication.

Chapter 3 contains operational details, and IOP-SVP communication.

Chapter 4 contains a description of the functional units.

Chapter 5 contains error conditions and their handling.

Chapter 6 contains maintenance information.

Chapter 7 contains reference information.

An appendix contains particular IOP '8' microprogram information.

Common abbreviations for the system and an explanation of the symbols used are given in *IBM 3125 Processing Unit, General System Information,* Maintenance Library Manual, Order No. SY33-1059.

IBM is grateful to the American National Standards Institute (ANSI) for permission to reprint its definitions from the American National Standard Vocabulary for Information Processing (Copyright © 1970 by American National Standards Institute, Incorporated), which was prepared by Subcommittee X3.5 on Terminology and Glossary of American National Standard Committee X3.

Prerequisite Reading

Maintenance Library Manuals

IBM 3125 Processing Unit, Microinstructions, Order No. SY33-1058.
IBM 3125 Processing Unit, General System Information, Order No. SY33-1059.
IBM 3125 Processing Unit, Main Storage Controller, Order No. SY33-1061.
IBM 3125 Processing Unit, Instruction Processing Unit, Order No. SY33-1062.
IBM 3125 Processing Unit, Service Processor Subsystem, Order No. SY33-1065.
Section 1: Service Processor (SVP).
Section 2: Console Disk File.
Section 3: Display Unit and Keyboard.

Associated Publications

System Library Manuals

IBM System/360 Principles of Operation, Order No. GA22-6821.
IBM System/370 Principles of Operation, Order No. GA22-7000.

Maintenance Library Manuals

IBM 3125 Processing Unit, Power Supplies, Order No. SY33-1060.
IBM 3125 Processing Unit, Magnetic Tape Adapter, Order No. SY33-1064.
IBM 3125 Processing Unit, Main Storage, Order No. SY33-1066.
IBM 3125 Processing Unit, Multiplexer Channel, Order No. SY33-1067.
IBM 3125 Processing Unit, 2560 Attachment, Front End, Order No. SY33-1068.
IBM 3125 Processing Unit, 3525 Attachment, Front End, Order No. SY33-1070.
IBM 3125 Processing Unit, 3504 Attachment, Front End, Order No. SY33-1071.
IBM 3125 Processing Unit, 1403 Attachment, Front End, Order No. SY33-1072.
IBM 3125 Processing Unit, 3330 Direct Disk Attachment, Order No. SY33-1073.
IBM 3125 Processing Unit, Integrated Console Printer Attachment, Order No. SY33-1074.
IBM 3125 Processing Unit, Integrated Communications Adapter, Part B/M 1876075.
IBM 3125 Processing Unit, Installation Instructions, Part 4014001.
IBM 3125 Central Test Manual, contains pages appropriate to the individual 3125 Processing Unit.
IBM 3125 Processing Unit, Parts Catalog, Order No. S135-1000.

Contents

Safety

Personal Safety

Personal safety cannot be over-emphasized; it is a vital part of customer engineering. To ensure your safety and that of co-workers, always observe the safety precautions given during your safety training and adhere to the following:

General Safety Practices

Observe the general safety practices and the procedure for performing artificial respiration that are outlined in *CE Safety Practices* card, order no. S229-1264 (shown here).

Grounding

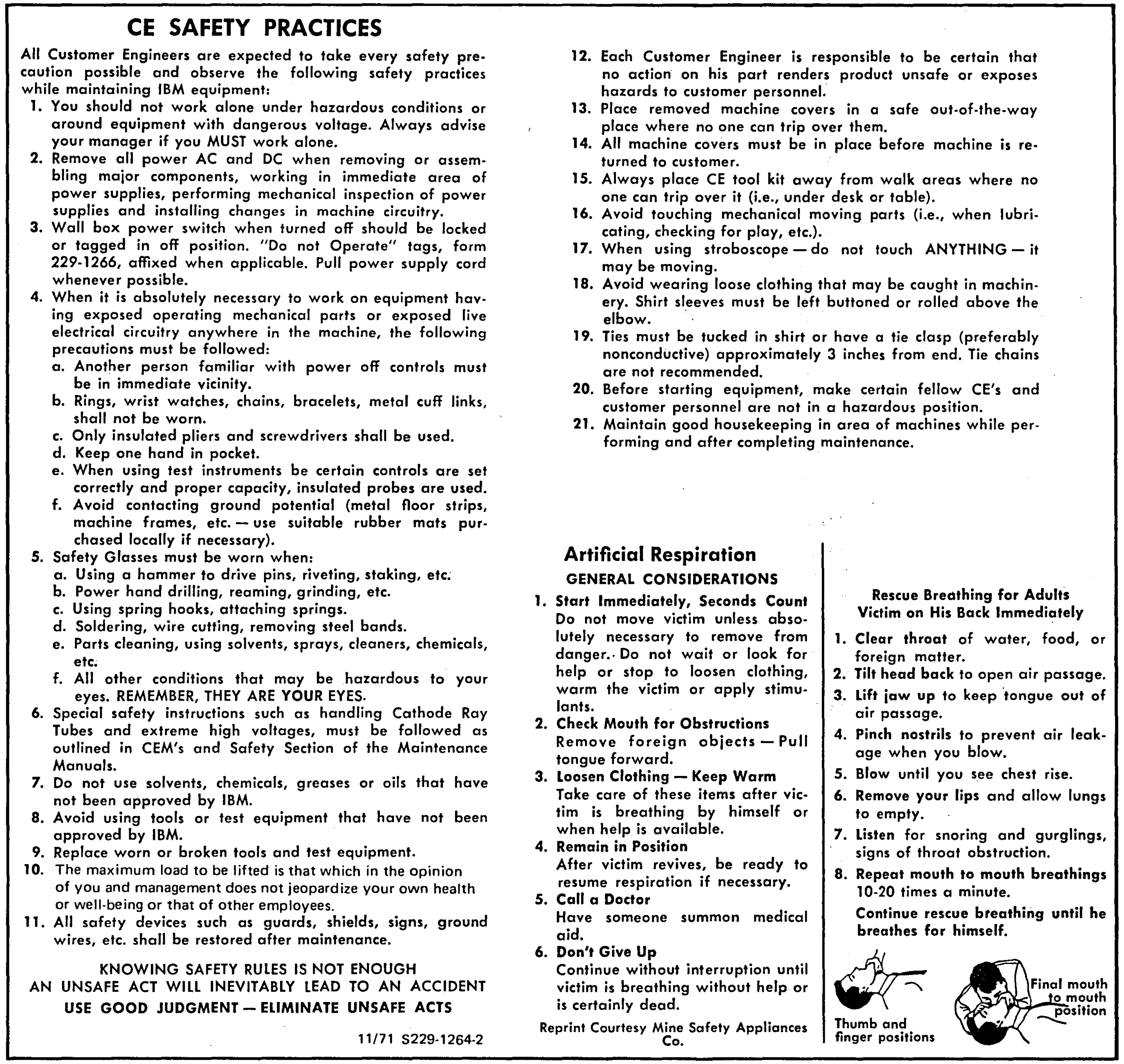
Ground current may reach dangerous levels. Never operate the system with the grounding conductor removed.

Line-Powered Equipment

Ground all line-powered test equipment through the third-wire grounding conductor in the power cord of the machine being tested.

Machine Warning Labels

Heed the warning labels placed in hazardous areas of the machines.



CE SAFETY PRACTICES

All Customer Engineers are expected to take every safety precaution possible and observe the following safety practices while maintaining IBM equipment:

1. You should not work alone under hazardous conditions or around equipment with dangerous voltage. Always advise your manager if you MUST work alone.
2. Remove all power AC and DC when removing or assembling major components, working in immediate area of power supplies, performing mechanical inspection of power supplies and installing changes in machine circuitry.
3. Wall box power switch when turned off should be locked or tagged in off position. "Do not Operate" tags, form 229-1266, affixed when applicable. Pull power supply cord whenever possible.
4. When it is absolutely necessary to work on equipment having exposed operating mechanical parts or exposed live electrical circuitry anywhere in the machine, the following precautions must be followed:
    a. Another person familiar with power off controls must be in immediate vicinity.
    b. Rings, wrist watches, chains, bracelets, metal cuff links, shall not be worn.
    c. Only insulated pliers and screwdrivers shall be used.
    d. Keep one hand in pocket.
    e. When using test instruments be certain controls are set correctly and proper capacity, insulated probes are used.
    f. Avoid contacting ground potential (metal floor strips, machine frames, etc. — use suitable rubber mats purchased locally if necessary).
5. Safety Glasses must be worn when:
    a. Using a hammer to drive pins, riveting, staking, etc.
    b. Power hand drilling, reaming, grinding, etc.
    c. Using spring hooks, attaching springs.
    d. Soldering, wire cutting, removing steel bands.
    e. Parts cleaning, using solvents, sprays, cleaners, chemicals, etc.
    f. All other conditions that may be hazardous to your eyes. REMEMBER, THEY ARE YOUR EYES.
6. Special safety instructions such as handling Cathode Ray Tubes and extreme high voltages, must be followed as outlined in CEM's and Safety Section of the Maintenance Manuals.
7. Do not use solvents, chemicals, greases or oils that have not been approved by IBM.
8. Avoid using tools or test equipment that have not been approved by IBM.
9. Replace worn or broken tools and test equipment.
10. The maximum load to be lifted is that which in the opinion of you and management does not jeopardize your own health or well-being or that of other employees.
11. All safety devices such as guards, shields, signs, ground wires, etc. shall be restored after maintenance.
12. Each Customer Engineer is responsible to be certain that no action on his part renders product unsafe or exposes hazards to customer personnel.
13. Place removed machine covers in a safe out-of-the-way place where no one can trip over them.
14. All machine covers must be in place before machine is returned to customer.
15. Always place CE tool kit away from walk areas where no one can trip over it (i.e., under desk or table).
16. Avoid touching mechanical moving parts (i.e., when lubricating, checking for play, etc.).
17. When using stroboscope — do not touch ANYTHING — it may be moving.
18. Avoid wearing loose clothing that may be caught in machinery. Shirt sleeves must be left buttoned or rolled above the elbow.
19. Ties must be tucked in shirt or have a tie clasp (preferably nonconductive) approximately 3 inches from end. Tie chains are not recommended.
20. Before starting equipment, make certain fellow CE's and customer personnel are not in a hazardous position.
21. Maintain good housekeeping in area of machines while performing and after completing maintenance.

KNOWING SAFETY RULES IS NOT ENOUGH
AN UNSAFE ACT WILL INEVITABLY LEAD TO AN ACCIDENT
USE GOOD JUDGMENT — ELIMINATE UNSAFE ACTS

11/71 S229-1264-2

Artificial Respiration

GENERAL CONSIDERATIONS

1. **Start Immediately, Seconds Count**
   Do not move victim unless absolutely necessary to remove from danger. Do not wait or look for help or stop to loosen clothing, warm the victim or apply stimulants.
2. **Check Mouth for Obstructions**
   Remove foreign objects — Pull tongue forward.
3. **Loosen Clothing — Keep Warm**
   Take care of these items after victim is breathing by himself or when help is available.
4. **Remain in Position**
   After victim revives, be ready to resume respiration if necessary.
5. **Call a Doctor**
   Have someone summon medical aid.
6. **Don't Give Up**
   Continue without interruption until victim is breathing without help or is certainly dead.

Reprint Courtesy Mine Safety Appliances Co.

Rescue Breathing for Adults
Victim on His Back Immediately

1. **Clear throat** of water, food, or foreign matter.
2. **Tilt head back** to open air passage.
3. **Lift jaw up** to keep tongue out of air passage.
4. **Pinch nostrils** to prevent air leakage when you blow.
5. **Blow** until you see chest rise.
6. **Remove your lips** and allow lungs to empty.
7. **Listen** for snoring and gurglings, signs of throat obstruction.
8. **Repeat mouth to mouth breathings** 10-20 times a minute.

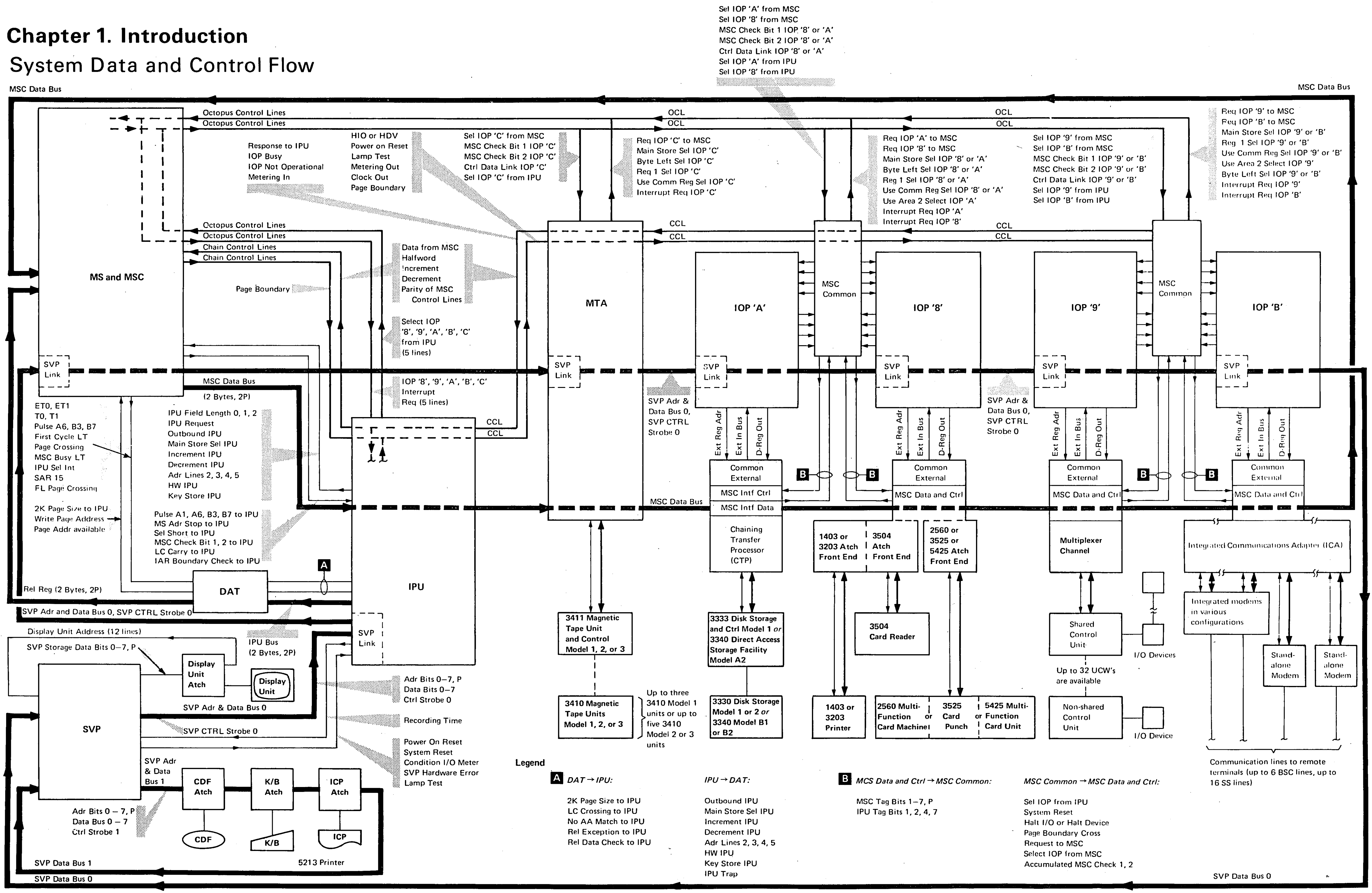
Continue rescue breathing until he breathes for himself.

Thumb and finger positions

Final mouth to mouth position

Chapter 1. Introduction

System Data and Control Flow



Legend

A *DAT → IPU:*

2K Page Size to IPU
LC Crossing to IPU
No AA Match to IPU
Rel Exception to IPU
Rel Data Check to IPU

IPU → DAT:

Outbound IPU
Main Store Sel IPU
Increment IPU
Decrement IPU
Adr Lines 2, 3, 4, 5
HW IPU
Key Store IPU
IPU Trap

B *MCS Data and Ctrl → MSC Common:*

MSC Tag Bits 1–7, P
IPU Tag Bits 1, 2, 4, 7

MSC Common → MSC Data and Ctrl:

Sel IOP from IPU
System Reset
Halt I/O or Halt Device
Page Boundary Cross
Request to MSC
Select IOP from MSC
Accumulated MSC Check 1, 2

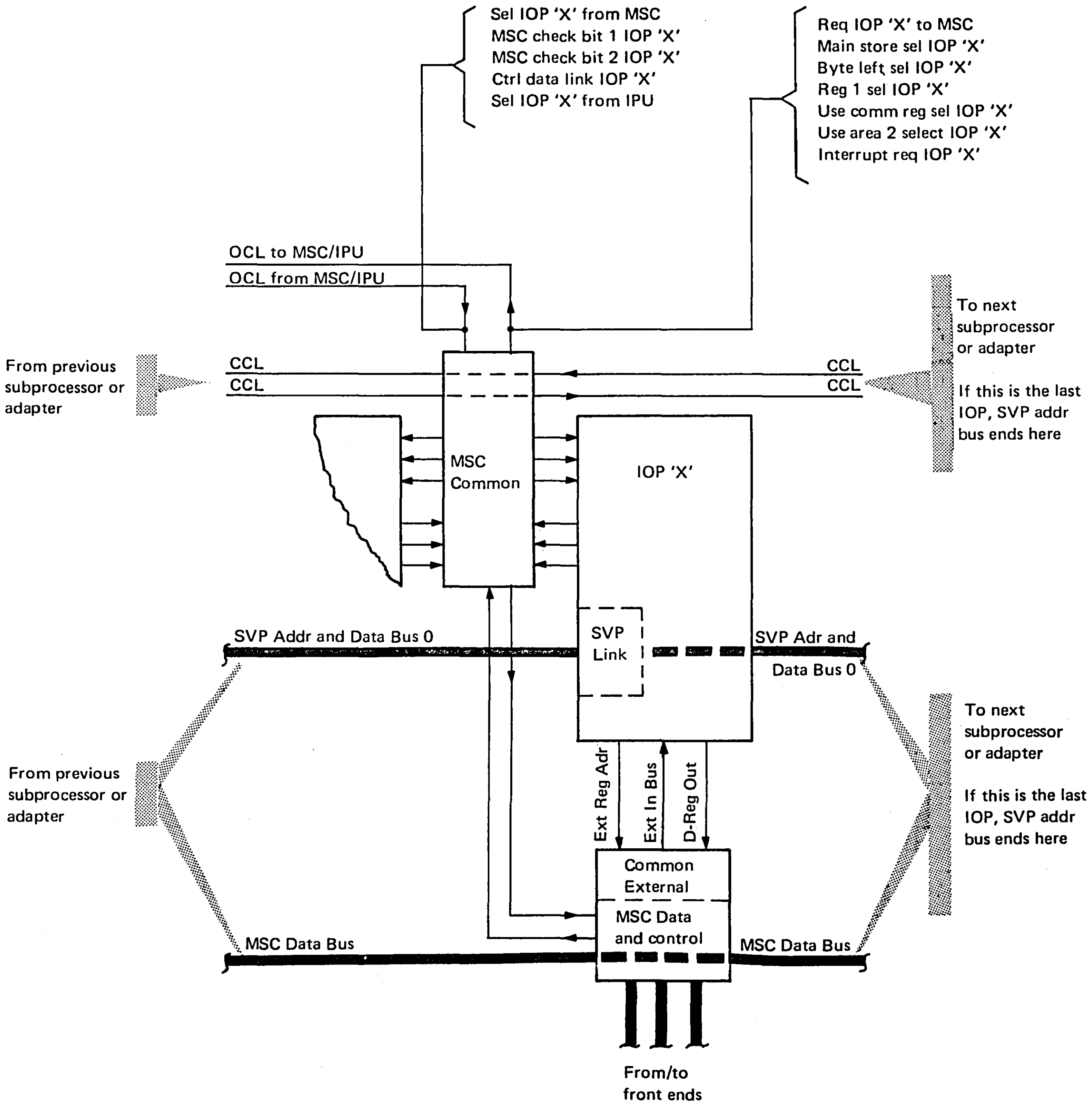
General Information

The input/output processor (IOP) is that part of the IBM System/370 Model 125 that controls one or more I/O devices.

After an IOP is selected and successfully started by the instruction processing unit (IPU), the IOP executes the different I/O instructions under control of its own microprogram and the IOP releases the IPU for further processing.

After the execution of an I/O instruction is completed, the IOP requests an interrupt.

The number of IOPs in a system varies according to the system configuration.



System Interval Buses

The interconnections between the subprocessors and adapters of the IBM System/370 Model 125 are made by an internal bus system.

This internal bus system consists of the following:

- 18-bit wide MSC data bus (2 bytes) + 2P
- 18-bit wide SVP address and data bus 0. This SVP address and data bus in turn consists of
  - 9 bit wide 'address' portion
  - 8 bit wide 'data' portion
  - 1 line named 'control strobe'.
- 16-bit wide bus, containing chain control lines (CCL)
- 12-bit wide bus, containing octopus control lines (OCL). This is the number of lines used by *one* IOP, and there are also lines existing, that are common for two IOPs.
- SVP address and data bus 1 interconnect SVP and SVP console units.

All buses except the OCLs are connected from one subprocessor or adapter to the next one throughout the whole system.

OCLs are to be considered as individual control lines interconnecting IOPs and MSC/IPU directly.

Connections between IOPs and front ends are accomplished via common external cards.

For detailed description of CCLs and OCLs refer to Pages 2-120 and 2-130.

Channel Concept

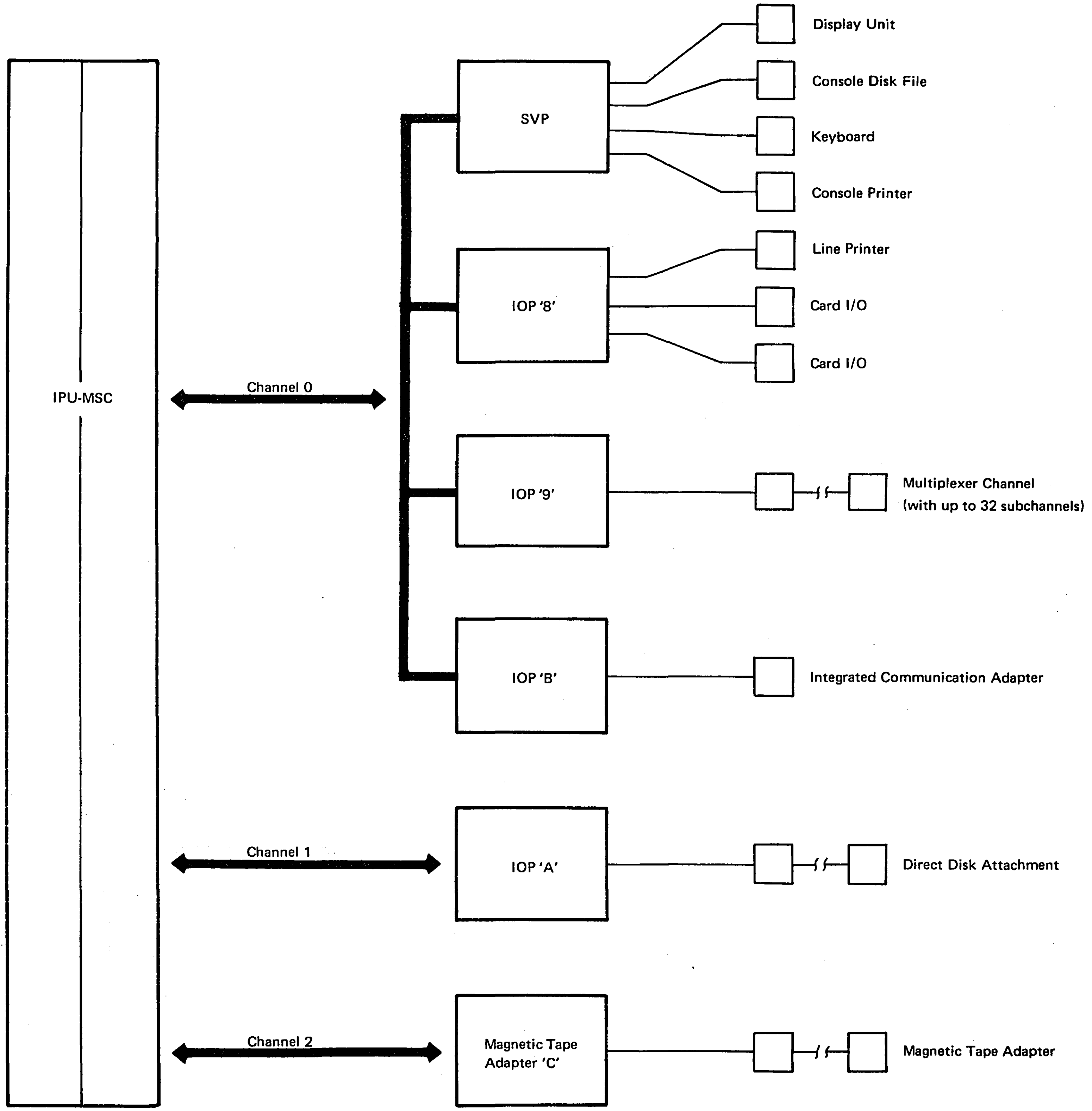
I/O devices can be attached to the IBM System/370 Model 125 via native attachments or channels.

To the operating system all I/O devices appear as channel attached. This means the I/O devices are started by a start input/output (SIO) instruction.

This I/O instruction contains channel and device address. The actual operation, that is to be performed by the I/O device "is defined in the Channel Command Word (CCW); refer to Page 2-015" Instructions and Formats.

Before the actual operation is started, an initial status is presented. This initial status indicates, whether or not an I/O device is ready to execute commands.

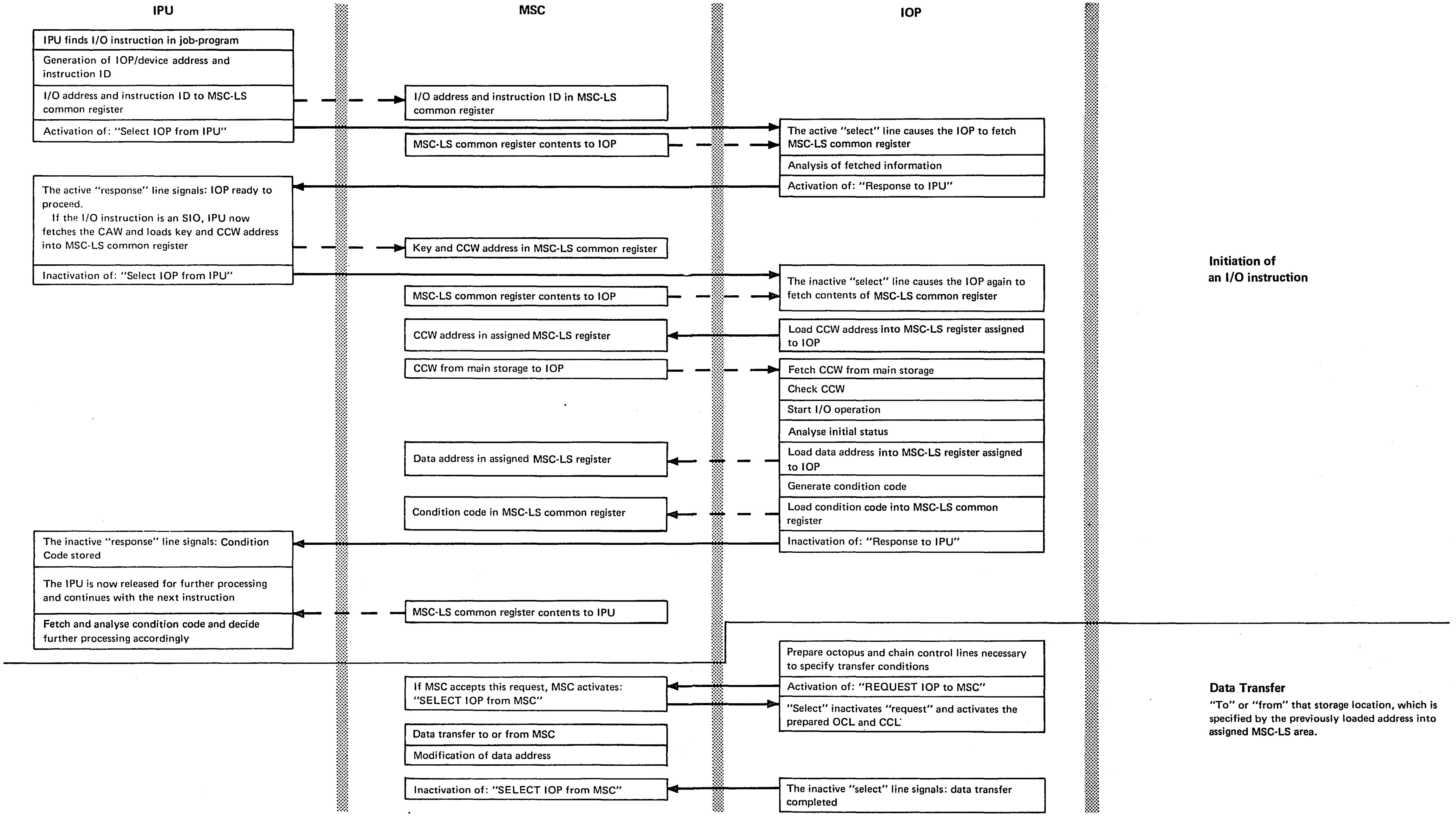
After the actual operation is completed, an ending status is presented. This ending status indicates whether or not the command or commands were executed successfully. Refer to Pages: 2-020, 2-026.



Principle of Operation

Initiation of an I/O Instruction Data Transfer

The principle of operation shows the communication between IPU, MSC, and IOP.



IPU	MSC	IOP
IPU finds I/O instruction in job-program		
Generation of IOP/device address and instruction ID		
I/O address and instruction ID to MSC-LS common register	I/O address and instruction ID in MSC-LS common register	
Activation of: "Select IOP from IPU"		The active "select" line causes the IOP to fetch MSC-LS common register
	MSC-LS common register contents to IOP	
		Analysis of fetched information
		Activation of: "Response to IPU"
The active "response" line signals: IOP ready to proceed. If the I/O instruction is an SIO, IPU now fetches the CAW and loads key and CCW address into MSC-LS common register	Key and CCW address in MSC-LS common register	
Inactivation of: "Select IOP from IPU"		The inactive "select" line causes the IOP again to fetch contents of MSC-LS common register
	MSC-LS common register contents to IOP	
	CCW address in assigned MSC-LS register	Load CCW address into MSC-LS register assigned to IOP
	CCW from main storage to IOP	Fetch CCW from main storage
		Check CCW
		Start I/O operation
		Analyse initial status
	Data address in assigned MSC-LS register	Load data address into MSC-LS register assigned to IOP
		Generate condition code
	Condition code in MSC-LS common register	Load condition code into MSC-LS common register
The inactive "response" line signals: Condition Code stored		Inactivation of: "Response to IPU"
The IPU is now released for further processing and continues with the next instruction		
Fetch and analyse condition code and decide further processing accordingly	MSC-LS common register contents to IPU	
		Prepare octopus and chain control lines necessary to specify transfer conditions
	If MSC accepts this request, MSC activates: "SELECT IOP from MSC"	Activation of: "REQUEST IOP to MSC"
		"Select" inactivates "request" and activates the prepared OCL and CCL
	Data transfer to or from MSC	
	Modification of data address	
	Inactivation of: "SELECT IOP from MSC"	The inactive "select" line signals: data transfer completed

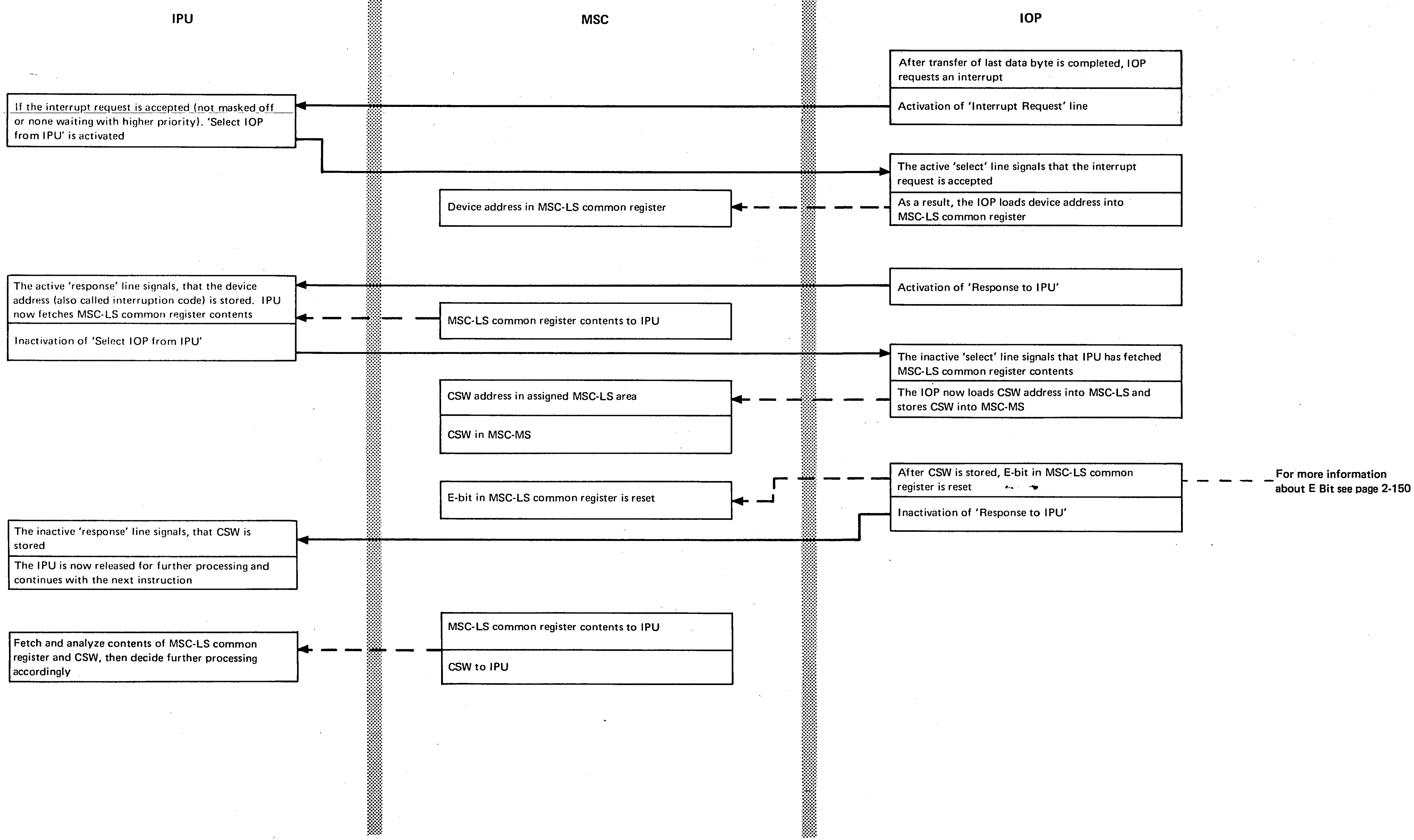
Initiation of an I/O instruction

Data Transfer

"To" or "from" that storage location, which is specified by the previously loaded address into assigned MSC-LS area.

Principle of Operation (continued)

Interrupt Request



After transfer of last data byte is completed, IOP requests an interrupt

Activation of 'Interrupt Request' line

If the interrupt request is accepted (not masked off or none waiting with higher priority). 'Select IOP from IPU' is activated

The active 'select' line signals that the interrupt request is accepted

As a result, the IOP loads device address into MSC-LS common register

Device address in MSC-LS common register

Activation of 'Response to IPU'

The active 'response' line signals, that the device address (also called interruption code) is stored. IPU now fetches MSC-LS common register contents

MSC-LS common register contents to IPU

Inactivation of 'Select IOP from IPU'

The inactive 'select' line signals that IPU has fetched MSC-LS common register contents

The IOP now loads CSW address into MSC-LS and stores CSW into MSC-MS

CSW address in assigned MSC-LS area

CSW in MSC-MS

After CSW is stored, E-bit in MSC-LS common register is reset

For more information about E Bit see page 2-150

E-bit in MSC-LS common register is reset

Inactivation of 'Response to IPU'

The inactive 'response' line signals, that CSW is stored

The IPU is now released for further processing and continues with the next instruction

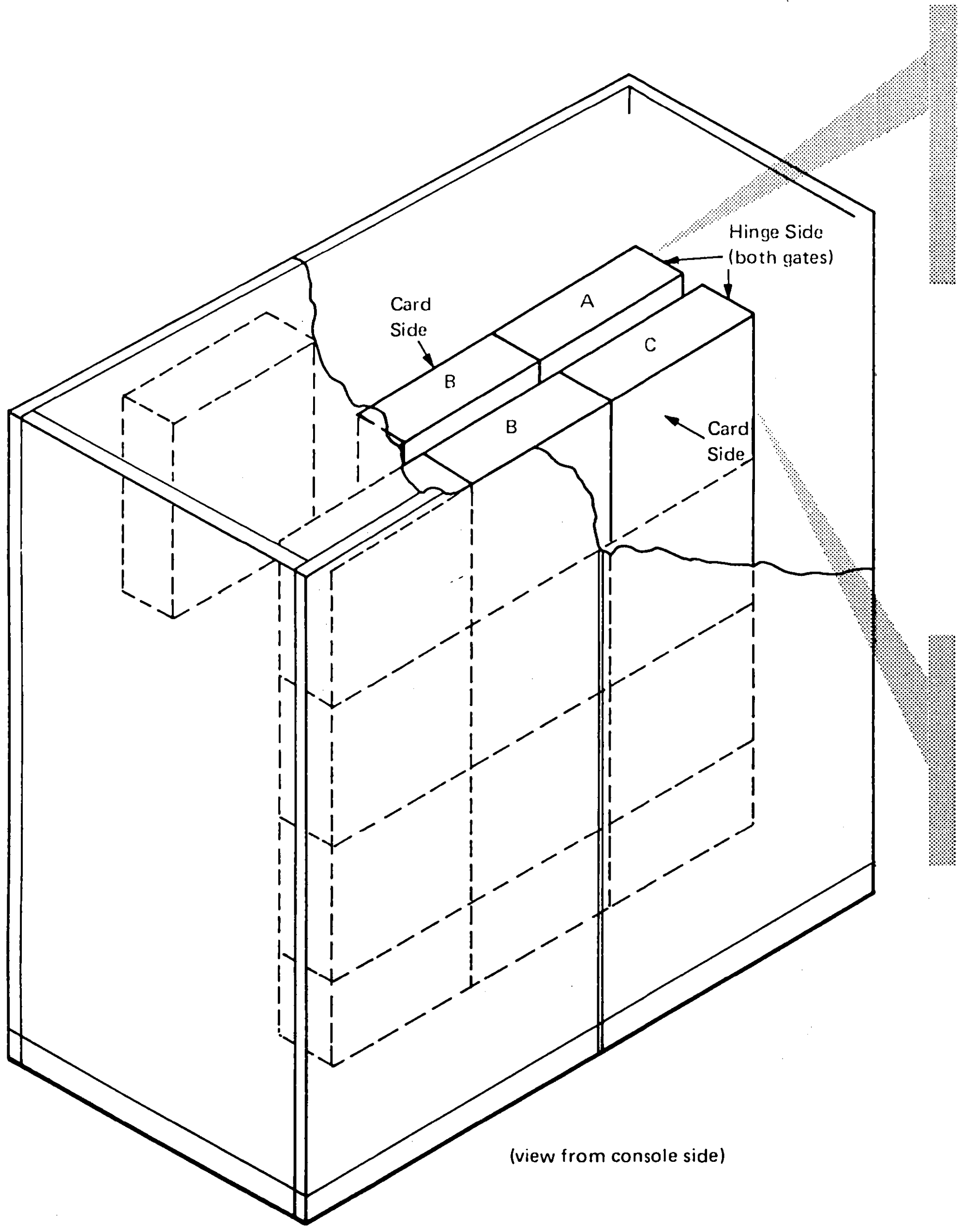
MSC-LS common register contents to IPU

CSW to IPU

Fetch and analyze contents of MSC-LS common register and CSW, then decide further processing accordingly

Physical Locations (seen from card side)

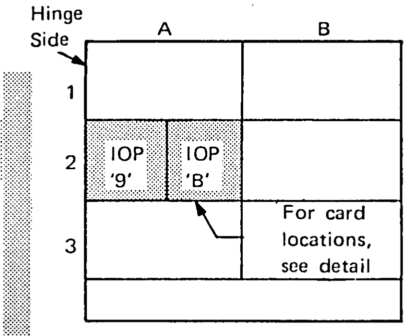
IOP Board Locations



(view from console side)

Gate 01B

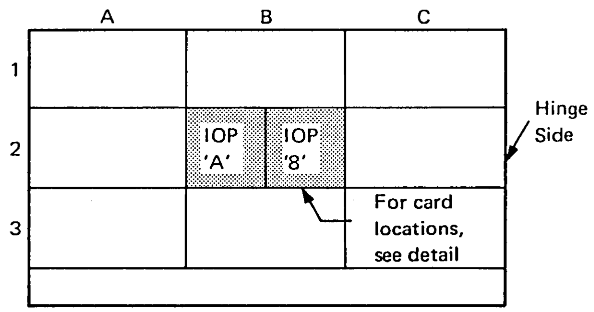
IOP '9' controls the multiplexer channel
IOP 'B' controls the communication adapter



(view from card side)

Gate 01A

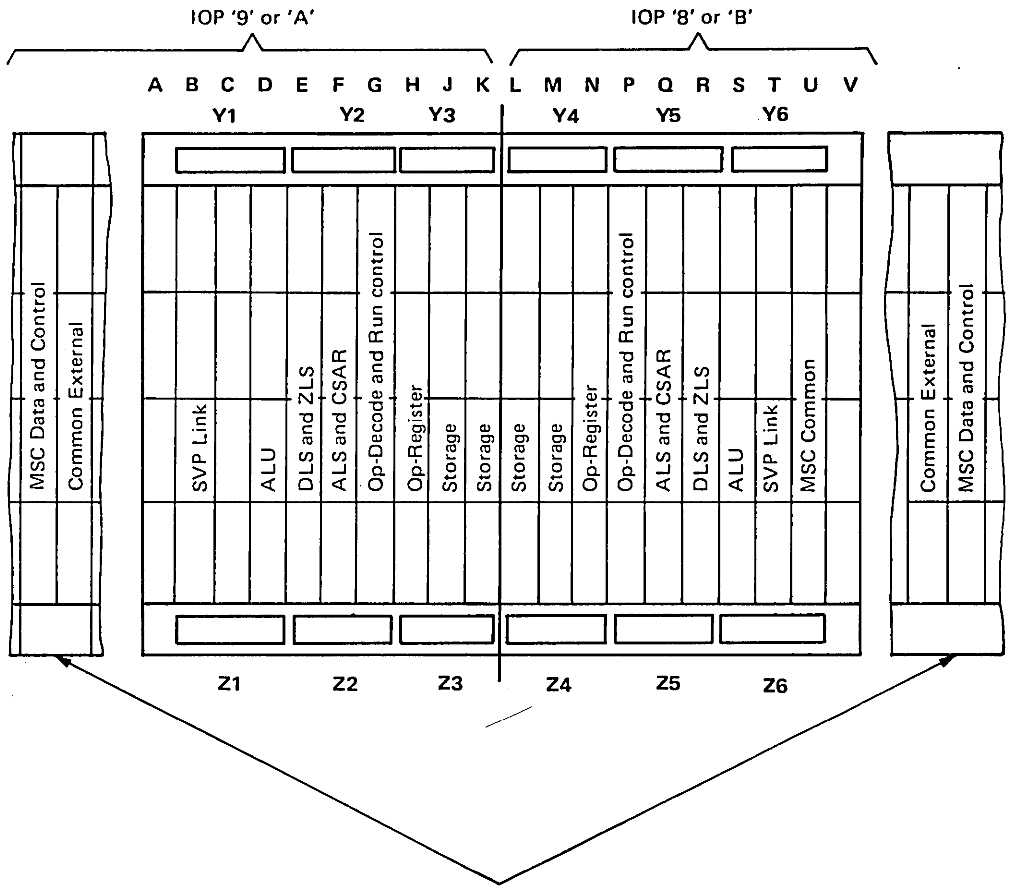
IOP '8' controls card I/Os and printer
IOP 'A' controls the direct disk attachment



(view from card side)

IOP '8' controlling Card I/Os and Printer
IOP '9' controlling Multiplexer Channel
IOP 'A' controlling Direct Disk Attachment
IOP 'B' controlling Integrated Communications Adapter

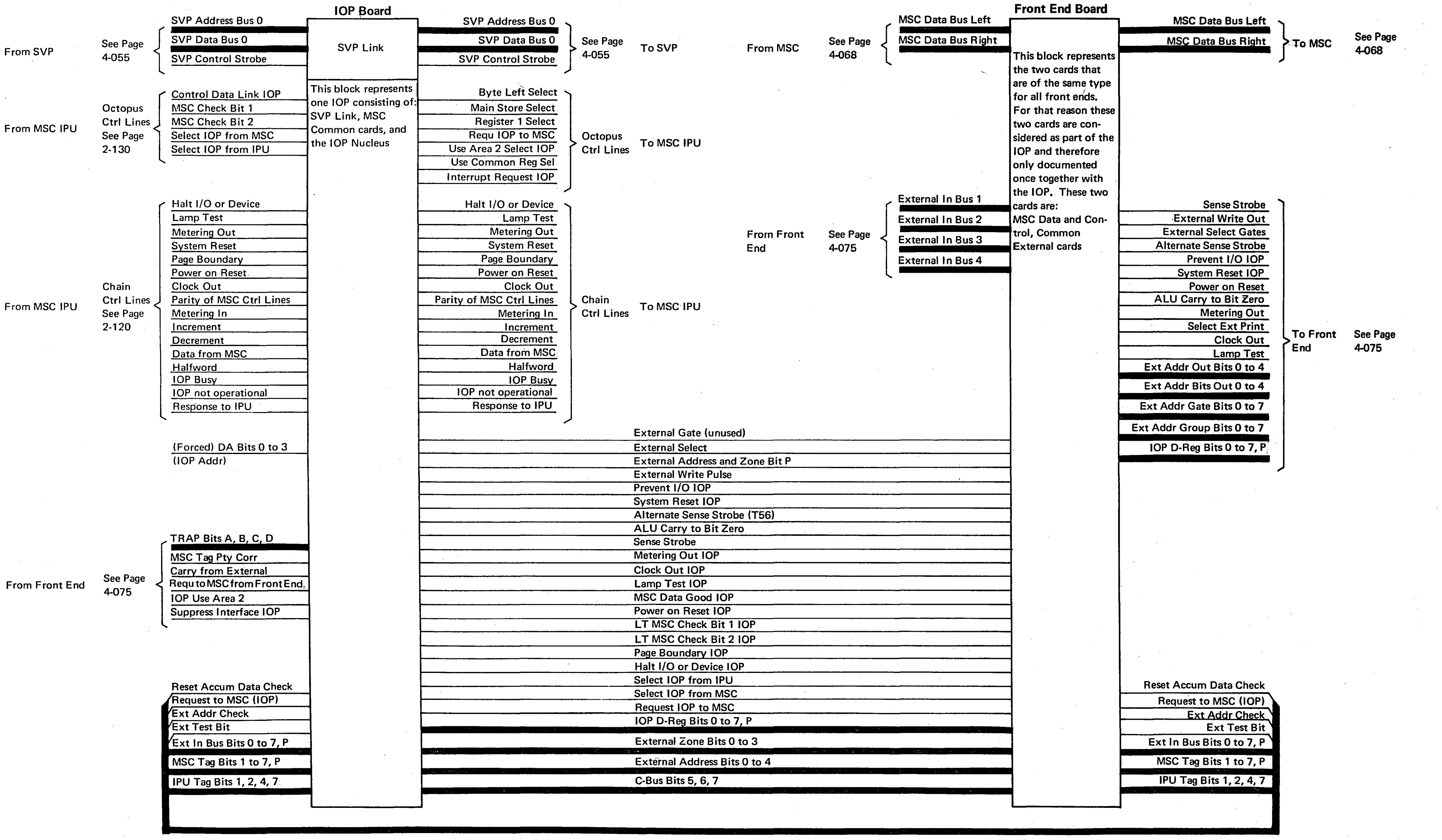
IOP Card Locations



For physical locations of these cards, refer to the appropriate Front End documentation

Signal Interface Chart

This page shows interconnections to/from IOPs, grouped together according to their functions.



Chapter 2. Principle of Operations

General Data Flow

This general IOP data flow shows all the components of the Input/Output Processor (IOP).

Part of this general data flow is also used in Chapter 3 for the explanation of the IOP microinstructions.

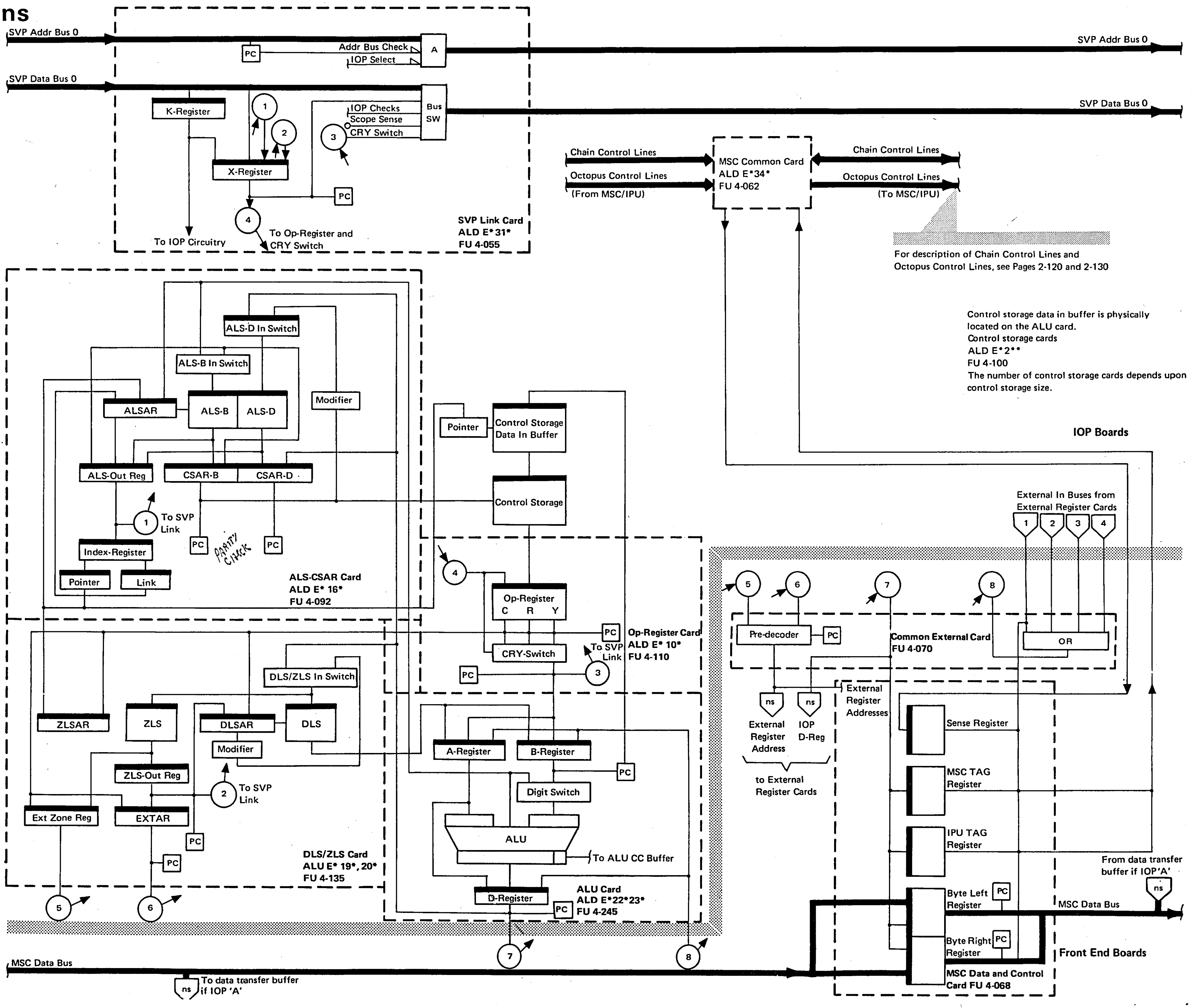
A more detailed data flow is provided on pages: 4-010, 4-020.

Because two sets of ALDs are provided (one for IOPs '8' and 'B', letters EA) (another for IOPs '9' and 'A', letters EB) and a number of ALD pages are used to document one card (designated by three digits) references to ALDs are made in the following way:

E*16*

- These three characters define the group of ALD pages for one card. The asterisk here stands for the figures 0 to 9, which means that ten ALD pages may exist for this card and the reference points to one of these ALD pages.
- The letter E defines IOP ALDs. The asterisk here stands for either A or B. These letters A or B designate a set of ALDs that is valid for two IOPs (see above).

ALS - ADDRESS LOCAL STOR. (32 x 18)
ALS-B BLOCK ADDR. OR INDEX WORD
ALS-D DISPLACEMENT " "
ALSAR ADDRESS Reg for ALS.
CSAR ADDRESS Reg. of CONTROL STORAGE (B - block, D - Displacement)
DAR DATA ADDRESS Register - IN ALS
DLS DATA LOCAL STORAGE
DLSAR ADDR. Reg. for DLS
EXTAR REG. for EXTERNAL ADDRESS
MIAR MAIN INST. ADDR Reg. IN ALS
SIAR Sbrtine INST. ADDR Reg. IN ALS
TSR Time Slice Reg. (ZLS AND ALS)
ZLS ZONE LOCAL STORAGE
ZLSAR ADDR. Reg. for ZLS
ZONE group (4, 8, 16) OF DLS OR EXTERNAL Registers



For description of Chain Control Lines and Octopus Control Lines, see Pages 2-120 and 2-130

Control storage data in buffer is physically located on the ALU card.
Control storage cards
ALD E*2**
FU 4-100
The number of control storage cards depends upon control storage size.

Instructions and Formats

I/O Instructions

There are seven I/O instructions:

SIO – Start I/O
HIO – Halt I/O
HDV – Halt Device
TIO – Test I/O
TCH – Test Channel
STIDC – Store Channel Identifier

SIOF – Start I/O Fast Release

Format

0	8	15		20 31
Op Code	(Not used)		Register Address	Displacement

This bit is used for 'HIO' and 'HDV' instructions only

All I/O instructions use this format. The displacement is added to the contents of a specified register; the result represents a main storage location that contains the channel and device addresses.

0	16	24 31
(Not used)	Channel Address	Device Address

After successful generation of the channel and device addresses, the CAW is fetched.

'SIO' Instruction

The 'SIO' instruction initiates an I/O operation. The address part of the instruction specifies the channel and the device.

'Halt' Instructions

Any 'Halt' instruction terminates an operation that was started by an 'SIO' instruction. The termination is performed in the electronic circuits, while the mechanical operation of the device runs to its normal end.

'HIO' Instruction

There are two instances where the 'HIO' instruction is used (assume that devices A and B are connected to a channel):

(a) If both devices are working in multiplex mode the addressed device is stopped without affecting the other device.

(b) If device A is not working and device B is working in burst mode, an 'HIO' instruction to device A would stop the burst operation of device B.

'HDV' Instruction

This instruction stops the addressed device. While one device is operating in burst mode, an 'HDV' instruction to another device is not performed until the burst operation is complete.

'TIO' Instruction

A 'TIO' instruction sets a condition code into the 'PSW' to indicate the status of the addressed channel, subchannel, and device. A 'CSW' may be stored.

This instruction may also be used to clear interrupts.

'TCH' Instruction[1]

A 'TCH' instruction sets a condition code into the 'PSW' to indicate the status of the addressed channel.

'STIDC' Instruction

An 'STIDC' instruction sets four bytes of information into main storage starting at location 168 (decimal).

0	4	16 31
Channel Type	Channel Model Number	Maximum IOEL Length (Set to zero)

This instruction is handled entirely by the IPU and does not require any IOP action.IBM System/370 Model 125 bits 16 to 31 are set to zero.

'SIOF' Instruction

An 'SIOF' instruction in the IBM System/370 Model 125 is treated as a normal 'SIO' instruction.

Control Word Formats

CAW (Channel Address Word)

- The CAW is located in the MSC main storage location 72 decimal (48 hexadecimal).

0	4	8 31
Key	0000	Command Address

Specifies the main storage location of the first CCW

CCW (Channel Command Word)

- The CCW is transferred and stored in the IOP that is associated with the addressed device.

0	8	32	38	40	48 63
Command	Data Address or (for TIC) New CCW Address	Flags	00	(Not used)	Length Count or Byte Count

Read:	Transfer data from device to MSC
Write:	Transfer data from MSC to device
Read backward:	Read, but in reverse order
SNS:	Device information to MSC (sense bytes)
CTL:	Set up conditions in addressed control unit and device
TIC (transfer in channel):	Chaining of CCWs that are not located in adjacent doubleword main storage locations. TIC does not initiate an I/O operation

Bit 32:	CD (chain data)
Bit 33:	CC (chain command)
Bit 34:	SILI (suppress incorrect length indication)
Bit 35:	Skip (suppress transfer of zero bytes)
Bit 36:	PCI (program-controlled interrupt – allows interrupts during chaining)
Bit 37:	IDA (indirect data address)

After the CCW is transferred to the IOP, and the command does not contain the TIC, data transfer starts with the specified data address.

If the IDA flag is set, the data address specifies an indirect data address. This means that the actual data address has to be composed in a specified time before the data transfer can start.

Three conditions for normal ending of data transfers may occur:

1. Byte count 0 and I/O count 0.
2. Byte count 0 and I/O count not 0.
3. Byte count not 0 and I/O count 0.

CSW (Channel Status Word)

- The CSW is located in MSC main storage location 64 decimal (40 hexadecimal).

0	4	8	32	40	48 63
Key	0000	Command Address	Unit or Device Status	Channel Status	Residual Count

Last CCW address + 8

Bit 32:	Attention
Bit 33:	Status modifier
Bit 34:	Control unit end
Bit 35:	Busy
Bit 36:	Channel end
Bit 37:	Device end
Bit 38:	Unit check
Bit 39:	Unit exception

Bit 40:	PCI (program-controlled interrupt)
Bit 41:	Incorrect length
Bit 42:	Program check
Bit 43:	Protection check
Bit 44:	Channel data check
Bit 45:	Channel control check
Bit 46:	Interface control check
Bit 47:	Chaining check

Number of bytes that have not been transferred

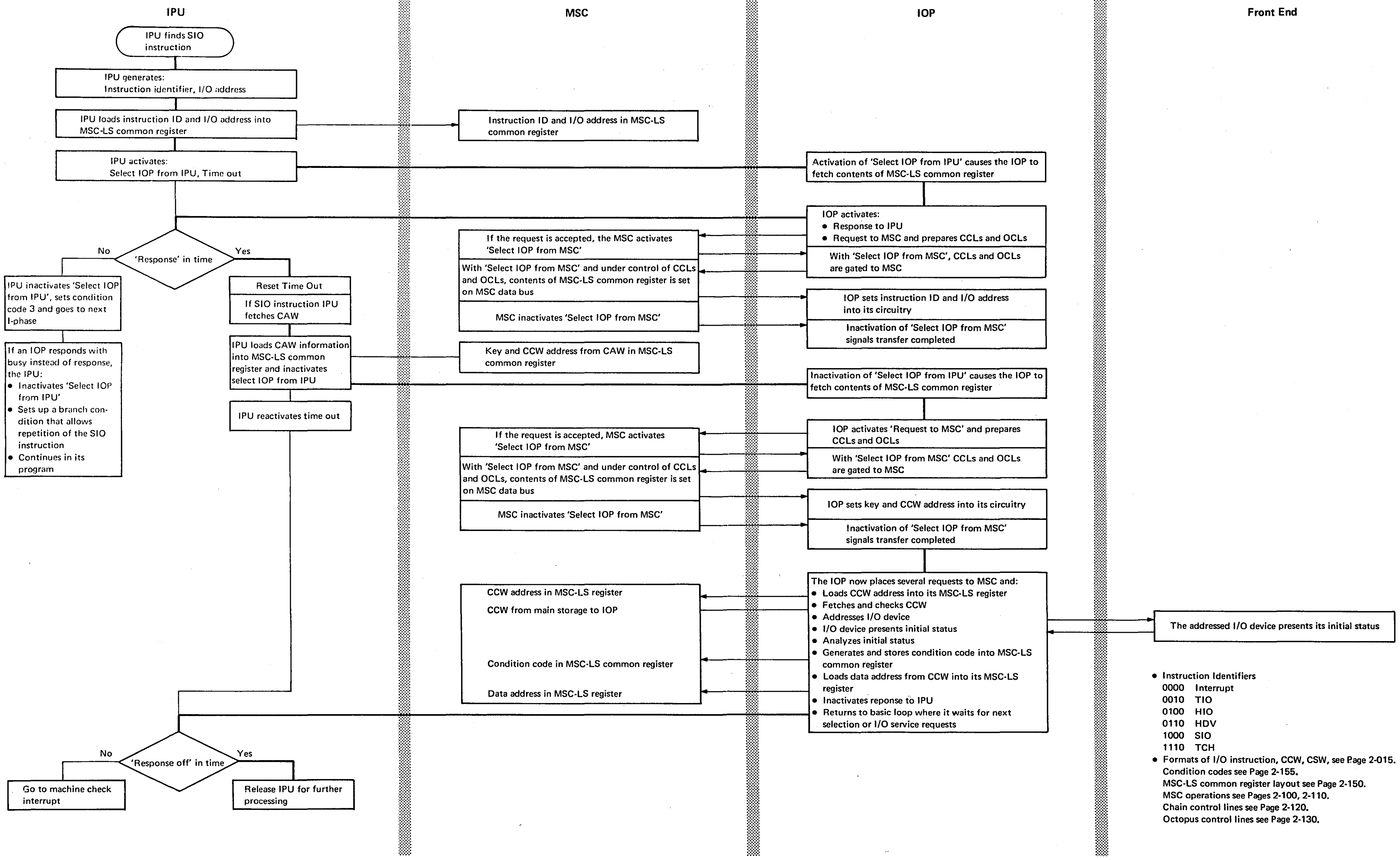
After the last byte is transferred, the IOP requests an interrupt. If the interrupt is accepted, an interruption code (device address) is set into the PSW and either the status or the complete CSW is stored according to conditions occurring during the I/O operation. If the interrupt is not accepted, the interrupt is stacked. (See 'Interrupts' on Page 2-155.)

Principle of IOP Operation

Initiation of an SIO

The flowcharts on this and the following pages show the communication between IPU – MSC – IOPs.

The initiation of an SIO is shown.



- Instruction Identifiers

0000	Interrupt
0010	TIO
0100	HIO
0110	HDV
1000	SIO
1110	TCH

- Formats of I/O instruction, CCW, CSW, see Page 2-015.
  Condition codes see Page 2-155.
  MSC-LS common register layout see Page 2-150.
  MSC operations see Pages 2-100, 2-110.
  Chain control lines see Page 2-120.
  Octopus control lines see Page 2-130.

Data Transfer for IOP's '8', '9', 'B'

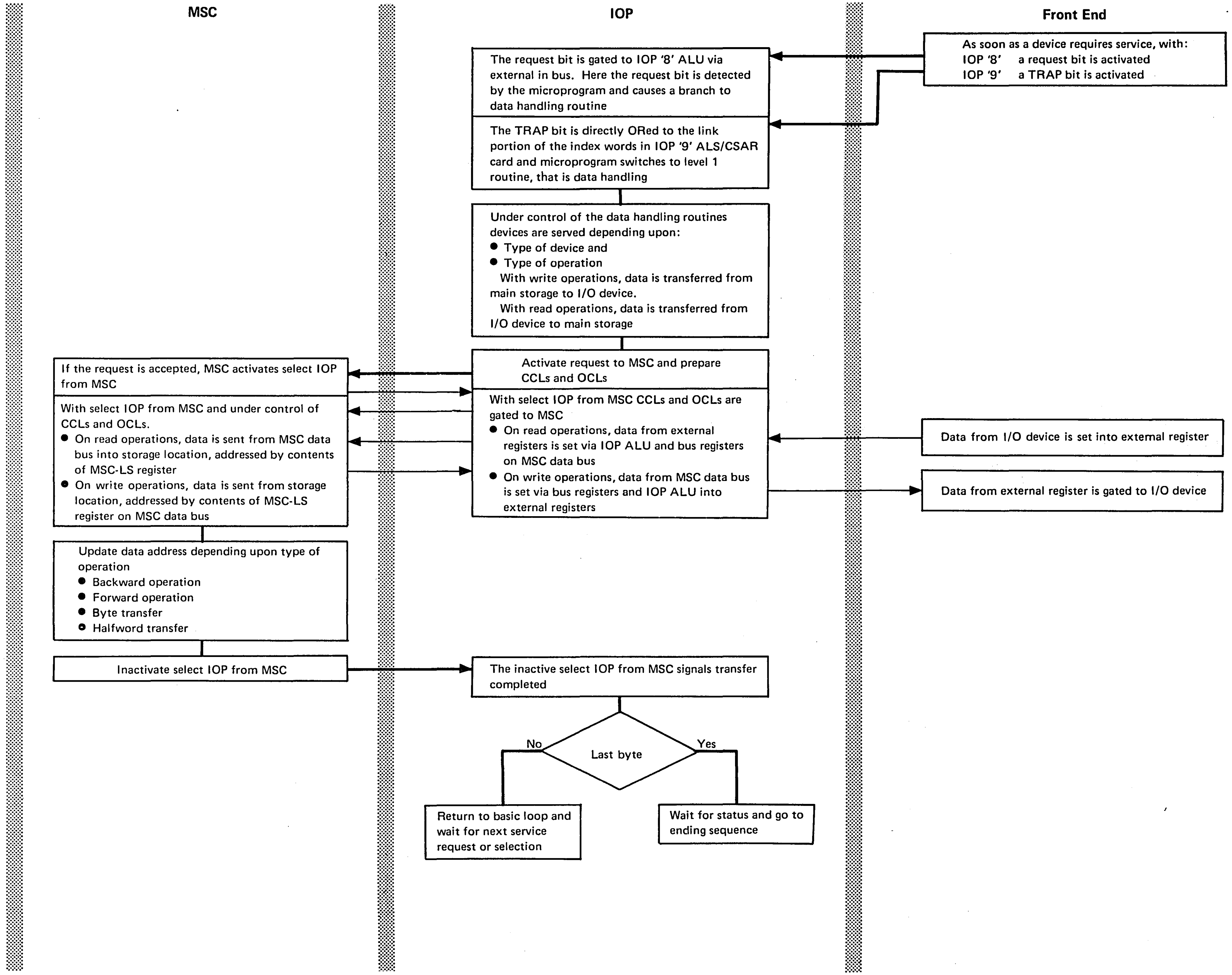
The flowchart on this page shows communication between MSC and IOPs. The data transfer of an I/O instruction is shown as it is performed by the IOPs '8', '9', 'B'.

IOPs serving I/O devices with relatively low data rates control their data transfers to and from MSC by microprogram.

With an active service request, microprogram passes to data handling routine.

Data Flow for:

- Write operations is shown on Pages 2-030, 2-035, 2-050.
- Read operations is shown on Pages 2-040, 2-045, 2-055.



As soon as a device requires service, with:
IOP '8' a request bit is activated
IOP '9' a TRAP bit is activated

The request bit is gated to IOP '8' ALU via external in bus. Here the request bit is detected by the microprogram and causes a branch to data handling routine

The TRAP bit is directly ORed to the link portion of the index words in IOP '9' ALS/CSAR card and microprogram switches to level 1 routine, that is data handling

Under control of the data handling routines devices are served depending upon:
- Type of device and
- Type of operation

With write operations, data is transferred from main storage to I/O device.

With read operations, data is transferred from I/O device to main storage

Activate request to MSC and prepare CCLs and OCLs

If the request is accepted, MSC activates select IOP from MSC

With select IOP from MSC CCLs and OCLs are gated to MSC
- On read operations, data from external registers is set via IOP ALU and bus registers on MSC data bus
- On write operations, data from MSC data bus is set via bus registers and IOP ALU into external registers

With select IOP from MSC and under control of CCLs and OCLs.
- On read operations, data is sent from MSC data bus into storage location, addressed by contents of MSC-LS register
- On write operations, data is sent from storage location, addressed by contents of MSC-LS register on MSC data bus

Data from I/O device is set into external register

Data from external register is gated to I/O device

Update data address depending upon type of operation
- Backward operation
- Forward operation
- Byte transfer
- Halfword transfer

Inactivate select IOP from MSC

The inactive select IOP from MSC signals transfer completed

Return to basic loop and wait for next service request or selection

Wait for status and go to ending sequence

Principle of IOP Operation (continued)

Data Transfer IOP 'A'

The flowchart shows communication between MSC and IOP'A'.

The data transfer of an I/O instruction is shown as it is performed by IOP'A'.

IOPs serving high speed I/O devices, do not have microprogram controlled data transfers, but special circuitry ensures proper communication between I/O devices and MSC.

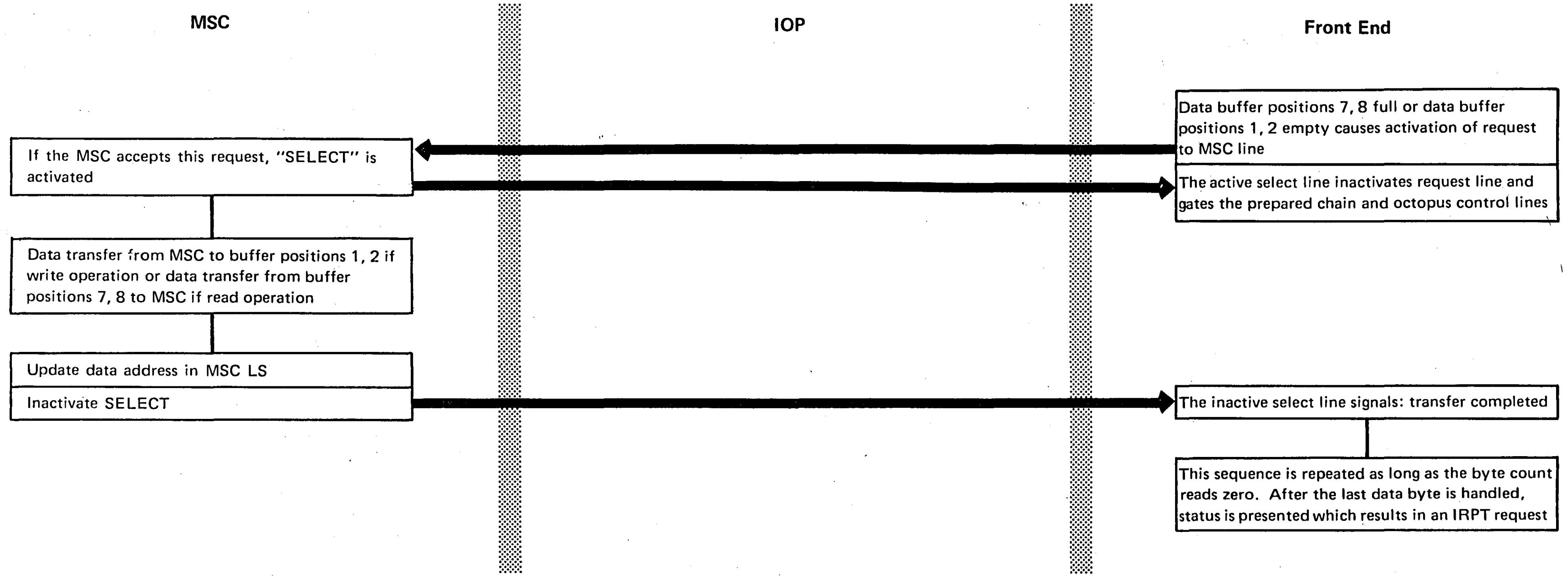
This circuitry also includes a 4 halfword data buffer to compensate delays until requests to MSC are accepted.

This data buffer is used in both directions:

- MSC to I/O device (write operation)
- I/O device to MSC (read operation).

Also some of the control lines (octopus and chain control lines) which define the transfer conditions, are generated by special circuitry.

Consequently gating of part of the microprogram generated control lines to MSC (by the active "select" line) has to be prevented. This is done by the line "Suppress Interface IOP". (See page 4-062 and appropriate front end documentation, e.g. *IBM 3125 Processing Unit, 3330 Direct Disk Attachment,MLM,* SY33-1073



Data buffer positions 7, 8 full or data buffer positions 1, 2 empty causes activation of request to MSC line

If the MSC accepts this request, "SELECT" is activated

The active select line inactivates request line and gates the prepared chain and octopus control lines

Data transfer from MSC to buffer positions 1, 2 if write operation or data transfer from buffer positions 7, 8 to MSC if read operation

Update data address in MSC LS

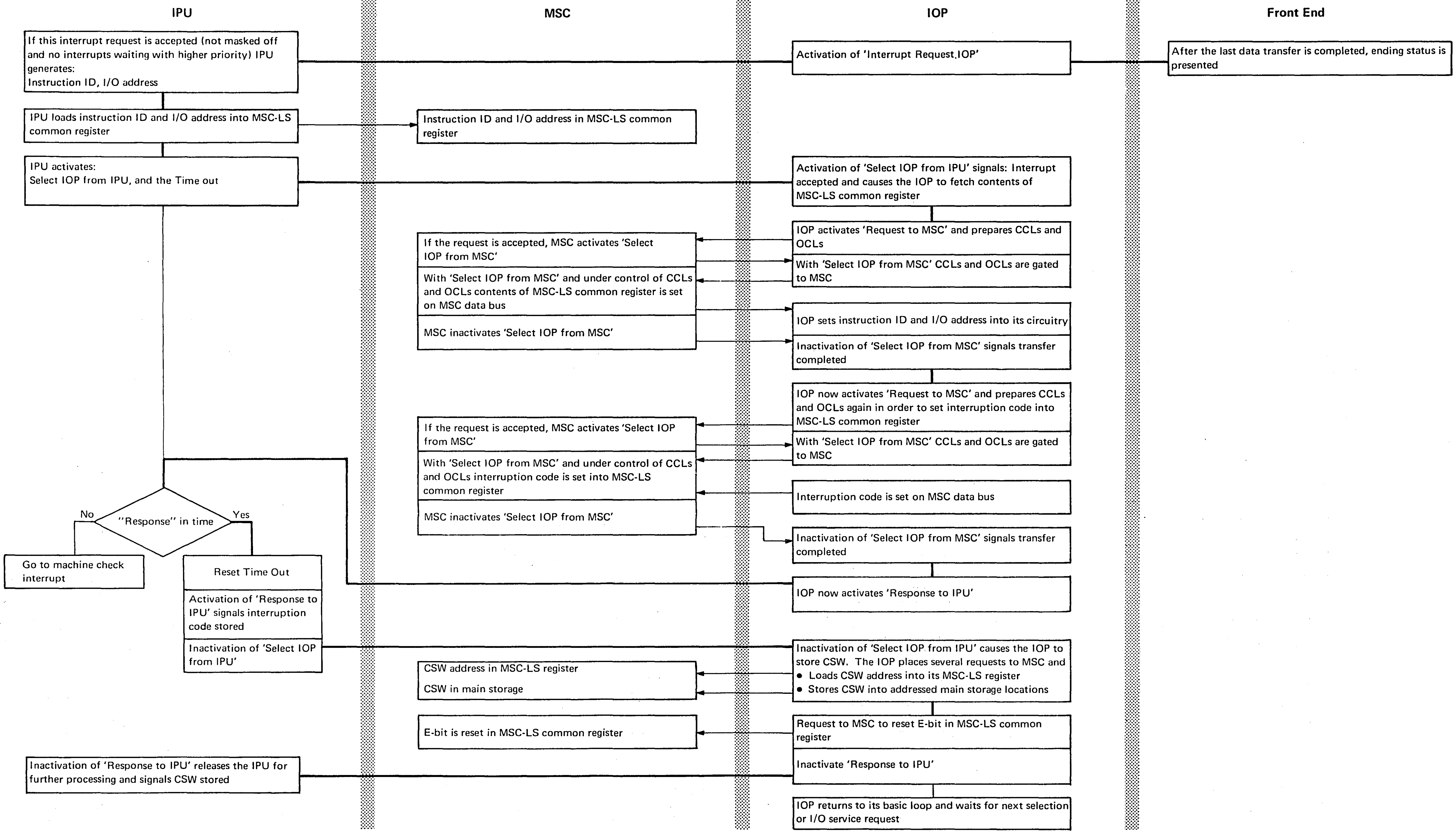
Inactivate SELECT

The inactive select line signals: transfer completed

This sequence is repeated as long as the byte count reads zero. After the last data byte is handled, status is presented which results in an IRPT request

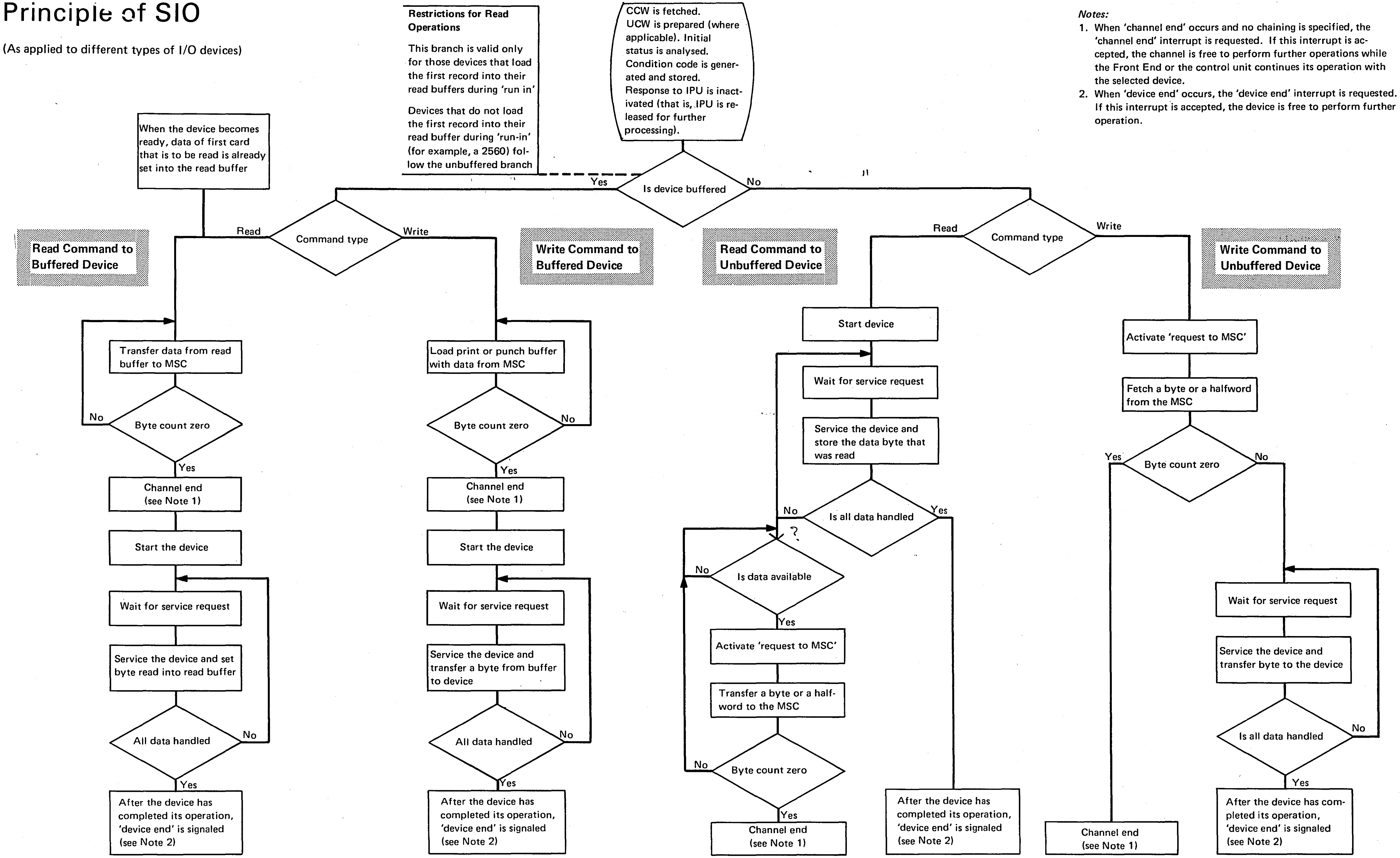
Interrupt Request

The flowchart shows communication between IPU, MSC, and IOPs. The ending sequence is shown.



Principle of SIO

(As applied to different types of I/O devices)



Notes:

1. When 'channel end' occurs and no chaining is specified, the 'channel end' interrupt is requested. If this interrupt is accepted, the channel is free to perform further operations while the Front End or the control unit continues its operation with the selected device.
2. When 'device end' occurs, the 'device end' interrupt is requested. If this interrupt is accepted, the device is free to perform further operation.

This page is intentionally left blank

IOP Write Operation

(Data Transfer from MSC to I/O Device)

Data Transfer IOPs '8', '9', 'B' Write Operation

Those I/O devices which do not have a buffer in IOP Control Storage.

The data flow is additional to the flowchart on page 2-022 (data transfer). It shows the data path for a write operation.

A write operation (related to IOPs) means, outbound operation for the MSC under control of chain control lines and octopus control lines. (CCLs and OCLs are not shown here.)

For more details refer to pages:

2-100 MSC Local Storage Operations
2-110 MSC Main Storage Operations
2-120 Description of CCLs
2-130 Description of OCLs.

A write operation consists of 5 main steps:

1 Sets I/O data from the addressed main storage location via the bus out registers onto the MSC data bus.

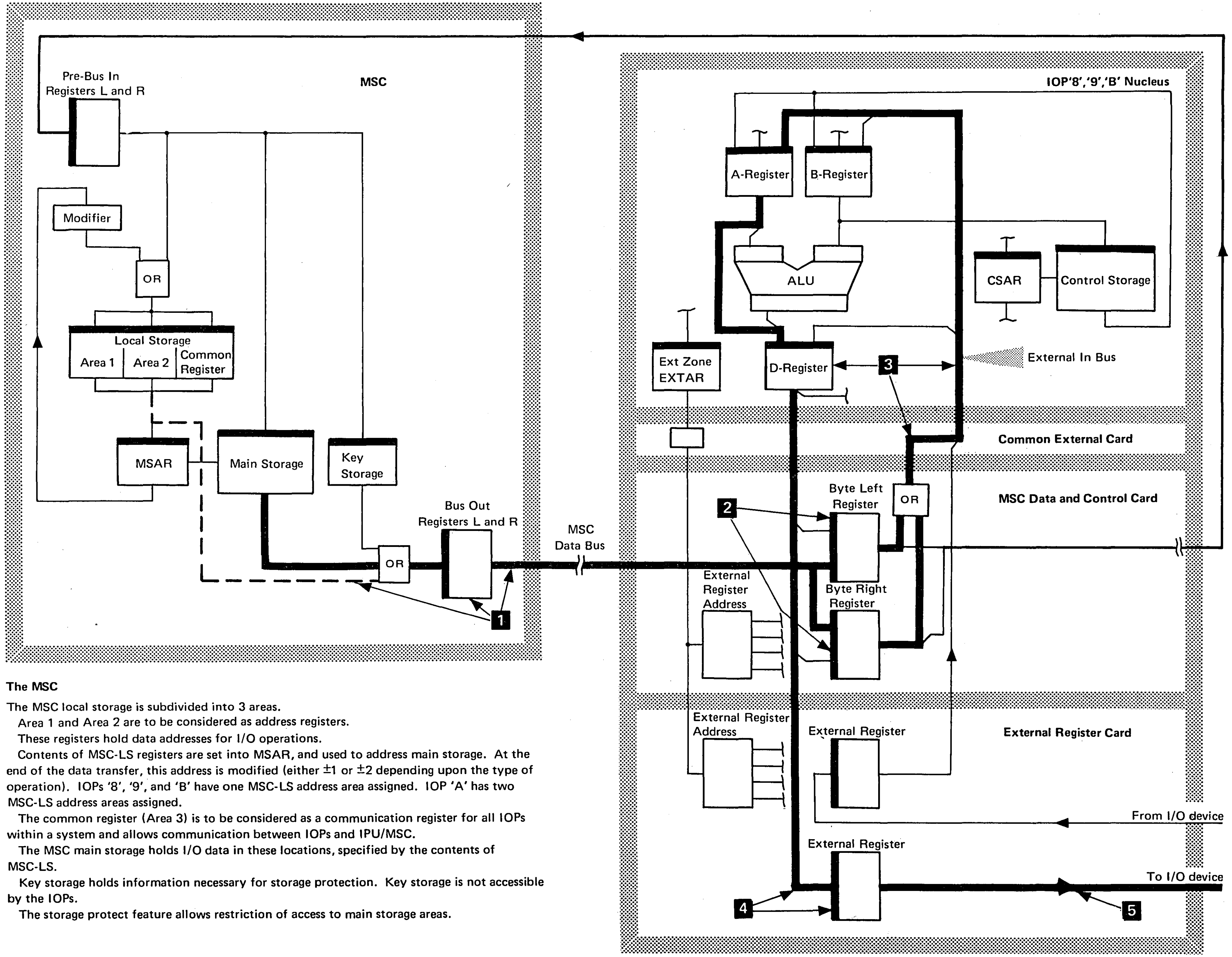
2 Sets the I/O data from the MSC data bus into the byte left and/or byte right registers.

3 Transfers contents of the byte left and/or byte right register via the external in bus and common external card into D-register on the IOP ALU card.

4 Transfers the contents of the D-register on the IOP ALU card into the external register (located on the external register card) that is addressed by the contents of the external zone register and EXTAR located on the IOP ALS/CSAR card.

5 From the external register, I/O data is then transferred to the I/O device.

See also flowchart on pages 2-022, 2-028, 2-060, 2-065.



The MSC

The MSC local storage is subdivided into 3 areas.

Area 1 and Area 2 are to be considered as address registers.

These registers hold data addresses for I/O operations.

Contents of MSC-LS registers are set into MSAR, and used to address main storage. At the end of the data transfer, this address is modified (either ±1 or ±2 depending upon the type of operation). IOPs '8', '9', and 'B' have one MSC-LS address area assigned. IOP 'A' has two MSC-LS address areas assigned.

The common register (Area 3) is to be considered as a communication register for all IOPs within a system and allows communication between IOPs and IPU/MSC.

The MSC main storage holds I/O data in these locations, specified by the contents of MSC-LS.

Key storage holds information necessary for storage protection. Key storage is not accessible by the IOPs.

The storage protect feature allows restriction of access to main storage areas.

▬▬ MSC main storage operation (data path for I/O data)

— — — The dotted line in MSC represents an MSC local storage operation, when data is transferred from MSC-LS to IOPs

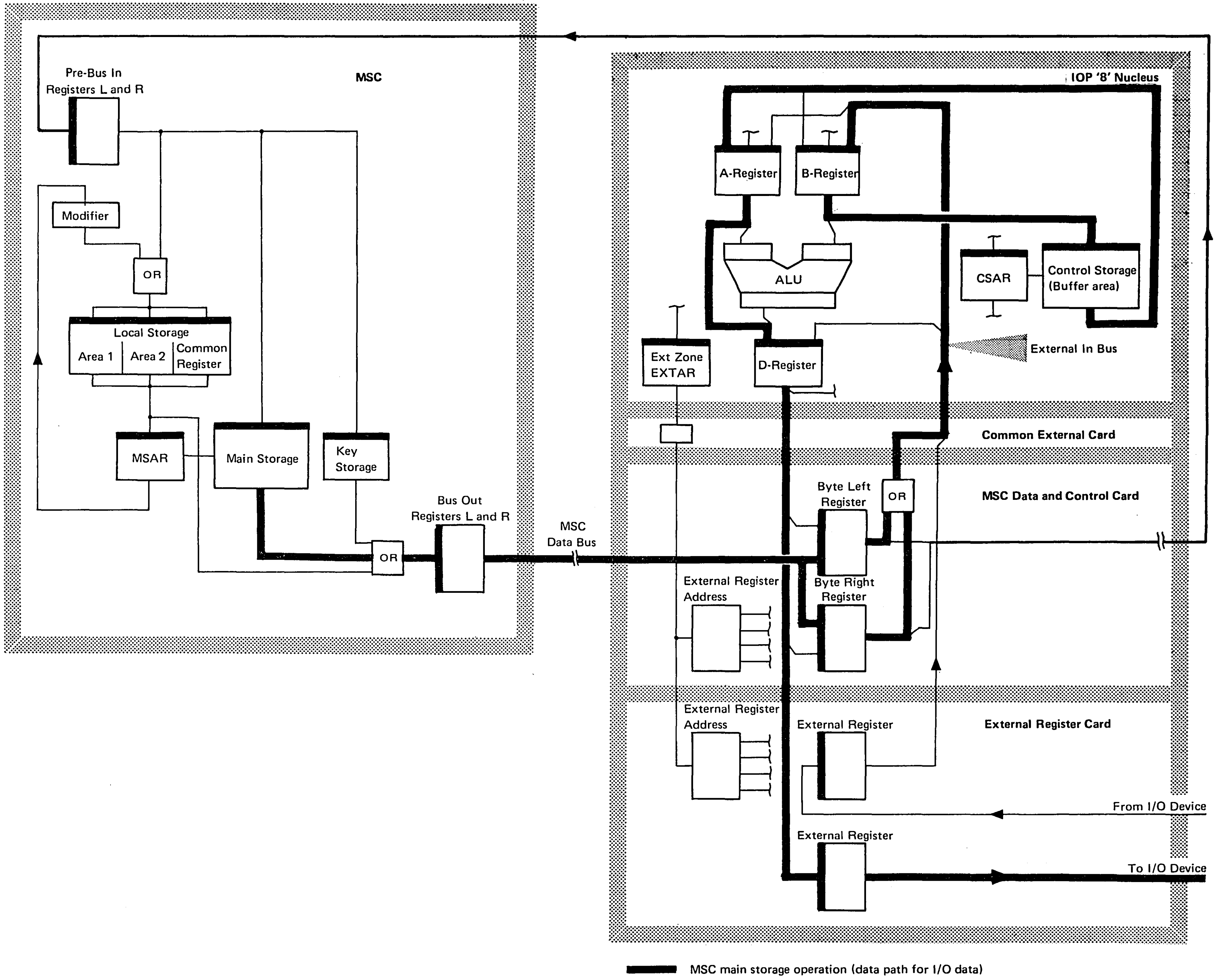
Data Transfer IOP '8' Write Operation

To those I/O devices that have a data buffer in IOP control storage

The data flow shows the data path for a write operation to those I/O devices that have a data buffer in the control storage of the controlling IOP.

- Data transfers from MSC to IOP are the same as for an unbuffered I/O device, except:
- Data fetched from MSC is set into the buffer of the addressed device (and not directly gated to the I/O device).
- When all bytes of the record are fetched from MSC and set into this buffer, then the data transfer from the buffer to the I/O device starts.

(See also flowchart on page 2-028).



IOP Read Operation (Data Transfer from I/O Device to MSC)

Data Transfer IOPs '8', '9', 'B' Read Operation

Those I/O devices that do not have a buffer in IOP control storage.

This data flow is additional to the flowchart on page 2-022 (data transfer) and shows the data path for a read operation.

A read operation (related to IOPs) means, inbound operation for the MSC under control of chain control lines and octopus control lines. (CCLs and OCLs are not shown here.)

For more details refer to pages:

- 2-100 MSC Local Storage Operations
- 2-110 MSC Main Storage Operations
- 2-120 Description of CCLs
- 2-130 Description of OCLs.

A read operation consists of 5 main steps:

1 I/O data from the I/O device is set into the external register (on the external register card) that is addressed by the contents of the external zone register and EXTAR, located on the IOP ALS/CSAR card.

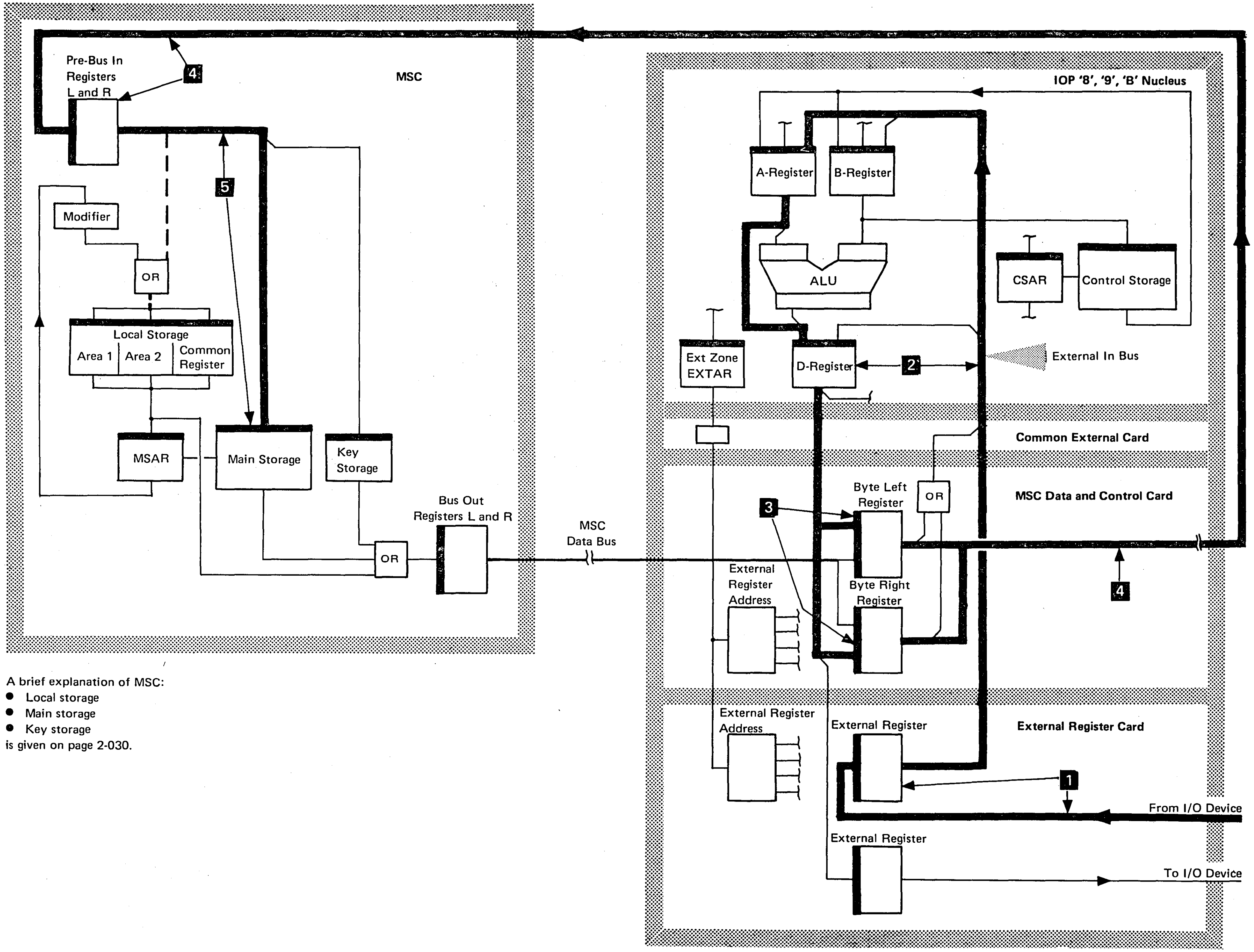
2 Contents of the external register are gated via the external in bus and the common external card into the D-register on the IOP ALU card.

3 From the D-register on IOP ALU card I/O data is transferred into either the byte left or byte right register.

4 Contents of the byte left and byte right registers are gated via the MSC data bus into the MSC pre-bus in register.

5 Contents of the MSC pre-bus in register are set into the addressed main storage location.

See also flowchart on pages 2-022, 2-028, 2-060, 2-068.



A brief explanation of MSC:

- Local storage
- Main storage
- Key storage

is given on page 2-030.

▬ MSC main storage operation (data path for I/O data)

– – – The dotted line in MSC represents an MSC local storage operation, when data is transferred from IOPs into MSC-LS (data path for data addresses etc.)

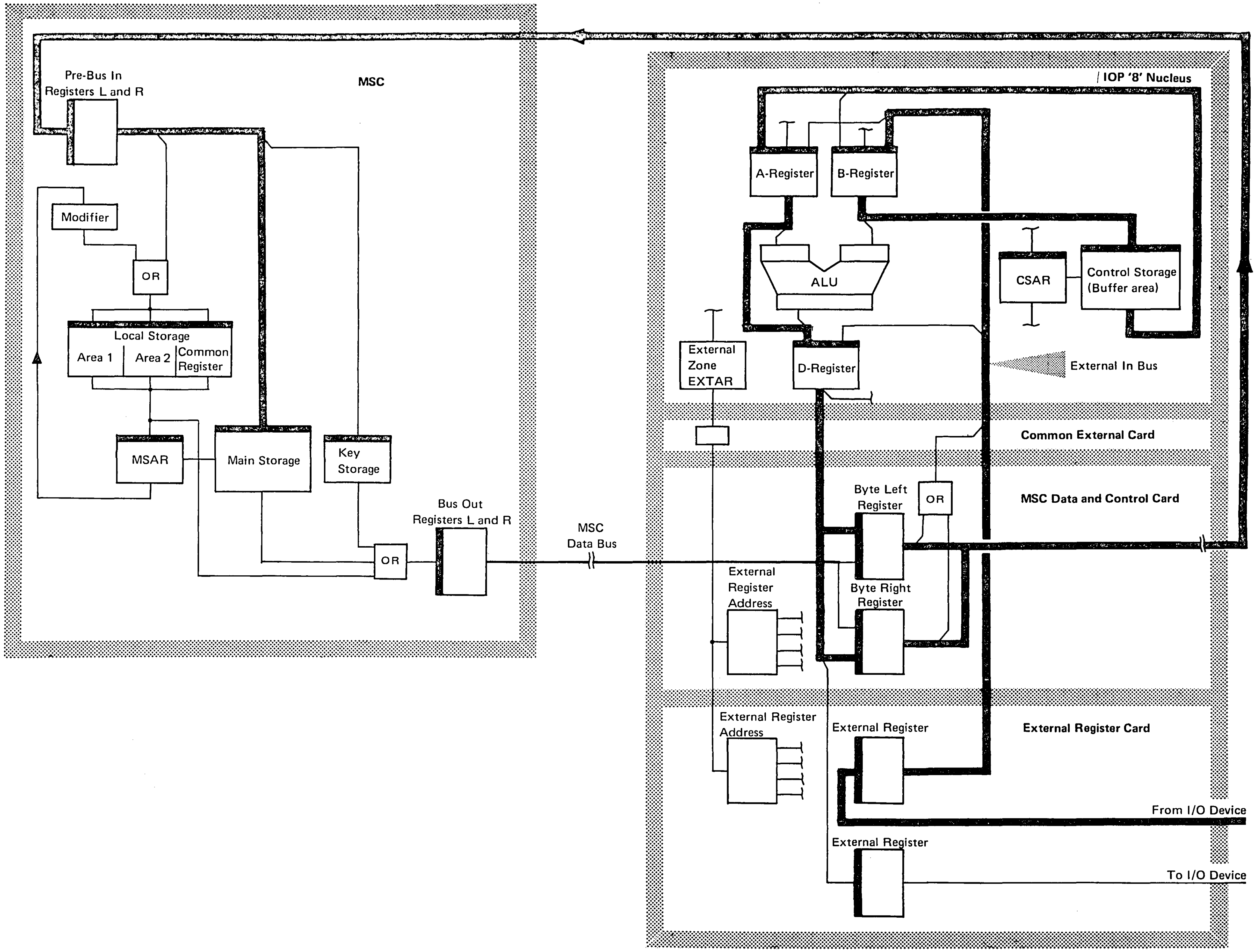
Data Transfer IOP '8' Read Operation

From those I/O devices that have a data buffer in IOP control storage.

The data flow shows the data path for a read operation to those I/O devices that have a data buffer in the control storage of the controlling IOP.

- Data from the I/O device is set into the buffer of the addressed I/O device (and not directly gated to the MSC).
- When all bytes of the record are read and set into this buffer, the data transfer from IOP to MSC starts.
- Data transfers from IOP to MSC are the same as for an unbuffered I/O device.

(See also flowchart on page 2-028).



▬ MSC main storage operation (data path for I/O data)

IOP Write Operation

Data Transfer IOP 'A' Write Operation

The data flow is additional to the flowchart on page 2-024 and shows the data path for:

- I/O data (solid line), and
- Control information.

A write operation (related to IOPs) means, outbound operation for the MSC under control of chain control lines and octopus control lines. (CCLs and OCLs are not shown here.)

For more details refer to pages:

2-100 MSC Local Storage Operations
2-110 MSC Main Storage Operations
2-120 Description of CCLs
2-130 Description of OCLs.

A write operation consists of 4 main steps:

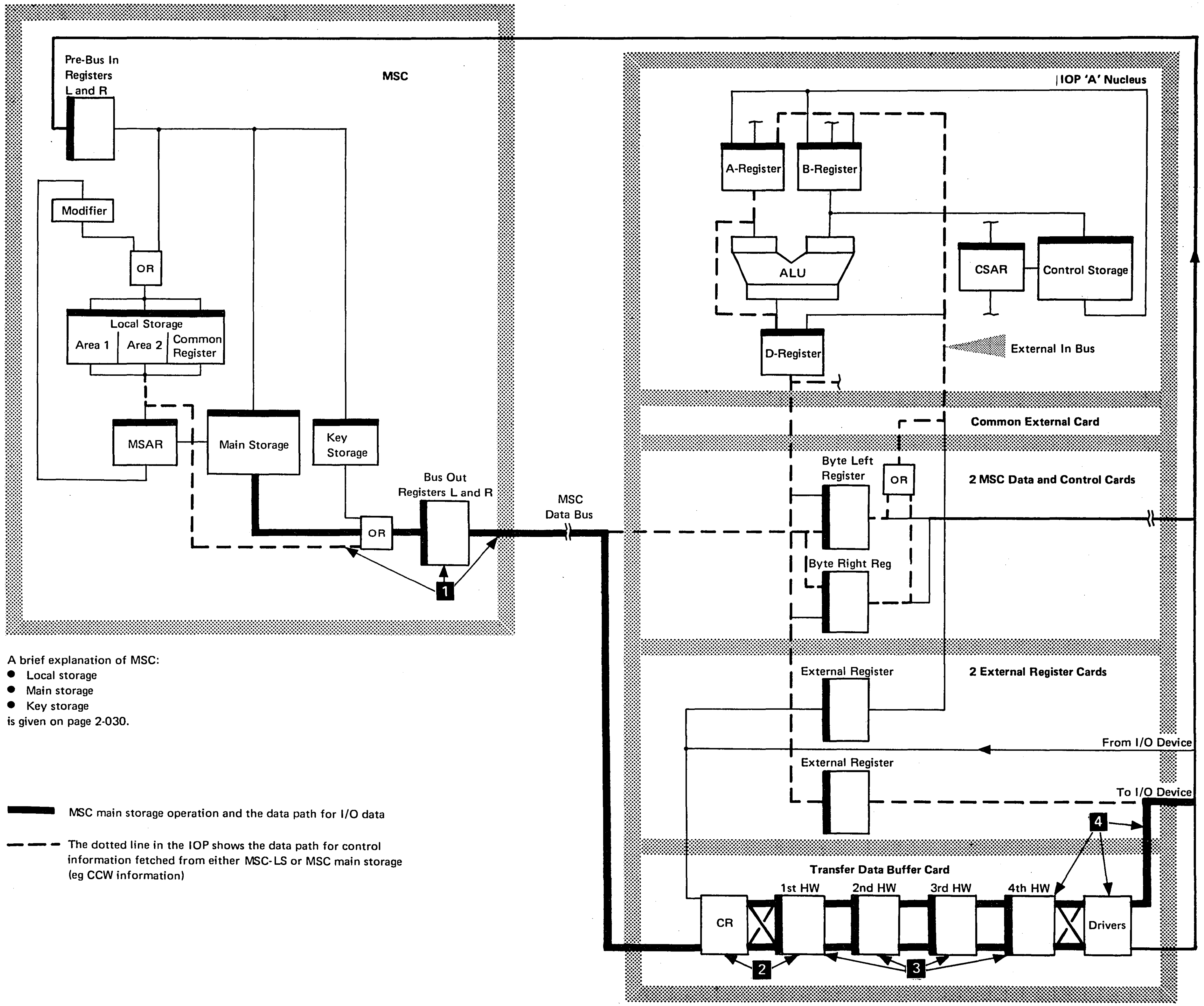
1 Sets I/O data from the addressed main storage location via the bus out registers onto the MSC data bus.

2 Sets I/O data from the MSC data bus into the transfer data buffer 1st location.

3 Shifts I/O data through the transfer data buffer.

4 I/O data is gated from 4th transfer data buffer location, to the addressed I/O device.

For more details refer to *IBM 3125 Processing Unit, Direct Disk Attachment, MLM,* SY33-1073.



A brief explanation of MSC:

- Local storage
- Main storage
- Key storage

is given on page 2-030.

▬ MSC main storage operation and the data path for I/O data

– – – The dotted line in the IOP shows the data path for control information fetched from either MSC-LS or MSC main storage (eg CCW information)

IOP Read Operation

Data Transfer IOP 'A' Read Operation

The data flow is additional to the flowchart on page 2-024 and shows the data path for:

- I/O data (solid line), and
- Control information.

A read operation (related to IOPs) means, inbound operation for the MSC under control of chain control lines and octopus control lines. (CCLs and OCLs are not shown here.)

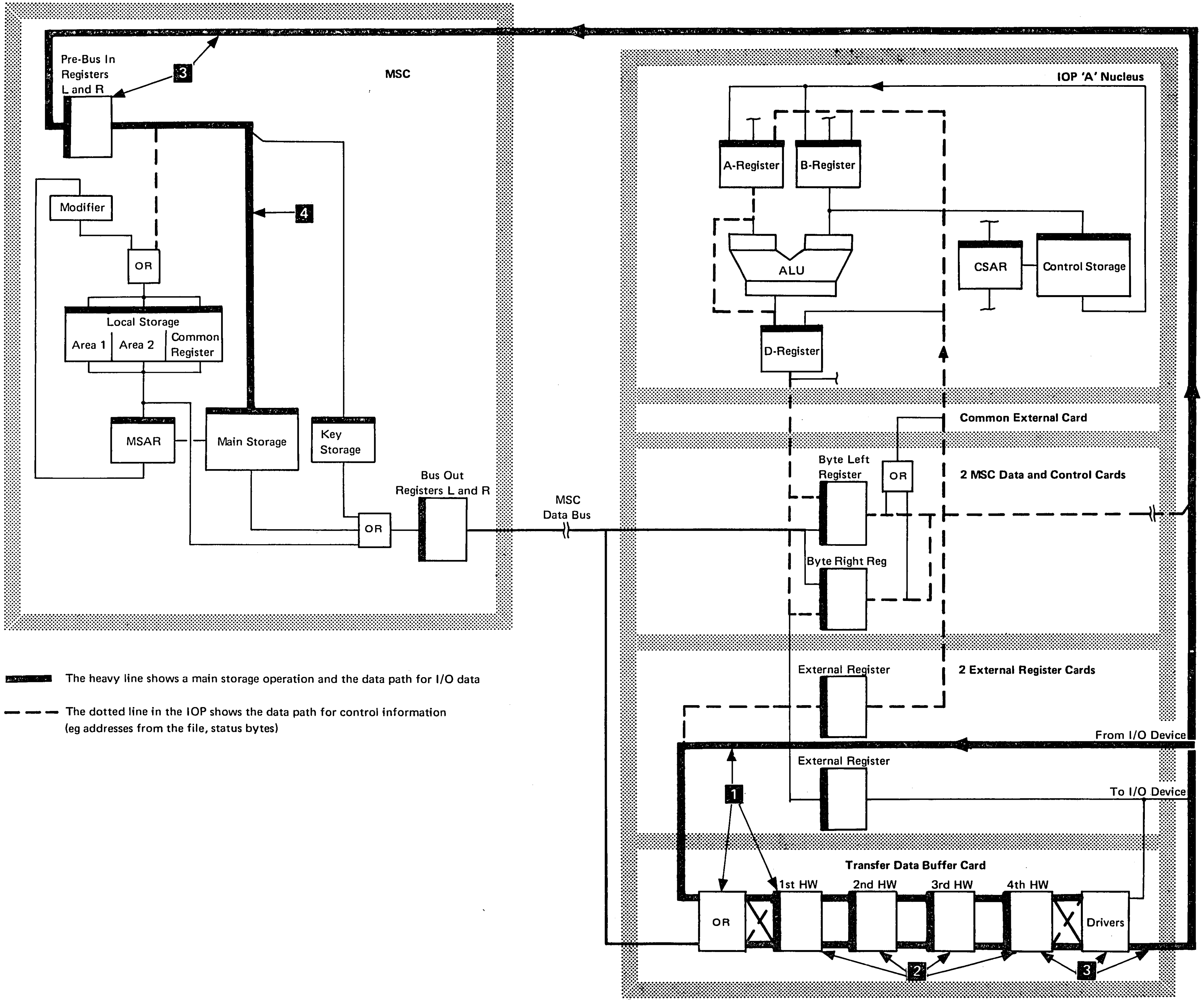
For more details refer to pages:

2-100 MSC Local Storage Operations
2-110 MSC Main Storage Operations
2-120 Description of CCLs
2-130| Descriptions of OCLs.

A read operation consists of 4 main steps:

1. Transfers I/O data from the I/O device to the 1st position of the transfer data buffer.
2. Shifts I/O data through the transfer data buffer.
3. Sets I/O data from the last position of the transfer data buffer via the MSC data bus into the MSC pre-bus in register.
4. Sets I/O data from the MSC pre-bus in register into the addressed MSC main storage position.

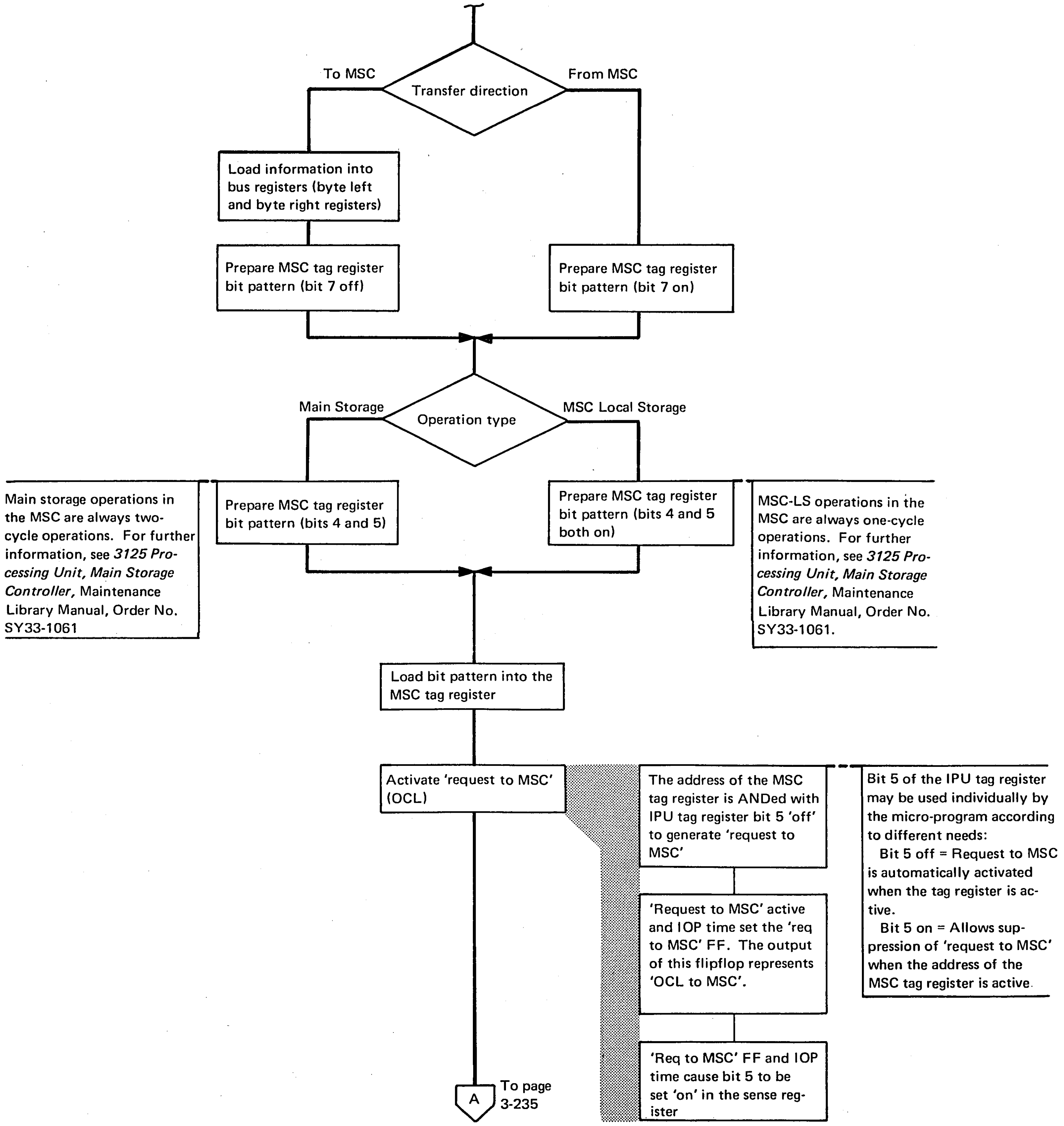
For more information refer to *IBM 3125 Processing Unit, Direct Disk Attachment,* MLM, SY33-1073.



The heavy line shows a main storage operation and the data path for I/O data

The dotted line in the IOP shows the data path for control information (eg addresses from the file, status bytes)

Interaction Between IOP and MSC



Main storage operations in the MSC are always two-cycle operations. For further information, see *3125 Processing Unit, Main Storage Controller,* Maintenance Library Manual, Order No. SY33-1061

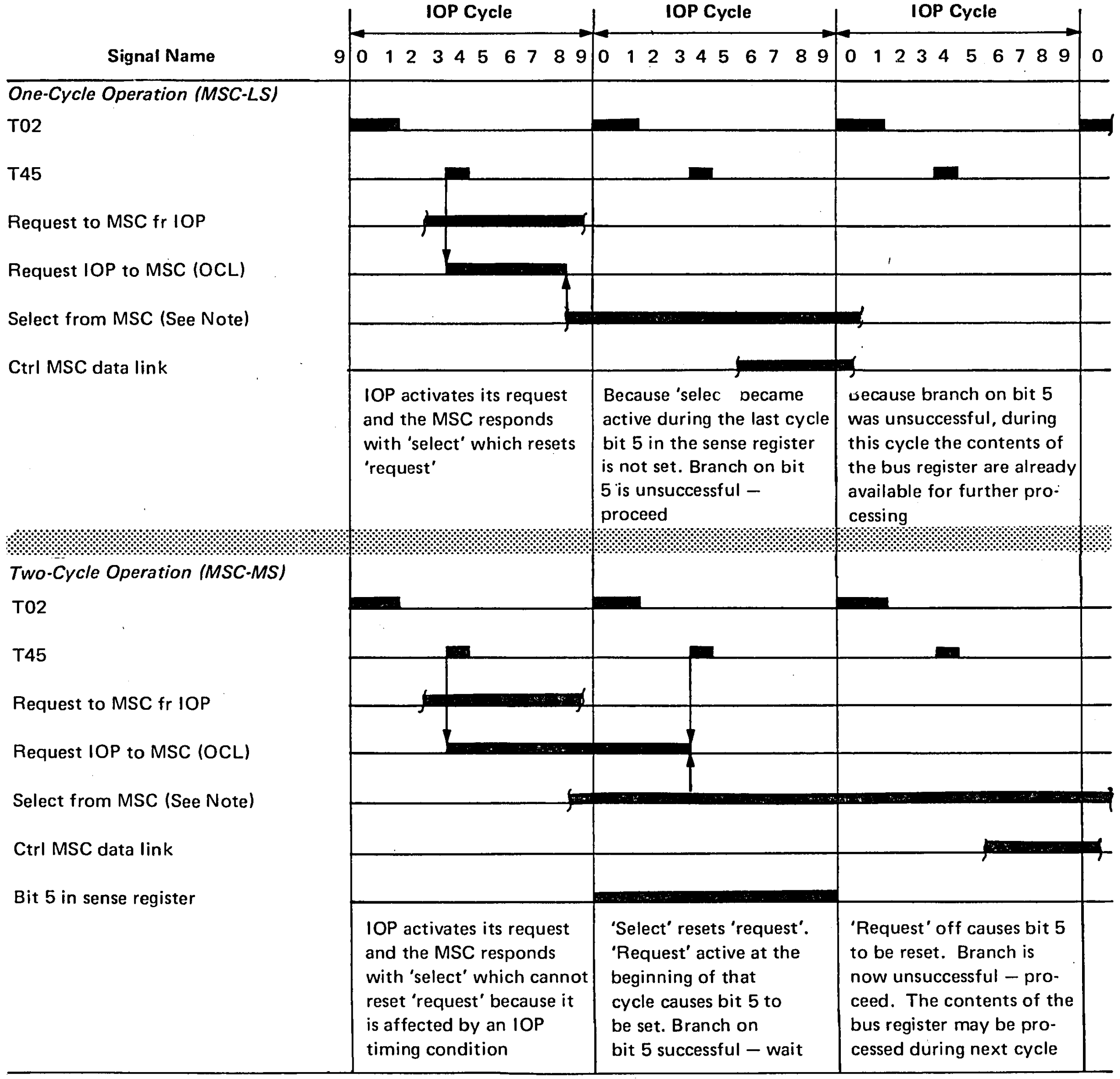
MSC-LS operations in the MSC are always one-cycle operations. For further information, see *3125 Processing Unit, Main Storage Controller,* Maintenance Library Manual, Order No. SY33-1061.

Bit 5 of the IPU tag register may be used individually by the micro-program according to different needs:
Bit 5 off = Request to MSC is automatically activated when the tag register is active.
Bit 5 on = Allows suppression of 'request to MSC' when the address of the MSC tag register is active.

Timing

The two diagrams below show the timing relationship (in principle) under the following conditions:

- IPU tag bit 3 off.
- Select from MSC active shortly after activation of 'request to MSC'.



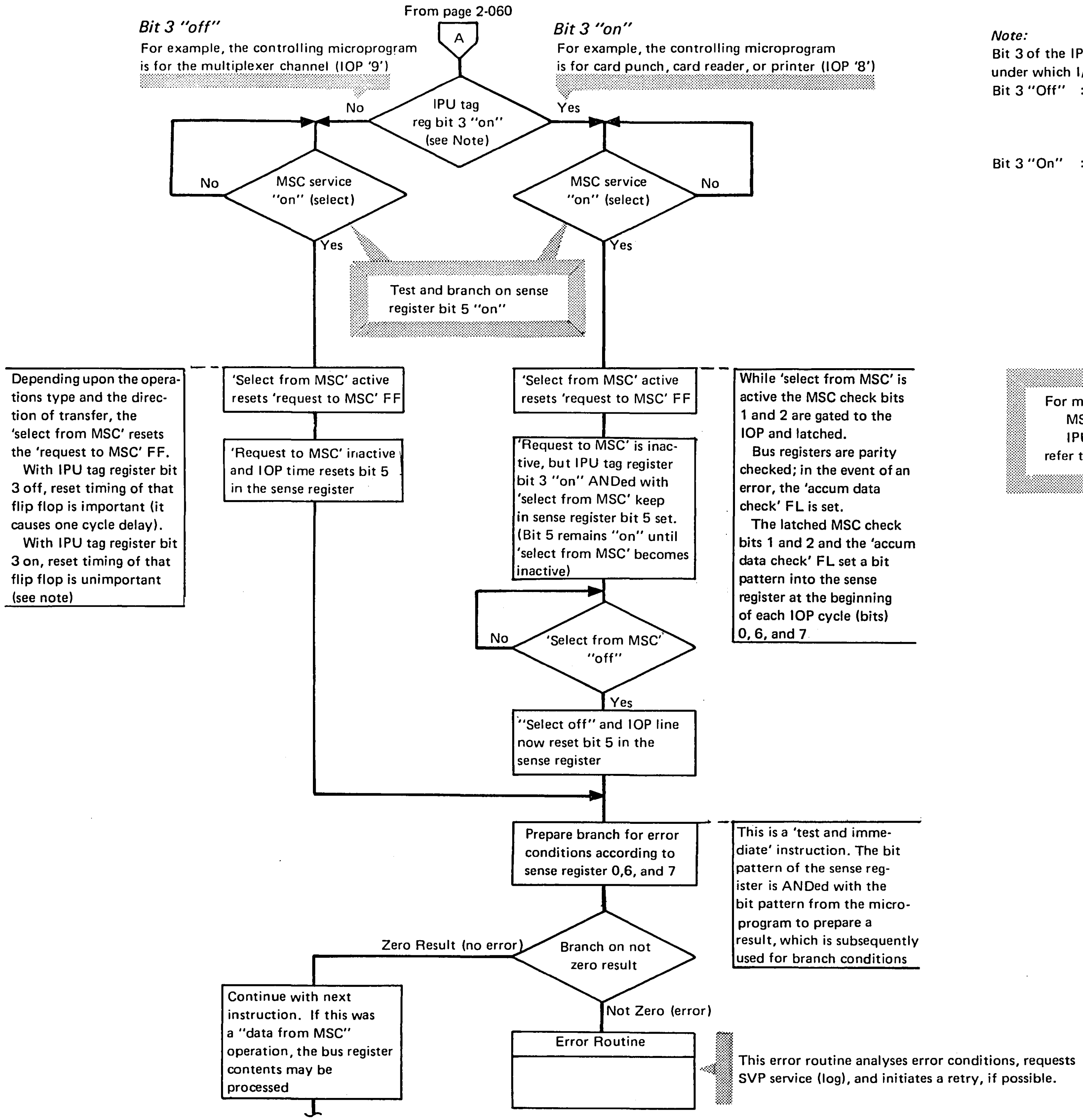
One-Cycle Operation (MSC-LS)

IOP activates its request and the MSC responds with 'select' which resets 'request'	Because 'select' became active during the last cycle bit 5 in the sense register is not set. Branch on bit 5 is unsuccessful — proceed	Because branch on bit 5 was unsuccessful, during this cycle the contents of the bus register are already available for further processing

Two-Cycle Operation (MSC-MS)

IOP activates its request and the MSC responds with 'select' which cannot reset 'request' because it is affected by an IOP timing condition	'Select' resets 'request'. 'Request' active at the beginning of that cycle causes bit 5 to be set. Branch on bit 5 successful — wait	'Request' off causes bit 5 to be reset. Branch is now unsuccessful — proceed. The contents of the bus register may be processed during next cycle

Note: If 'select from MSC' becomes active later, or if IPU tag bit 3 is on, a number of consecutive branch operations are performed (on sense register bit 5) for as long as bit 5 is 'off'.



Bit 3 "off"

For example, the controlling microprogram is for the multiplexer channel (IOP '9')

Bit 3 "on"

For example, the controlling microprogram is for card punch, card reader, or printer (IOP '8')

Depending upon the operations type and the direction of transfer, the 'select from MSC' resets the 'request to MSC' FF.
With IPU tag register bit 3 off, reset timing of that flip flop is important (it causes one cycle delay).
With IPU tag register bit 3 on, reset timing of that flip flop is unimportant (see note)

While 'select from MSC' is active the MSC check bits 1 and 2 are gated to the IOP and latched.
Bus registers are parity checked; in the event of an error, the 'accum data check' FL is set.
The latched MSC check bits 1 and 2 and the 'accum data check' FL set a bit pattern into the sense register at the beginning of each IOP cycle (bits) 0, 6, and 7

This is a 'test and immediate' instruction. The bit pattern of the sense register is ANDed with the bit pattern from the microprogram to prepare a result, which is subsequently used for branch conditions

This error routine analyses error conditions, requests SVP service (log), and initiates a retry, if possible.

Note:
Bit 3 of the IPU tag register may be individually used by the microprogram according to conditions under which I/O devices are served.

Bit 3 "Off" :	The beginning of 'select from MSC' is used to continue with the microprogram. It is important to control bit 5 in the sense register to ensure "good" information in the bus registers. If bit 5 in the sense register is set "off" too early, the bus register contents might not be "good" (caused by the bus delay).
Bit 3 "On" :	The end of 'select from MSC' is used to continue with the microprogram. The reset of bit 5 in the sense register is unimportant because at the end of 'select from MSC' the information in the bus registers is always "good".

For more details about
MSC tag register
IPU tag register
refer to Page 4-068

External Register Assignment and Layout — MSC Data and Control Cards

- Addresses '18' to '1F' are located on the MSC data and control card
- Address '1B' does not define a register, but activates a line, which is used to reset the 'accumulated data check' FL.
- Addresses '1C' to '1F' twice define the two bus registers. Once *without* "request to MSC" (used if one or both registers are to be loaded only) and once *with* "request to MSC" (used if one or both registers are to be loaded and transferred to the MSC).

Register Address	Register Designation	Bit Positions 0	1	2	3	4	5	6	7
00 to 17	CH EXT 0 to 23	The number, the use, and the contents of these registers depend upon device requirements. These registers are located on external register cards within the different Front Ends.							
18	CH EXT 24	Forced to 1	MSC Tag Register with Request to MSC Byte Left Select	Register 1 Selection	Use Common Register Selection	Increment	Decrement	Halfword	Data from MSC
19	CH EXT 25	Forced to 0	IPU Tag Register Response to IPU	Metering In	Slow Interface	IOP Busy	Suppress Request	MSC Bus Log (IOP '8' only)	Interrupt Request
1A	CH EXT 26	Sense Register Accumulated Data Check	Select IOP for IPU	Halt I/O or Device	System Reset	Page Boundary	Request to MSC	MSC Check Bit 1	MSC Check Bit 2
1B	CH EXT 27	Reset Accumulated Data Check							
1C	CH EXT 28	Byte Left Register (of MSC data bus)							
1D	CH EXT 29	Byte Left Register with Request to MSC							
1E	CH EXT 30	Byte Right Register (of MSC data bus)							
1F	CH EXT 31	Byte Right Register with Request to MSC							

MSC tag register and IPU tag register, see Page 4-068. External register addressing, see also Pages 4-080, 4-096

This page is intentionally left blank

IOP and MSC Operations

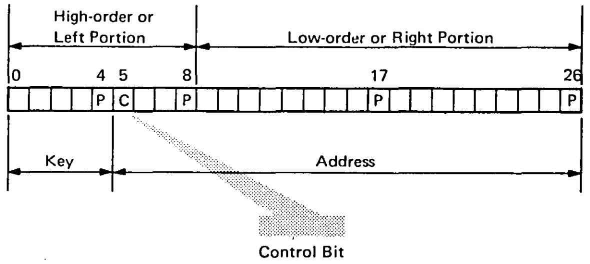
Local Storage Operations

- All IOPs communicate with the MSC in the same manner. Two directions for the data transfer are possible:
  Store operations – Inbound to the MSC
  Fetch operations – Outbound from the MSC.
- Store and fetch operations can be performed between:
  IOP and MSC-LS (shown on this page)
  IOP and MSC-MS (shown on facing page).
- Status shown in the table below is related to the CCLs and OCLs *not* to the bits in MSC tag register.

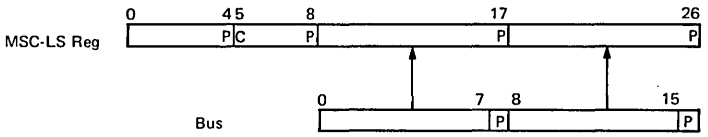
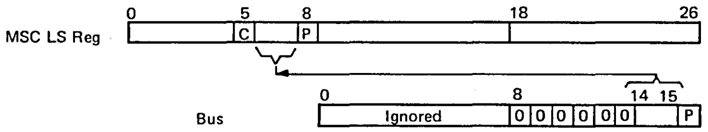
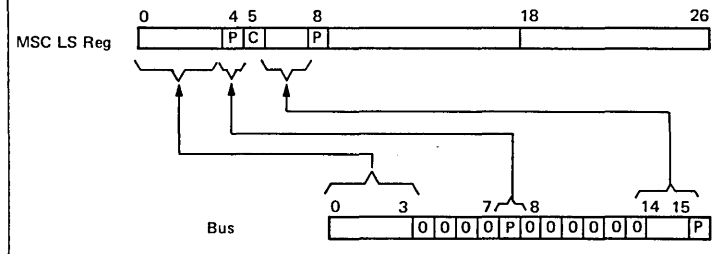
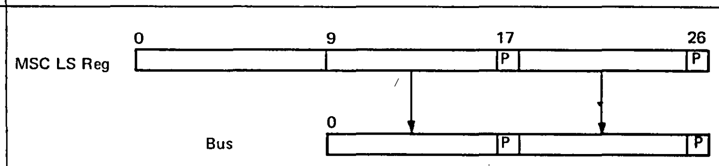
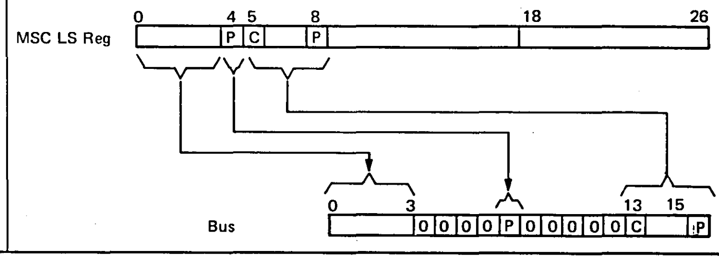
Fetching and Storing Possibilities

- The table below illustrates all the possibilities for storing data into, and fetching data from, the MSC-LS.

MSC-LS Layout



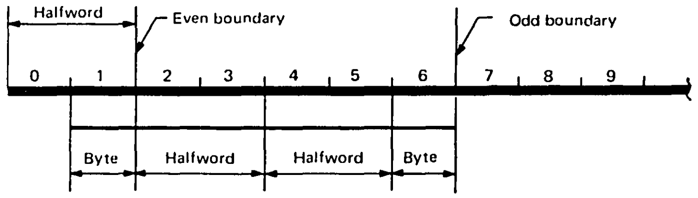
Note: The three signals marked with an asterisk* determine into what register of local storage the bus bits are stored. The following combinations are possible:
Register 0 in area 1
Register 1 in area 1
Register 0 in area 2
Register 1 in area 2.
If the 'common reg' line is active, the automatic selection of the assigned area (register 0 or 1 in area 1 or 2) is inhibited. For layout of MSC-LS Common Register, see Page 2-150.

	Signal Name										Examples (The upper part represents MSC-LS and the lower part represents MSC-Data Bus)	Comments
	'Reg' FL	MSC Tag Register								From Front End		
		Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 4 & 5	Bit 6	Bit 7			
Operation	Request IOP to MSC	Byte Left Sel	Reg 1 Sel* (See Note)	Use Com Reg Sel* (See Note)	Increment	Decrement	Main Store Sel	Halfword	Data from MSC	Use Area 2* (See Note)		
1. Store low-order part of address	Active	Inactive (right)	Active (reg 1)	Either inactive (assigned area) or active (Common Register)	Ignored for address modification, but used to define MSC-LS (both bits in TAG register active, cause inactivation of OCL main store select)		Inactive (LS)	Ignored	Inactive (to MSC)	Either active (area 2) or inactive (area 1)		One halfword (left and right byte of data bus) is transferred from the IOP into the low-order portion of a specified register
2. Store high-order part of address	Active	Active (left)	Active (reg 1)				Inactive (LS)	Inactive (byte)	Inactive (to MSC)			Bus bits 8 to 13 are to be zero, if not, LS register bit 5 (check bit) is set. LS register parity (bit 8) is generated.
3. Store high-order part of address and store key	Active	Active (left)	Active (reg 1)				Inactive (LS)	Active (halfword)	Inactive (to MSC)			Bus bits 4 to 7 must be zero to ensure correct parity for the key
4. Fetch low-order part of address	Active	Inactive (right)	Active (reg 1)				Inactive (LS)	Inactive (byte)	Active (from MSC)			The line 'halfword' is set to 'byte' even though a halfword is requested and transferred
5. Fetch high-order part of address and fetch key	Active	Active (left)	Active (reg 1)				Inactive (LS)	Inactive (byte)	Active (from MSC)			Bus bits 4 to 7 and 8 to 12 are filled with zeros, which are generated by the MSC (see comment of operation No. 4)

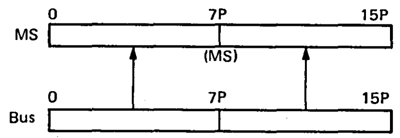
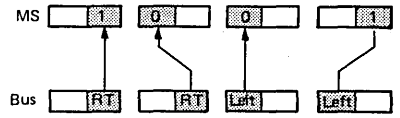
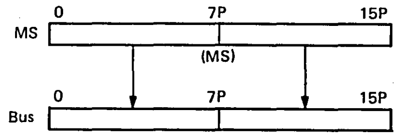
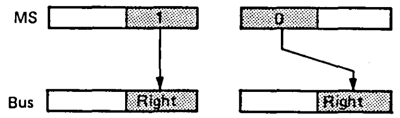
Main Storage Operations

Fetching and Storing Possibilities

- The table below illustrates all the possibilities for storing data into, and fetching data from, the MSC-MS.
- Bytes may be stored on either an even or an odd boundary.
- Halfwords may be stored on an even boundary only.
- If it is necessary to store data on an odd boundary when working in halfword mode, the transfer must be started with a byte op. The transfer may then continue with halfwords. The transfer ends with the transfer of a byte; see illustration.
- Every IOP may store in either ascending or descending address sequence (that is, forward or backward) or may specify no address modification.
- All main storage operations that are initiated by an IOP require two cycles because the MSC performs an automatic read operation prior to the actual access. This read operation is necessary for key comparison.
- Store operations are suppressed when:
  - Key does not match
  - Address is invalid
  - Hardware errors detected.
- For detailed information on MSC-MS refer to *3125 Processing Unit, Main Storage Controller,* Maintenance Library Manual, Order No. SY33-1061.
- Status shown in the table below is related to CCLs and OCLs, not to the bits in MSC tag register.



Note: 'Increment' and 'decrement' signals are set as desired, but both must not be active at the same time. For the same operation, both signals may be inactive to specify no modification, see also page 4-064.

	Signal Name										Examples	Comments
	'Req' FL	MSC Tag Register								From Front End		
		Bit 1	Bit 2	Bit 3	Bits 4 & 5	Bit 4	Bit 5	Bit 6	Bit 1			
Operation	Request IOP to MSC	Byte Left Sel	Reg 1 Sel	Use Com Reg Sel	Main Store Sel	Increment (see note)	Decrement (see note)	Halfword	Data from MSC	Use Area 2 Sel		
1. Store halfword	Active	Ignored	–	–	Active (MS)			Active (halfword)	Inactive (to MSC)	–		Data is transferred via the MSC data bus and is set into the MSC pre-bus in register. In the event of an error, data may be switched through to the SVP
2. Store byte	Active	Active (left) Inactive (right)	–		Active (MS)			Inactive (byte)	Inactive (to MSC)	–		The IOP may offer a byte on either the right- or left-half of the data bus, as indicated by the line 'byte left sel'. The byte is stored into either the left- or the right-half of the MS halfword, depending on the low-order bit of the address. Bit on (1): Offered byte enters right side of MS halfword. Bit on (0): Offered byte enters left side of MS halfword.
3. Fetch halfword	Active	Ignored	–	–	Active (MS)			Active (halfword)	Active (from MSC)	–		If an error occurs, the MSC does not suppress outbound data. The subprocessor must check the received data and initiate corrective action
4. Fetch byte	Active	Ignored	–	–	Active (MS)			Inactive (byte)	Active (from MSC)	–		The byte is taken from MS depending on the low-order bit of the address (see comments for operation No. 2). The fetched byte always appears on the right side of the data bus. The left side of the bus is ignored by the subprocessor

Chain Control Lines

Line Name	IOP Busy	IOP Not Operational	Response to IPU	Metering In	Decrement	Increment	Halfword	Data from MSC
Direction	To IPU	To IPU	To IPU	To IPU	To MSC	To MSC	To MSC	To MSC
	Shared	Shared	Shared		Shared	Shared	Shared	Shared
Line Level	Up = Active	Up = Active	Up = Active	Up = Active	Down = Active	Down = Active	Down = Halfword; Up = Byte	Down = Active
Explanation	The IPU tag register bit 4 is set as a result of a "busy" indication. This tag register bit, in conjunction with 'select IOP FROM IPU', activates the 'busy' line.	This line becomes active: 1. If an IOP is not installed. 2. As a result of an IOP error 'IOP halt' is connected to the SVP link card and is signaled to the SVP. Simultaneously, the 'prevent I/O' FL is set. 'Prevent I/O', in conjunction with 'select', activates the 'IOP not operational' line.	This line is activated by the selected IOP. In the active condition it indicates the following: • *Beginning of an operation:* channel and device address valid, proceed. • *End of an operation:* channel and device address stored in MSC-LS common Register. In the inactive condition it indicates the following: • *Beginning of an operation:* Condition code stored. • *End of an operation:* CSW stored.	IPU tag register bit 2 and (Not) 'prevent I/O' activate the 'metering in' line, which signals to the IPU that an I/O operation is in process.	This line when active specifies the manner in which the address that is stored in the MSC-LS (assigned area) has to be updated. Both lines must be specified, but must not be active at the same time. Both lines inactive simultaneously specify *not modification.* (See also Page 4-064).	When this line is active, an IOP specifies the manner in which the address that is stored in the MSC-LS (assigned area) has to be updated.	With this line an IOP specifies the data format that is to be transferred.	With this line an IOP specifies the direction of a transfer. Related to the MSC, active means: data from MSC, or outbound, or fetch. Related to the MSC, inactive means: data to MSC, or inbound, or store.

Line Name	Halt I/O or Device	Metering Out	Clock Out	Lamp Test	System Reset	Power On Reset (POR)	Page Boundary	Parity of MSC Control Lines
Direction	From IPU	From IPU	From IPU	From SVP	From SVP	From SVP	From MSC	To MSC
		Shared					Shared	
Line Level	Down = Active	Up = Active	Down = Active	Down = Active	Down = Active	Down = Active	Down = Active	Up = Active
Explanation	This line when active signals that an I/O op has to be terminated. The active line causes a bit to be set into the sense register that is located on the MSC data and control card. The contents of the sense register are gated to the IOP nucleus, where the bit is detected by the micro-program. Termination is electronic but the mechanical operation of the device runs to its normal end.	This line is activated by the IPU at the moment when the process meters start running.	This line when active indicates that the IPU is running and that no exceptional or unusual condition exists and prevents the changing of enable state to disable state (and vice versa) in control units.	This line is activated by operating a switch on the operator's panel. With this line active, all lamps are turned on for test purposes.	This line is activated by the manual operation "System Reset". With this line active, a bit is set into the sense register that is on the MSC data and control card. The contents of the sense register are gated to the IOP necleus, where the bit is detected by the microprogram, which selects the SYSRES routine.	This line is generated when power is switched on and causes generation of the 'clock reset' signal, which resets and sets clock ring latches and triggers to a defined status. (see Op Decode and Run Control card on Page 4-120).	This line is activated from the MSC and indicates that the next halfword or the next byte will be read from, or written into, a new page (area) of main storage. If this new page is the next adjacent one, this signal may be ignored. If this new page is not the next adjacent one the IOP has to make available, in sufficient time, the new data address. The 'page boundary' signal is activated during the last fetch or store operation before the page boundary is actually crossed.	This line is generated from MSC tag register bits 4, 5, and P in conjunction with the 'page boundary' line. Parity of the MSC control lines represents the parity of the chain control lines and is used in the MSC for check purposes (see Page 4-064). IOPs that serve high-speed I/O devices use two additional lines ('MSC tag parity corr' and 'IOP use are 2') for this parity generation.

Octopus Control Lines

Line Name	**Interrupt Request IOP**	**Request IOP to MSC**	**Main Store Sel**	**Reg 1 Sel**	**Byte Left Sel**	**Use Com Reg Sel**	**Use Area 2 Sel**
Direction	**To IPU**	**To MSC**	**To MSC**	**To MSC**	**To MSC**	**To MSC**	**To MSC**
(See Note)	**Non Shared**	**Non Shared**	**Shared**	**Shared**	**Shared**	**Shared**	**Shared**
Line Level	**Up = Active**	**Up = Active**	**Up = To MS** **Down = To LS**	**Up = To Reg 1** **Down = To Reg 0**	**Up = Byte Left** **Down = Byte Right**	**Up = Active**	**Up = Area 2** **Down = Area 1**
Explanation	An IOP activates this line under the following conditions: (a) Data transfer between the IOP and the MSC is completed (CE). (b) Data transfer between the IOP and device is completed and the device has completed its mechanical runout (DE). If interrupts of that IOP are not masked off and if none is waiting with a higher priority, the IPU responds with 'select'	Each IOP has its own request line, which identifies the requesting IOP and its MSC-LS area. Request lines are connected to priority networks and must remain active until the MSC responds with 'select'	This line specifies access to either MS or LS and must specify the destination until 'select' becomes inactive	This line specifies whether register 0 or register 1 of the assigned area in the MSC-LS is to be used	On main storage operations, the IOP indicates on what part of the MSC data bus the byte is delivered. On local storage operations, the IOP specifies into what part of the designed register delivered data is to be set	With this line active an IOP specifies that delivered data is to be set into the MSC-LS common register instead of register 0 or register 1 of the area that is assigned to the IOP. The automatic selection of the assigned area is inhibited.	This line is used only on those IOPs that service devices with high data transfer rates, for example, direct disk attachment. The assigned area of these IOPs consists of four MSC-LS registers to ensure that the necessary addresses are available in sufficient time

Line Name	**Select IOP from IPU**	**Control Data Link**	**MSC Check Bit 1**	**MSC Check Bit 2**	**Select IOP from MSC**
Direction	**From IPU**	**From MSC**	**From MSC**	**From MSC**	**From MSC**
(See Note)	**Non Shared**		**Shared**	**Shared**	**Non Shared**
Line Level	**Up = Active**	**Down = Active**	**(See text below)**	**(See text below)**	**Up = Active**
Explanation	This line signals to the IOP that the IPU expects action from the IOP. When initiating an I/O operation, the IOP fetches the contents of the MSC-LS common register. With interrupt requests the selected IOP sets the channel and device address into the MSC-LS common register	This line is ANDed with 'sel IOP from MSC' and then named 'MSC data good'. The line 'MSC data good' in conjunction with 'MSC tag reg bit 7' is used to generate 'byte left reg' and 'byte right reg clk' for outbound operations.	The combination of these two lines represents coded information that is determined for the IOP which started the current operation **Bit 1** **Bit 2** Up Up : No checks Up Down : Key mismatch and not invalid address Down Up : Invalid address Down Down : Any MSC circuit check		Activation of this line informs the requesting IOP that its request has been accepted. The line indicates for what duration data and control lines have to be kept valid

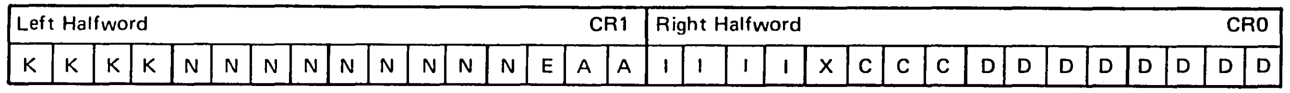
Note: Octopus control lines may be shared or non shared.

Shared lines are common for two IOPs. To prevent confusion (if two IOPs specify a shared line differently) a shared line is considered to be active by the MSC only as long as 'select' is active. (Because only one IOP may be selected at one time, only one IOP defines the levels of these control lines.)

Non-shared lines are individual IOP lines, each IOP having its own line.

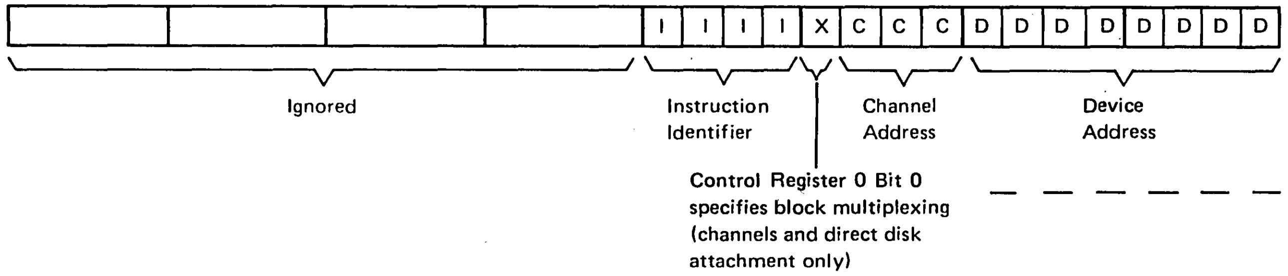
Supplementary Information

MSC-LS Common Register Layout

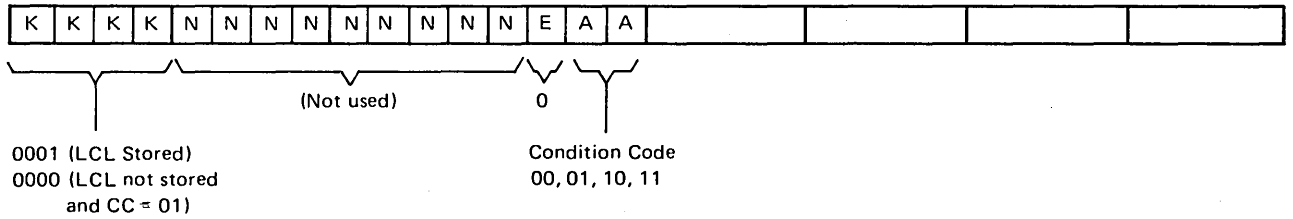


MSC-LS Common Register Contents

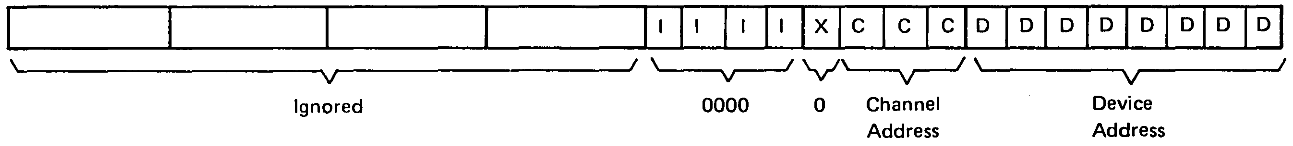
IPU to IOP for Initiation of XIO



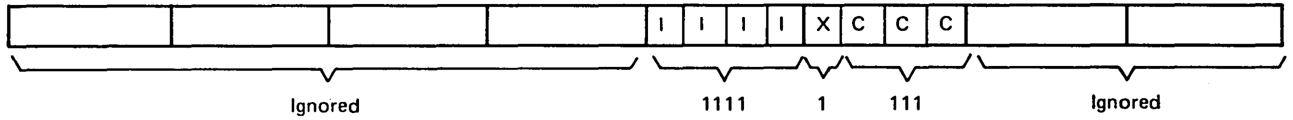
IOP to IPU after Initial Selection of Device



IOP to IPU for Normal Interrupt Request



IOP to IPU to Ignore Interrupt Request



Note: The instruction identifiers are listed on Page 1-030.

Block Multiplexing Control

When bit 0 of Control Register 0 is set, disk files operate in block multiplexing mode and interleaving is allowed. For example, if command chaining has been specified and channel end is presented separately from device end, the channel or CTL-interface logically disconnects from the control unit to make the interface available for other disk operations. In block multiplexing mode, the channel or CTL-interface also provides a 'channel available' interruption to indicate to an instruction which previously caused a channel busy response (condition code 2) that it may now proceed.

The E bit is set by the IPU for all IPU and IOP communication and in all normal cases is reset by the selected IOPs. If an E bit is not reset at the end of an IOP-IPU communication, this condition signals the IPU that something is wrong (microprogram did not read the instruction and causes the IPU to go to machine check IRPT).

I/O Condition Codes

- Resulting from certain tests performed by the IOP a condition code (CC) is formed. This condition code is generated within the IOP and is set, via the MSC-LS common register, into PSW bits 34 and 35
- The condition code is set after initiation and after execution of an I/O instruction, thereby releasing the IPU-MSC for further processing
- The condition code indicates whether or not the IOP has started or executed the operation, and also indicates the reason for the rejection (see diagram below)
- Subsequent 'branch-on-condition' operations can be used to decide what actions have to follow.

Conditions	Condition Codes			
	SIO	**TIO**	**HIO**	**TCH**
Available	00	00	01	00
Interrupt pending in the device	01	01	01	00
Device is working	01	01	01	00
Device is not operational	11	11	11	00
Interrupt pending in subchannel	10	01 or 10*	00	00
Subchannel working	10	10	01 or 00*	00
Subchannel is not operational	11	11	11	00
Interrupt pending in channel	(Note 3)	(Note 3)	(Note 3)	01
Channel working	10	10	10	10
Channel is not operational	11	11	11	11

Notes:

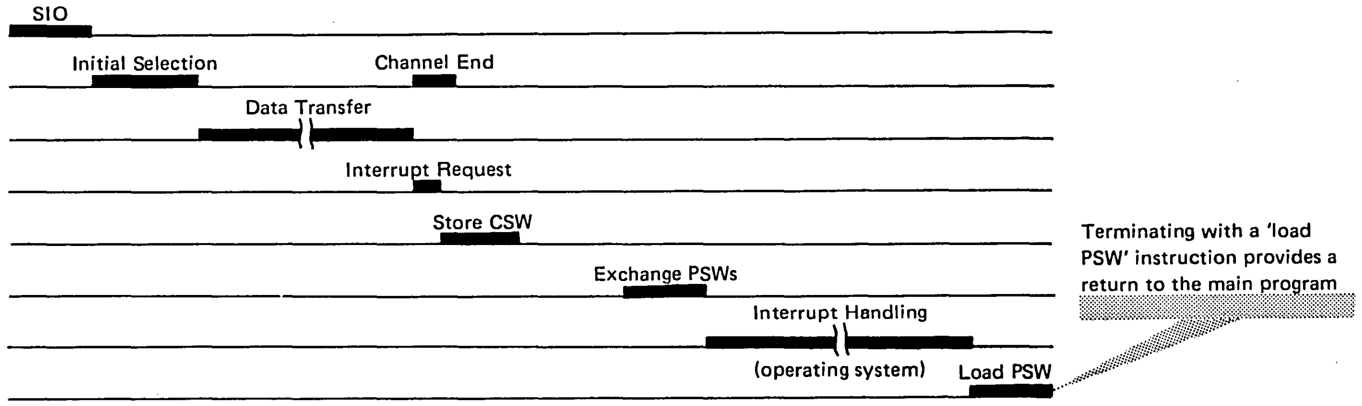
1. Condition code 01 indicates that either a complete CSW or the status portion of a CSW is stored
2. Condition codes marked with *:
   The first condition code is for the addressed device
   The second condition code is for another device (but only if it uses a shared UCW of the multiplexer channel)
3. A channel that contains a pending interrupt appears the same as an 'available' channel. The condition code depends upon the state of the subchannel and the device.

- The 'STIDC' instruction is not handled in the same manner as the other I/O instructions (see Pages 2-015 and 2-185 for further information).
- The STIDC condition codes are generated entirely within the IPU and are as follows:
  - 00 : Indicates that the channel identification is correctly stored
  - 01 : Indicates that the CSW is stored
  - 10 : Indicates that the channel is active, but execution is prohibited
  - 11 : Indicates that the channel is not operational.

Interrupts

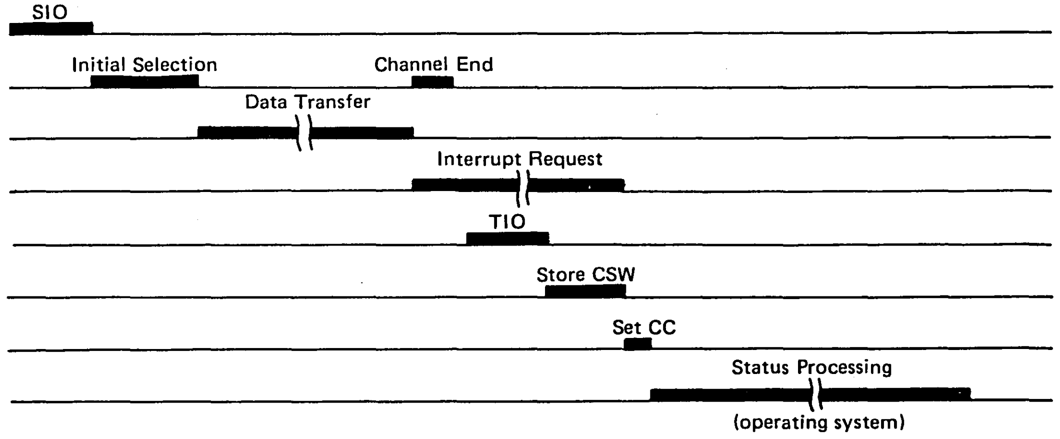
- At the completion of an I/O operation, the IOP requests an interrupt
- If this interrupt is accepted (that is, it is not masked off and none is waiting with a higher priority) the following events happen:

  The Interruption code (device address) is stored into MSC-LS common register
  The CSW is stored into main storage
  The interrupt condition is reset
  The PSW is exchanged in the IPU
  The interrupt handling routine in the operating system evaluates the CSW and initiates the appropriate steps
- In the example below, the PSW exchange (that is, store old PSW, load new PSW to lead to interrupt handling routine) is performed automatically in the IPU.



Terminating with a 'load PSW' instruction provides a return to the main program

- If an interrupt is not accepted it is stacked. (Channels and/or I/O devices that have stacked interrupts respond to subsequent SIO instructions issuing the 'busy' status)
- When an interrupt is stacked, a 'TIO' instruction may be used to clear the interrupt condition, but a condition code is set
- In the example below, the PSW exchange is *not* performed automatically. The period called 'status processing' can be part of any program depending on the operating system that is being used. Further actions by the operating system depend upon the result of status processing.



IOP Microprogram

Task and Operation of IOPs

The IOP, in connection with one or more front ends, controls I/O operations and links one or more I/O devices to SVP, IPU, and MSC.

The IOP initiates I/O operations, performs data transfers between I/O devices and the MSC and terminates I/O operations.

The IOP may be considered as a common and standardized part of an I/O attachment. The front ends in turn are those parts of an attachment, that fulfil the individual requirements of the I/O devices to be attached to an IOP. The front ends represent implementation of the required circuitry, and perform those functions that exceed the capacity of the IOP.

Each IOP is controlled by its own device oriented microprograms (see example on page A-500). These individual microprograms are stored in the control storage of each IOP and are loaded under control of the SVP during IMPL.

After IMPL is successfully completed each IOP idles in its basic loop and waits for selection or I/O service requests.

These microprograms may either run in:

- Normal mode, or
- Time slice mode.

Normal mode may be used if only one microprogram (or routine) is to be executed. Time slice mode allows the execution of several microprograms (or routines) simultaneously.

Time slicing here means that the execution of one or more microinstructions of the one routine is followed by the execution of one or more microinstructions of another routine, but in a pre-determined sequence.

Execution of one or more microprograms (or routines) is controlled by index words. Index words consist of:

- A pointer portion, and
- A link portion.

The pointer portion points to the Instruction Address Register (IAR) that holds the control storage address of the instruction to be executed, while the link portion always leads to the next index word. (See also pages 3-050 and 3-500).

These index words (or chain of index words) are generated during initialization, after IMPL has been successfully completed.

These may be altered in a pre-determined way (according to the time requirements of the attached I/O device) during execution of the microprograms either by:

- Load ALS instructions, or
- Trap bits.

Load ALS instructions allow the overwriting of ALS registers, while the trap bits are directly ORed to the link portion of an index word in the IOPs circuitry.

The last index word of the chain always leads back to the first index word of the chain.

This ensures execution of microinstructions and routines in a proper and pre-determined sequence.

A chain of index words may also be defined as a time slice period.

Normally an IOP runs in its basic loop monitoring:

- Select from IPU and
- I/O service requests.

This basic loop mainly consists of a number of test and branch operations.

After selection of an IOP by the IPU, the IOP fetches, under control of its own microprogram, all necessary information from the MSC, to execute an I/O instruction.

After the I/O instruction is initiated, the IOP releases the IPU for further processing, while the IOP supervises the execution of the I/O operation under control of its own microprogram.

For data transfers of IOP '8' and IOP '9', the IOP requests service from MSC because data transfers are microprogram controlled.

For data transfers of IOP 'A', the front end circuitry requests service from MSC because here data transfers are circuitry controlled.

If a request is accepted by the MSC, the MSC activates its select line and the data transfer takes place.

After the data transfer is completed, MSC inactivates its select line, the IOP continues in its microprogram, and either the IOP or the front end prepares the next data transfer.

After the last data byte has been handled, the status byte is generated and the IOP activates its interrupt request line to IPU.

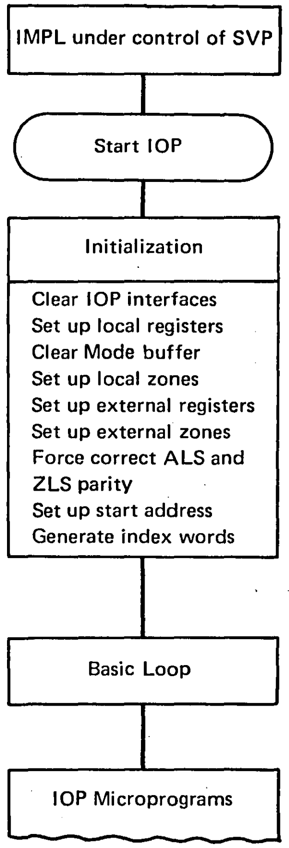
If the interrupt is accepted by the IPU (none waiting with higher priority, and not masked off) the IPU selects the requesting IOP. The IOP stores the I/O device address (interruption code) into the MSC-LS common register and CSW information into main storage.

The IOP signals to the IPU, when this information transfer is completed. This causes the IPU to continue in its own program which normally is the interrupt handling routine, while the IOP returns to its basic loop, waiting for next select or I/O service request.

Because IOPs are controlled by their own individual microprograms, the flowcharts on the following pages can show only the convention that all IOP microprograms have to follow.

For individual and detailed information refer to respective front end documentation.

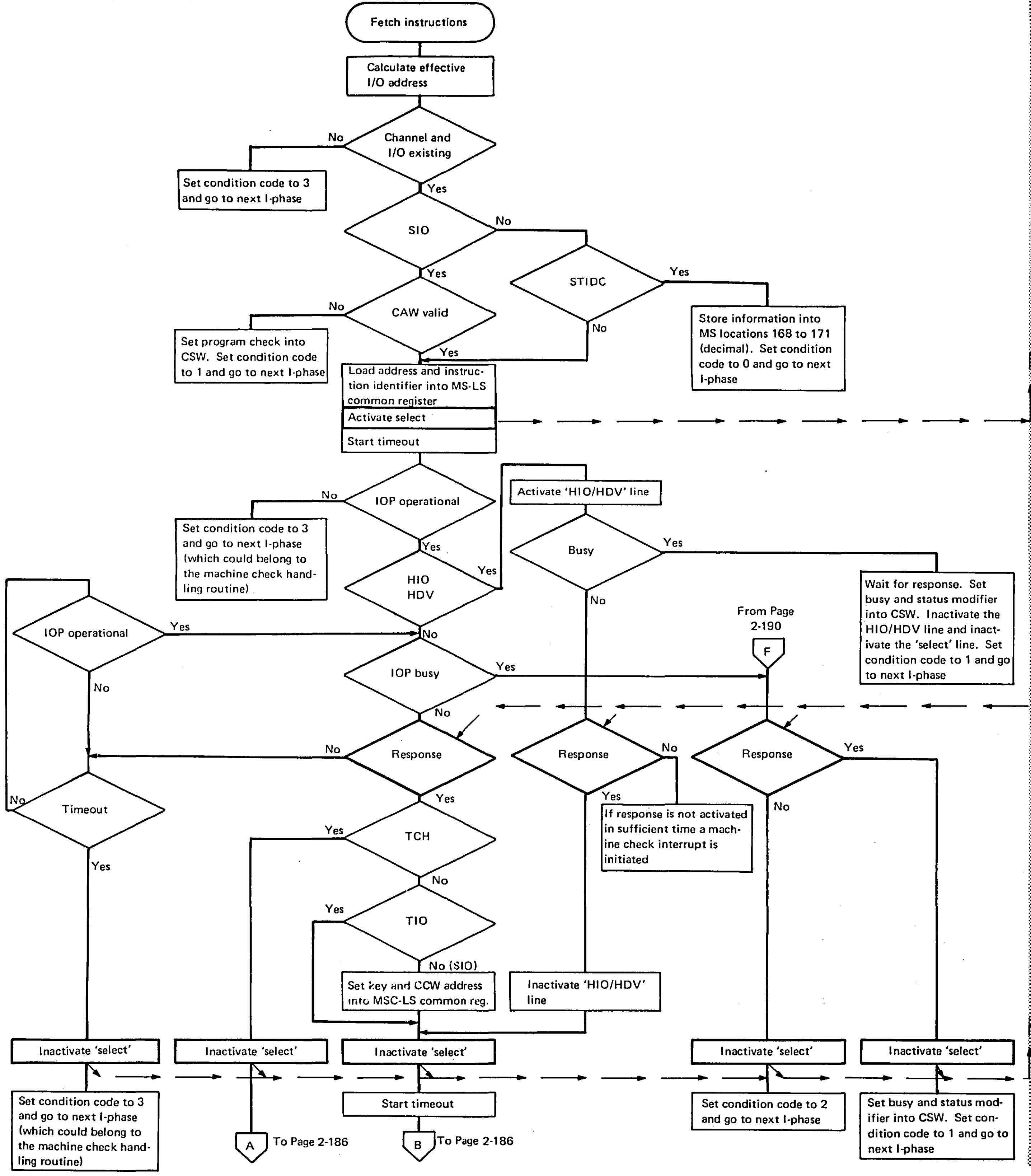
Execution of IOP microinstructions and cycle arrangement is shown in Chapter 3.



IPU/IOP Communications

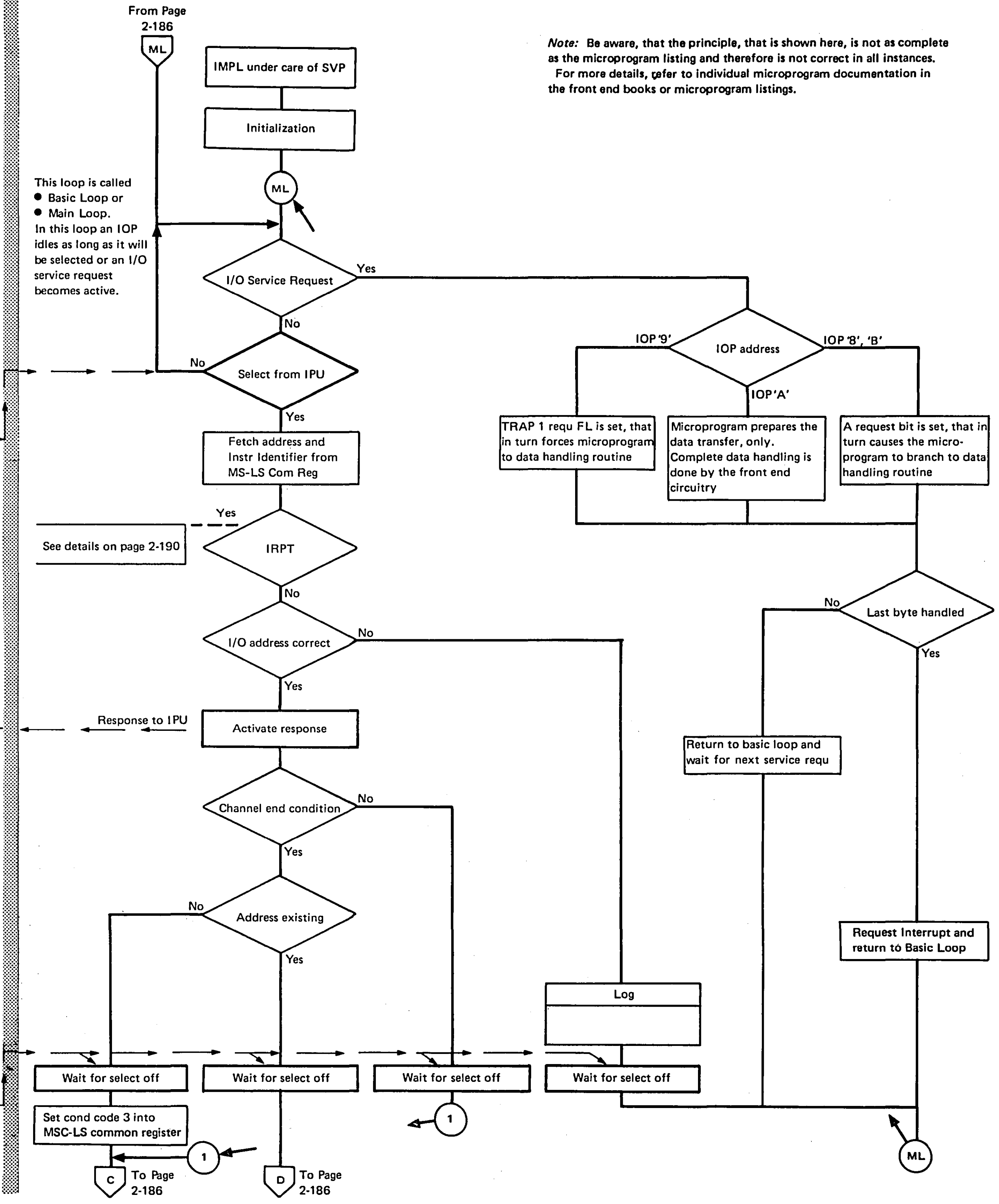
- Microprogram convention

IPU Microprogram (that part which controls the IOPs)



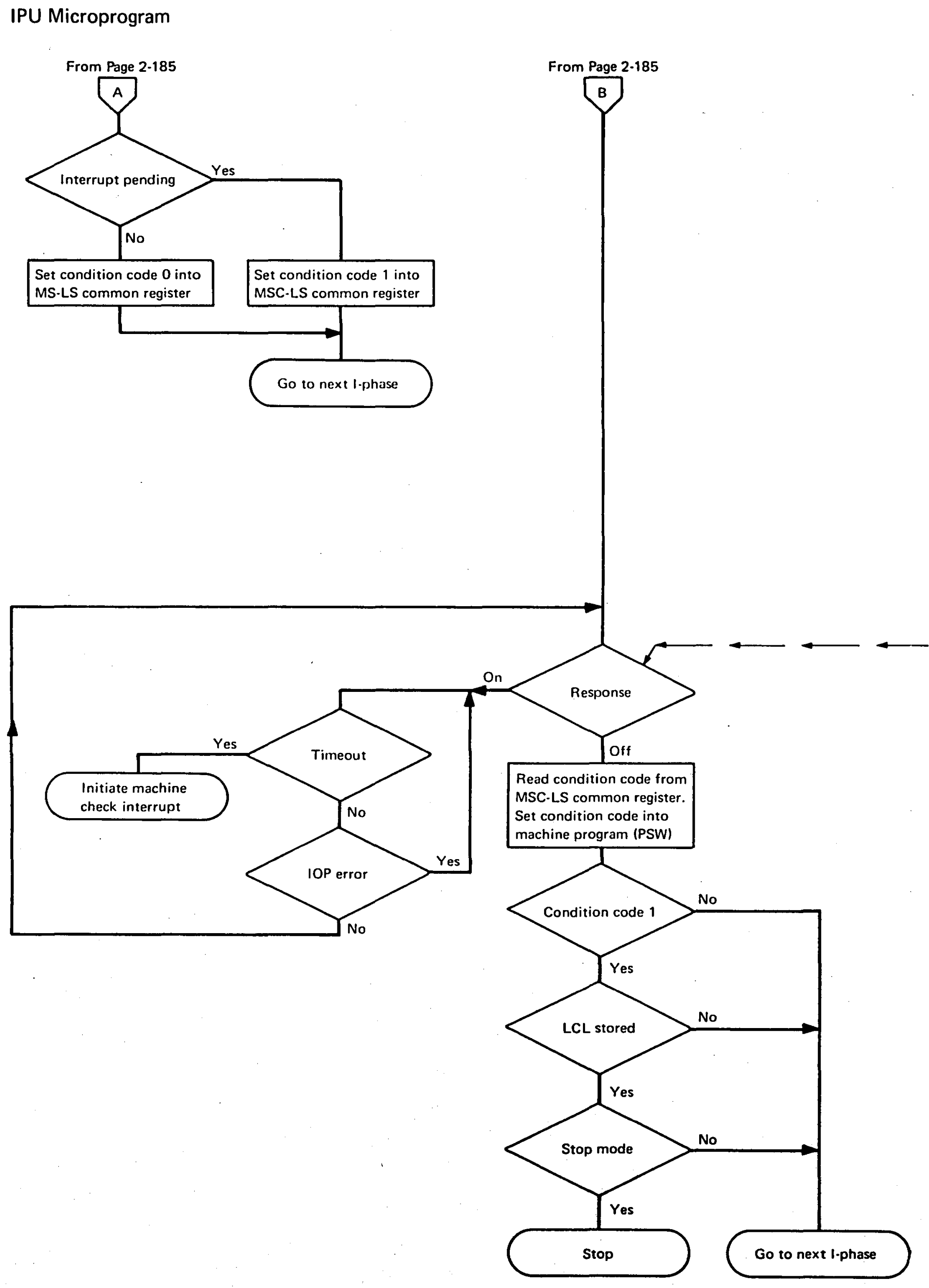
IOP Microprogram Principle

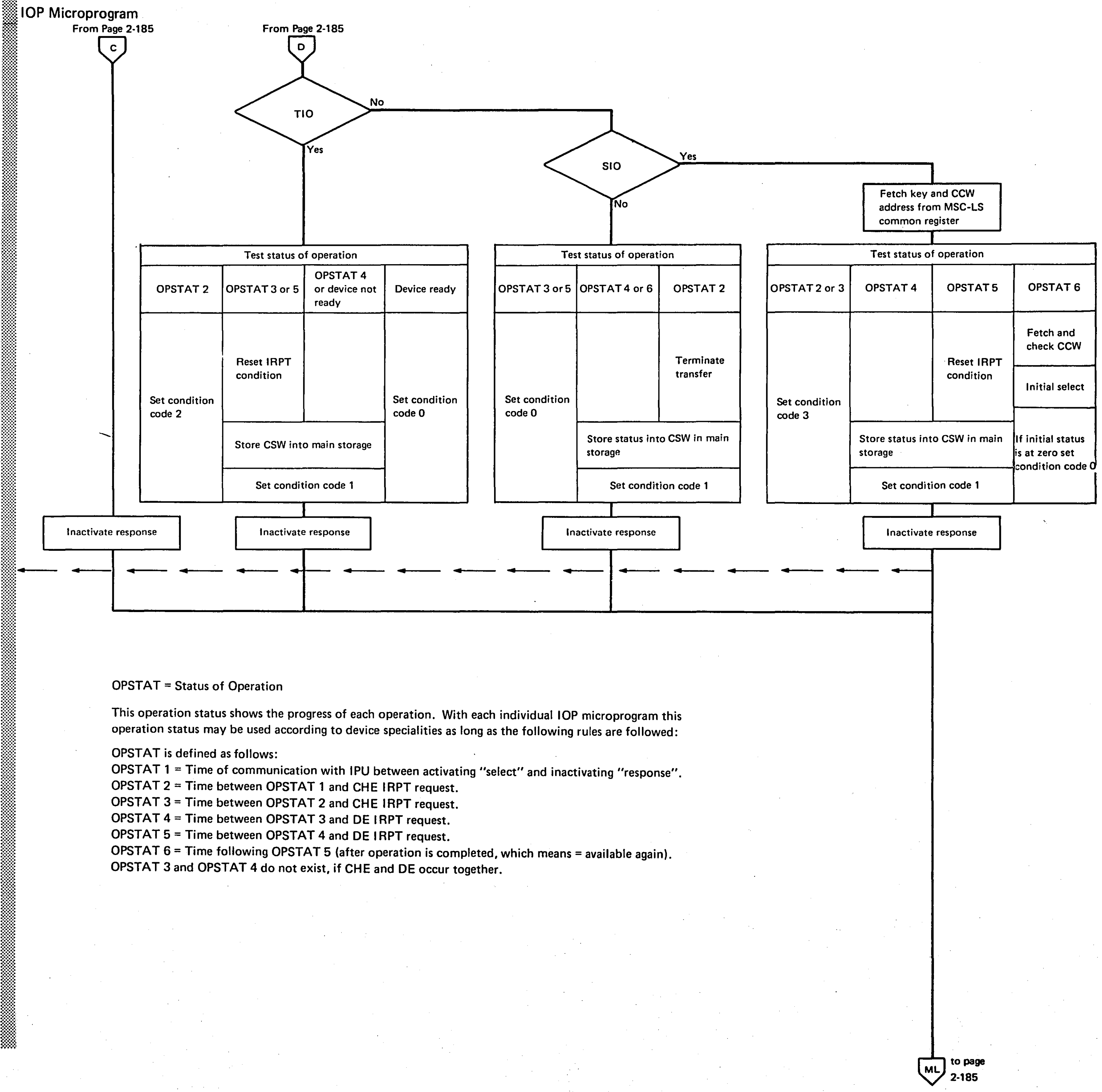
Note: Be aware, that the principle, that is shown here, is not as complete as the microprogram listing and therefore is not correct in all instances.
For more details, refer to individual microprogram documentation in the front end books or microprogram listings.



IOP Microprogram (continued)

IPU/IOP Communications (continued)

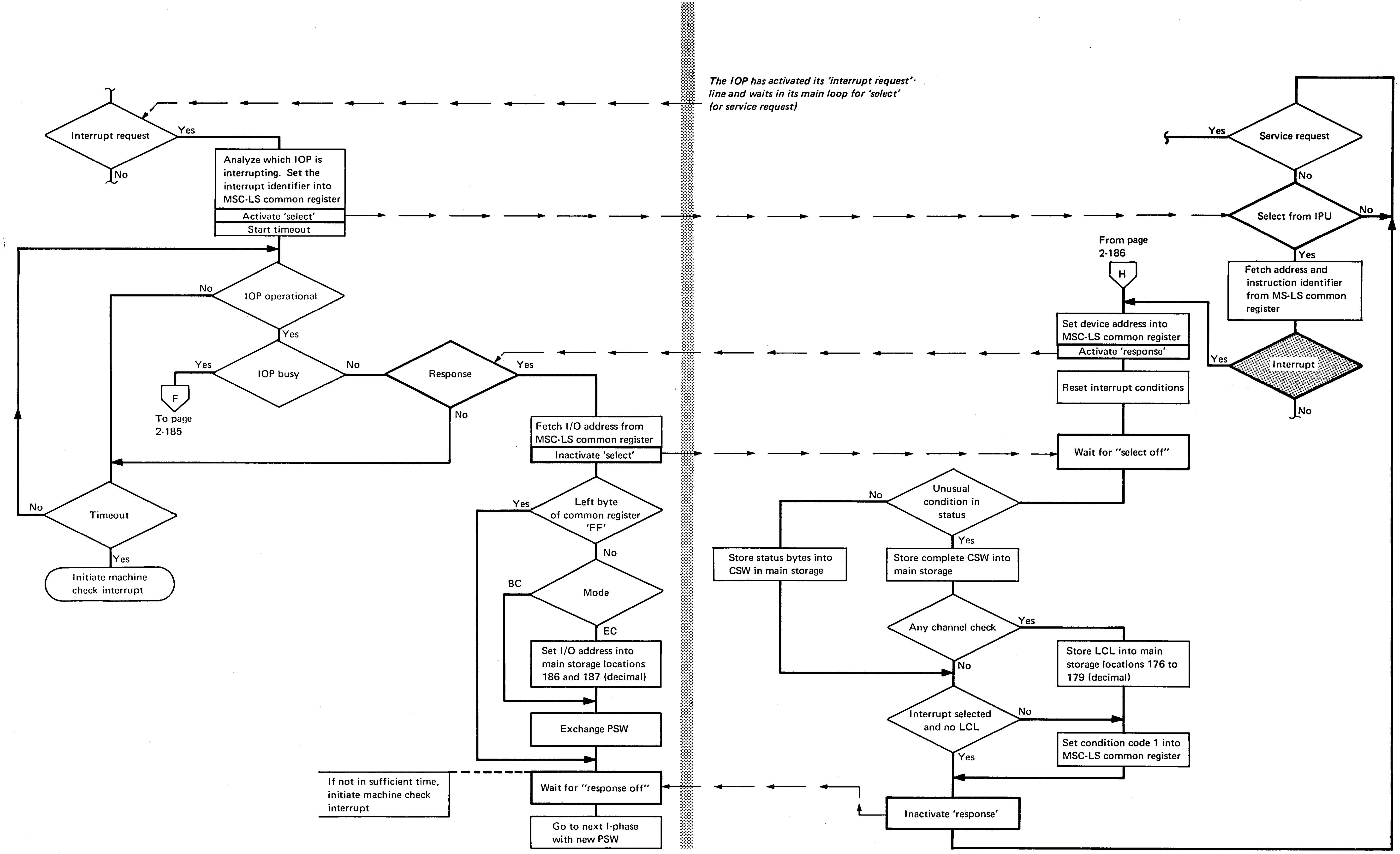




OPSTAT = Status of Operation

This operation status shows the progress of each operation. With each individual IOP microprogram this operation status may be used according to device specialities as long as the following rules are followed:

OPSTAT is defined as follows:
OPSTAT 1 = Time of communication with IPU between activating "select" and inactivating "response".
OPSTAT 2 = Time between OPSTAT 1 and CHE IRPT request.
OPSTAT 3 = Time between OPSTAT 2 and CHE IRPT request.
OPSTAT 4 = Time between OPSTAT 3 and DE IRPT request.
OPSTAT 5 = Time between OPSTAT 4 and DE IRPT request.
OPSTAT 6 = Time following OPSTAT 5 (after operation is completed, which means = available again).
OPSTAT 3 and OPSTAT 4 do not exist, if CHE and DE occur together.


The IOP has activated its 'interrupt request' line and waits in its main loop for 'select' (or service request)
Interrupt request
Yes
No
Analyze which IOP is interrupting. Set the interrupt identifier into MSC-LS common register
Activate 'select'
Start timeout
IOP operational
No
Yes
IOP busy
Yes
No
F
To page 2-185
Response
Yes
No
Timeout
No
Yes
Initiate machine check interrupt
Fetch I/O address from MSC-LS common register
Inactivate 'select'
Left byte of common register 'FF'
Yes
No
Mode
BC
EC
Set I/O address into main storage locations 186 and 187 (decimal)
Exchange PSW
If not in sufficient time, initiate machine check interrupt
Wait for "response off"
Go to next I-phase with new PSW
Service request
Yes
No
Select from IPU
No
Yes
Fetch address and instruction identifier from MS-LS common register
Interrupt
Yes
No
From page 2-186
H
Set device address into MSC-LS common register
Activate 'response'
Reset interrupt conditions
Wait for "select off"
Unusual condition in status
No
Yes
Store status bytes into CSW in main storage
Store complete CSW into main storage
Any channel check
Yes
No
Store LCL into main storage locations 176 to 179 (decimal)
Interrupt selected and no LCL
No
Yes
Set condition code 1 into MSC-LS common register
Inactivate 'response'

IOP Microprogram (continued)

Indirect Data Addressing

Indirect data addressing may also be used with CCWs. If so, the IDA flag is set, indicating that the data address is an indirect address.

The indirect address has to be converted to the actual data address before data transfer may start. This means that the indirect address of the CCW is transferred to the MSC-LS and addresses the IDA list in main storage.

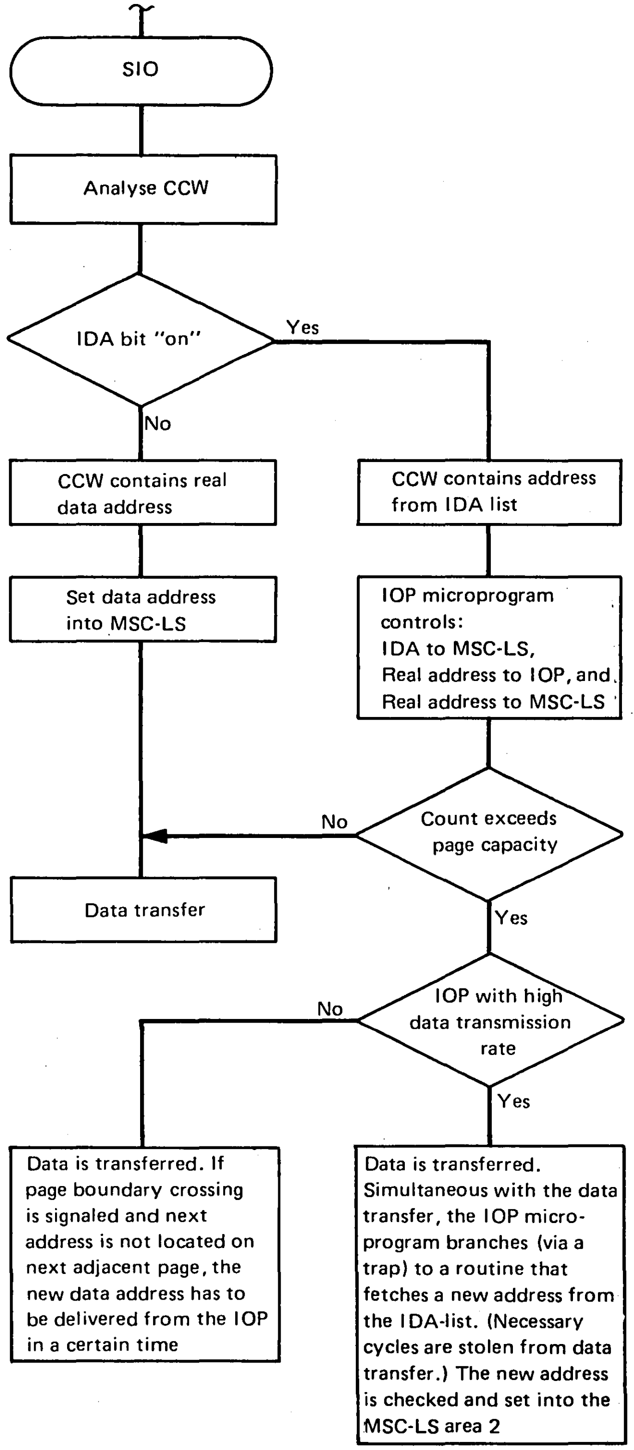
- An actual data address is transferred to the IOP, where the actual data address is checked, stored, and again transferred to the MSC-LS. Data transfer can now start.

  The very first address fetched from the IDA list, can specify any byte location in a storage page or area, but subsequent addresses must specify the *first page address* for forward operation, or *last page address* for backward operation.

If the address fetched from the IDA list does not conform to system architecture, the IDA check indicator is set. This results in a program check as soon as the new address is used.

For more information about indirect addressing and paging, refer to *3125 Processing Unit, Main Storage Controller,* Maintenance Library Manual, Order No. SY33-1061 and *3125 Processing Unit, Instruction Processing Unit,* Maintenance Library Manual, Order No. SY33-1062.

For a definition and use of chain control line "page boundary", see the following page 2-120.



Chaining

IOP activity started by an 'SIO' instruction may be continued by fetching a new CCW instead of using a second 'SIO' instruction. Chaining is indicated by a flag in the CCW.

Chaining proceeds in an ascending order of addresses, which means that the new CCW address is obtained by adding "eight" to the address of the currently used CCW. This indicates that chaining only takes place between CCWs that are located in successive doubleword locations in storage.

If CCWs that are located in non-contiguous storage areas are chained, a CCW using the 'transfer in channel (TIC)' command has to be used.

Two types of chaining are available:

Data chaining (CD)
Command chaining (CC).

Data Chaining (CD)

In data chaining, the new CCW provides a new start address for the new data storage area only. Data chaining is considered to occur immediately after the last data byte designated by the current CCW, has been transferred.

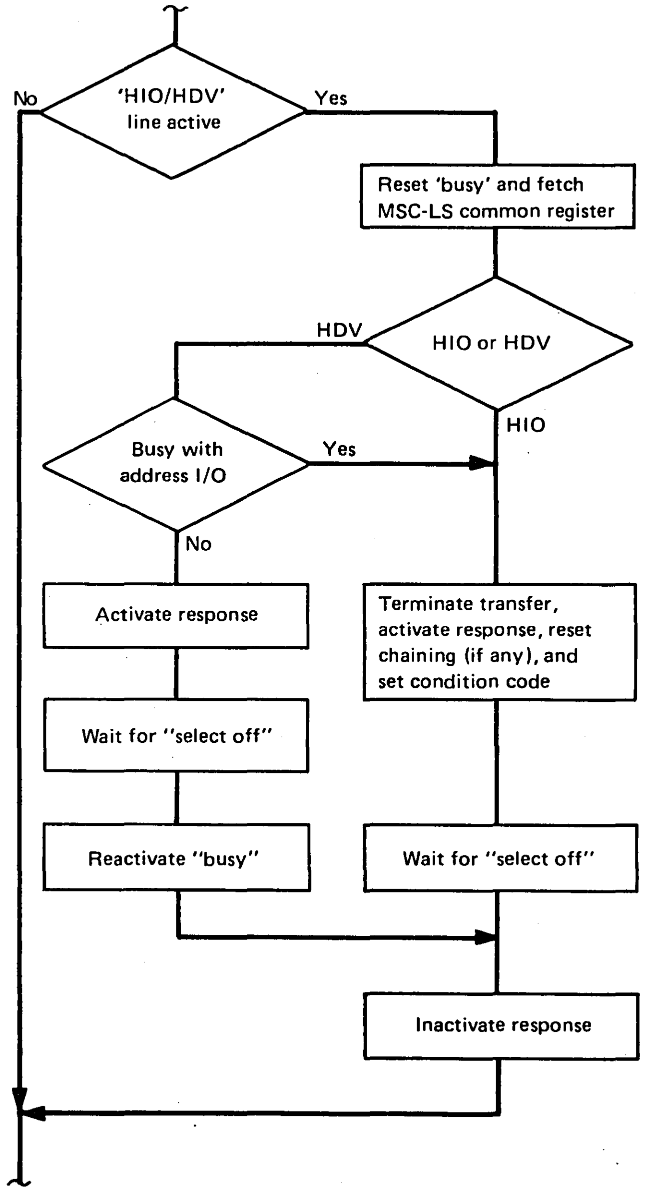
Command Chaining (CC)

In command chaining, the new CCW provides a new command and specifies a new I/O operation. The IOP fetches the new CCW and initiates the new operation after receipt of the 'device end' signal of the current operation. Completion of the current operation does not cause an I/O interruption.

'IOP Busy' Line

In opposition to IOPs that serve low-speed I/O devices (for example card reader, card puncher, line printer, communication adapter) other IOPs sometimes may not be able to answer the 'select' from the IPU. In this instance, the busy IOP periodically tests its 'HIO/HDV' line (bit 2 in the sense register); (see Page 4-068) and keeps its 'busy' line active.

The flowchart below illustrates the microprogram principle.



Chapter 3. Operational Details

Microinstructions

Visual Index of Groups

- IOPs are microprogram controlled and use their own microprogram language.
- Each microinstruction is 3 bytes or 6 hexadecimal characters wide. The first hexadecimal character is not used directly so that each microinstruction consists of 20 function bits.
- The IOP microprogram language consists of four main groups of instructions.
- These groups are:

Microinstruction Group	Definition of Microinstructions	C-Field Bits 2, 3 Part of 3rd Hex Character	Y-Field Bit 0 Part of 5th Hex Character
1	Branch Instructions	00	U
	Test Bit and Branch Instructions	01	0
2	Data Store/Load Instructions	01	1
3	Move Instructions	10	U
4	Logic Instructions	11	U

Further definition of the single microinstructions is achieved by decoding:
C-Field bits 4 to 7
R-Field bit 2
Y-Field bits 0 and 1.
For more details refer to the following pages:

Description of microinstruction groups, pages 3-015 and 3-020.

Microinstruction decoding, page 3-030.

Access cycle (all instructions), pages 3-050, 3-055.

Process cycles, microinstructions of
Group 1, pages 3-1XX
Group 2, pages 3-2XX
Group 3, pages 3-3XX
Group 4, pages 3-4XX.

Group 1: IOP Branch Instructions

Primary Function: The microprogram branches to another instruction anywhere in the control storage instead of continuing with the next sequential instruction. The branch may be unconditional or depend on a certain condition.

Secondary Functions: Program level switching is possible in case of a successful branch.

Group 2: IOP Data-Storage Instructions

Primary Function: The primary function either stores a byte from a register into data storage or loads a byte from data storage into a register. Multibyte transfer is possible if the multiple byte bit (ALS-B bit 0) has been set by a previous instruction. In such a case storage- and register-address will be incremented by one after each byte transfer. The operation is repeated until a double word boundary (8 bytes) in the register area is reached. In other words the operation is ended after a register with three low order one-bits in its address has been loaded or stored.

Secondary Functions: It is possible to increment or decrement the data-storage address contained in the register defined by the R-field. It must be emphasized that this type of increment/decrement has nothing to do with the incrementing in multibyte transfer. It is performed before the byte transfer, therefore, it affects the program only when this instruction is executed the next time.

Group 3: IOP Move Instructions

Primary Function: To move a byte of information from one location to another.

Secondary Function: With the "move" and "move crossed" instruction, level switching (main routine to subroutine and vice versa) is possible.

Group 4: Logical IOP Instructions

Primary Function: The contents of the 'From'-register and the 'To'-register are logically combined by the ALU (Added, ANDed, ORed or Exclusively ORed). Unless it is a test type instruction, the result is stored into the 'To' work register defined by the R-field.

Secondary Functions: The ALU condition code is set depending on the ALU output to indicate whether the ALU output (D-register) was zero or not and to indicate a carry or no carry out of the high order position of the ALU.
With test operations (mnemonic Txx..) the result is not stored into the 'To' work register, because the purpose of these instructions is only the setting of the ALU condition code.
With suffix 'U' instructions MIAR and SIAR will be interchanged (level switching).

Groups 1 & 2

The contents of the op-register is divided into three portions called: C-, R-, and Y-field.
Generally:

C-field represents the op-code.
R-field represents the *to* register.
Y-field represents either the *from* register or immediate data.

Group 1

Branch Operations

Test and Branch on Bit Conditions

- These instructions allow branching if:
  One bit is "on" or "off"
  All bits are "on" or "off".
- C-field bit 4 specifies whether branch is performed on the bits "on" or "off" condition.
- C-field bits 5, 6, and 7 specify the bit (within the byte) that is to be tested.
- R-field indirectly specifies the byte that is to be tested.
- Y-field contains the low-order part of the branch address (to be loaded into CSAR if the branch is successful).

Unconditional Branch

- Branch address is the R- and Y-field.
- If C-field bit 5 is "off", R-field is set into the high-order portion of CSAR and Y-field is set into the low-order portion of CSAR.
- If C-field bit 5 is "on", Y-field specifies a register. The content of this register is set into the low-order portion of CSAR.

Conditional Branch

- These instructions allow branching on conditions that were set in previously performed arithmetic or logical instructions.
- R-field is set into the high-order portion.
- Y-field or the content of the register specified by Y-field, is set into the low-order portion of CSAR, according to the setting of C-field bit 5.
- The branch condition is the ALU condition code of the cycle in which the branch operation is performed.

Note: In the event of time slicing, the branch address is stored in ALS and not set into CSAR. The branch address then becomes active when the pointer of the program, interrupted by time slicing, becomes active again.

Group 2

Load and Store Operations

Load Single Byte

- R-field contains a local or external register address, which in turn contains the displacement portion of the control storage address.
- The contents of this register is set, via the ALU into CSAR-D.
- The block portion of the control storage address has to be set into CSAR-B by a separate instruction.
- C-field bits 5 to 7 specify the amount of increment and decrement and C-field bit 4 specifies whether the address has to be incremented or decremented.
- Y-field specifies any register into which control storage data is to be set.

Store Single Byte

- This instruction uses the same principle as the load operation except that the flow of the data is reversed.
- Y-field specifies any register, from which data is stored into control storage.

Load and Store Multiple Byte

- These instructions allow fetching or storing of up to eight bytes.
- Data is transferred to or from DLS registers only. (The external registers may not be specified.)
- The number of bytes to be transferred is determined by the location of the first (or "start") register within an eight-register block.
- If the first register of a block is specified, eight registers are transferred.
- If the fifth register of a block is specified, three registers are transferred.
- The "start" register is specified by the Y-field of the instruction. The succeeding registers are addressed in ascending sequence up to the next byte block boundary.

Note: In the event that one of these operations is interrupted by time slicing, the necessary addresses are stored in the DLS and ZLS.
With LIST operations data and displacement registers must not both be external.

Microinstructions (continued)

Groups 3 & 4

Group 3

Move Operations

Immediate Data into Register

- This instruction loads Y-field into the register that is specified by R-field

Register to Register

- These instructions allow the transfer of data from register to register
- R-field specifies the *to* register
- Y-field specifies the *from* register.

Immediate Data into ALS, ZLS, or SVP Link

- These instructions allow loading of Y-field into either ALS, ZLS, or SVP link
- ALS, ZLS, and SVP link are specified by the C-field
- R-field specifies the *to* address or control information.

Register into ALS, ZLS, or SVP Link

- These instructions set the contents of a register that is specified by the Y-field into either ALS, ZLS, or SVP link
- ALS, ZLS, and SVP link are specified by the C-field
- R-field specifies the *to* address or control information.

SVP Link Data into Register

- These instructions allow the transfer of data from SVP link into local or external registers of the IOP
- Y-field specifies the *to* register, C-field specifies the SVP link, and R-field contains the *from* address.

Group 4

ALU Operations

Immediate Data and Register with Result

- One operand is represented by the immediate data in the Y-field of the instruction itself
- The second operand resides in the register that is specified by the R-field
- The result is set into the *to* register, which means that the original second operand is overwritten
- An ALU condition code is stored into the ALU condition code buffer
- ALU conditions are:
  ALU zero or not
  ALU carry or not.

Immediate Data and Register without Result (ALU Condition Code Only)

- Logically, these operations are the same as for "Immediate Data and Register with Result" with the exception that the result is not stored. (Both operands remain unchanged)
- An ALU condition code is stored into the ALU condition code buffer.

Register and Register with Result

- Both operands are represented by the contents of the registers. (A register may be either local or external, but only *one* register is allowed to be external)
- The logical result is set into the *to* register, which means that the second operand is overwritten
- An ALU condition code is stored into the ALU condition code buffer.

Register and Register without Result (ALU Condition Code Only)

- Logically, these operations are the same as for "Register and Register with Result" with the exception that the result is not stored. (Both operands remain unchanged)
- An ALU condition code is stored into the ALU condition code buffer
- Only *one* register is allowed to be external.

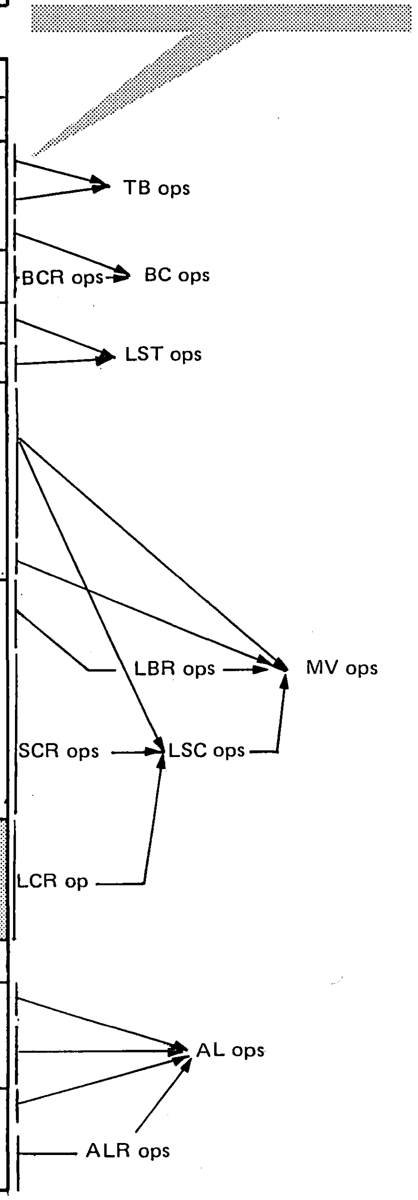
Decoding of Microinstructions

- This table shows the decoding of the op-register contents for the different microinstructions.

Hex characters on screen	DS	L/ST	I	CS Pty	2nd	3rd	4th	5th	6th

Instruction Group	Instruction			C-Field						R-Field						Y-Field							
				4	5	6	7	2	3	2 Note 5	3	4	5	6	7	0	1	2 Note 5	3	4	5	6	7
1	Test Bit and Branch	TB on	Bit "on"	0	Bit position to be tested			0	1	Address of register that contains byte to be tested						0	Displacement portion of control storage address						
		TB off	Bit "off'	1																			
	Branch on Condition	BC	Immediate	Note 2	0	ALU cond code mask (Note 4)		0	0	Block portion of control storage address						U							
		BCR	Register		1												▨	Address of register that contains displacement					
2	Store	SDEC, SINC		Note 1	Incr/Decr amount			0	1	Register that contains the control storage address						1	0	"From" register					
	Load	LDEC, LINC															1	"To" register					
3	Store Byte Immediate		(SABI)	0	0	1	0	1	0	0	ALS-B address					Immediate byte							
			(SADI)							1	ALS-D address												
			(SZI)				1			0	ZLS address												
			(SLKI)							1	SVP link address												
	Load Byte Immediate	LBI		1	0	0	▨	1	0	"To" register													
	Move LBR	MV	Straight	1	1		0	1	0							U	▨	"From" register					
		MVX	Crossed				1																
	Store Byte from Register		(SABR)	0	1	1	0	1	0	0	ALS-B address					0							
			(SADR)							1	ALS-D address												
			(SZR)				1			0	ZLS address												
			(SLKR)							1	SVP link address												
	Load SVP Link to Register	LLKR		1	0	1	0	1	0	0	▒					0	▨	▒					
										1													
							1			0													
										1	SVP link address							"To" register					
4	Logic Instructions	Y-field contains immediate data	Test only	0	0	ALU function 00 = AND 01 = OR 10 = XOR 11 = ADD		1	1	"To" register						Immediate byte							
			With store result	1																			
		Y-field contains register address	Test only	0	1											U	Note 3	"From" register					
			With store result	1																			

These lines on the right-hand side of the table lead to microinstruction groups, which are defined by letters. These labels represent abbreviations that are used for microinstruction groups



Legend

▒ (Not used)

▨ (Not used, but is set to zero)

U Identifier for 'suffix-U' microinstructions. If this bit is "on" after the execution of the microinstruction, IARs are exchanged. (For further information, see Page 3-040.)

Notes:

1. Bit C4 = 0 – Increment (+ve)
   Bit C4 = 1 – Decrement (–ve).
2. Bit C4 = 0 – Bit "off" condition
   Bit C4 = 1 – Bit "on" condition.
3. Bit Y1 = 1 – ALU performs normal AND, OR, and XOR functions
   Bit Y1 = 0 – ALU performs ADD function, with bits C6 and C7 decoded as follows:

Bit C6	Bit C7	Function
0	0	ADD normal (reset previous carry)
0	1	ADD carry to bit 7 (previous carry)
1	0	ADD external bit (external carry)
1	1	ADD '01' (forced carry)

4. The ALU condition code masks are as follows:

Bit C6	Bit C7	Mask
0	0	No condition
0	1	Zero
1	0	Not carry
1	1	Zero or not carry

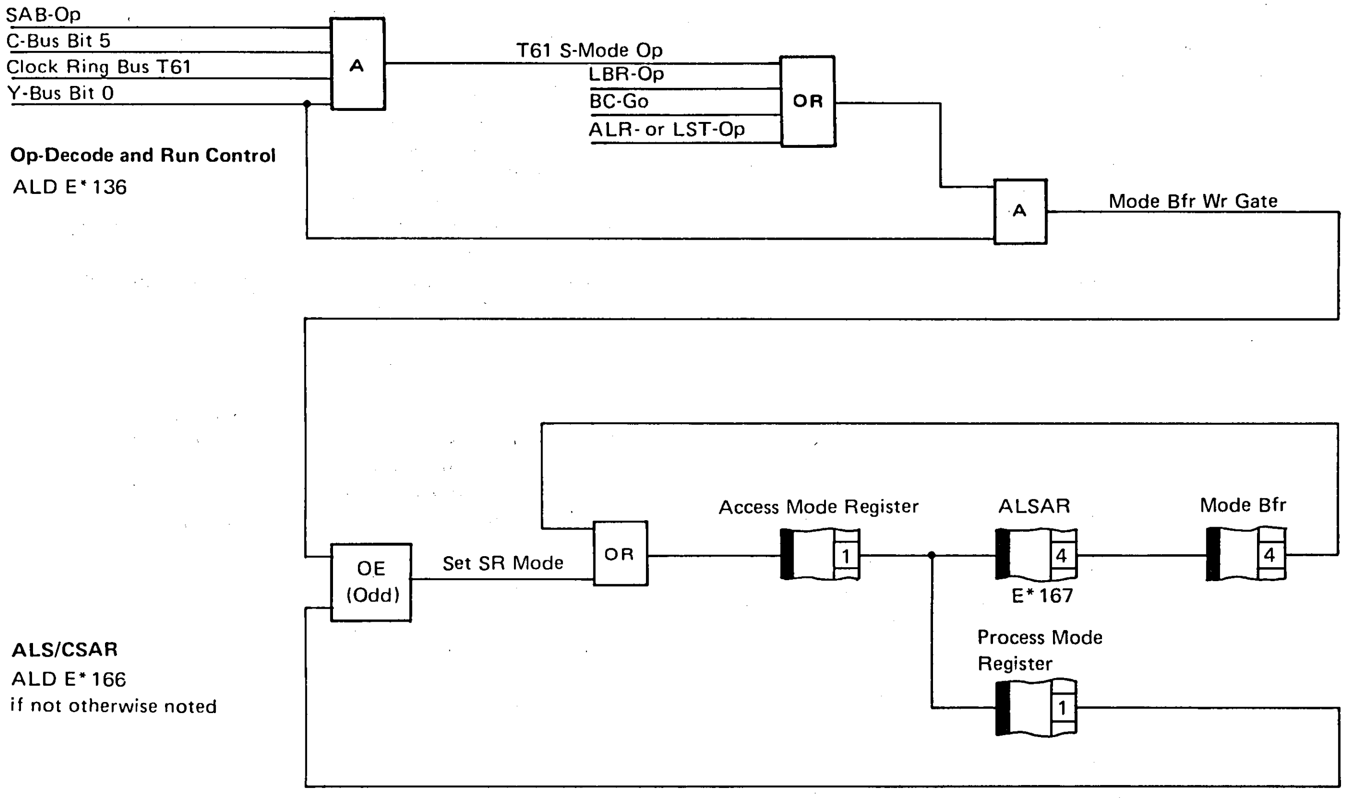
5. Bits R2 and Y2:
   If "on" specify external registers
   If "off" specify local registers (DLS)

   Exceptions:
   Bit R2: if LSC ops
   Bit Y2: if immediate data

Group 1 Microinstructions see Pages 3-1xx
Group 2 Microinstructions see Pages 3-2xx
Group 3 Microinstructions see Pages 3-3xx
Group 4 Microinstructions see Pages 3-4xx

Microinstructions (continued)

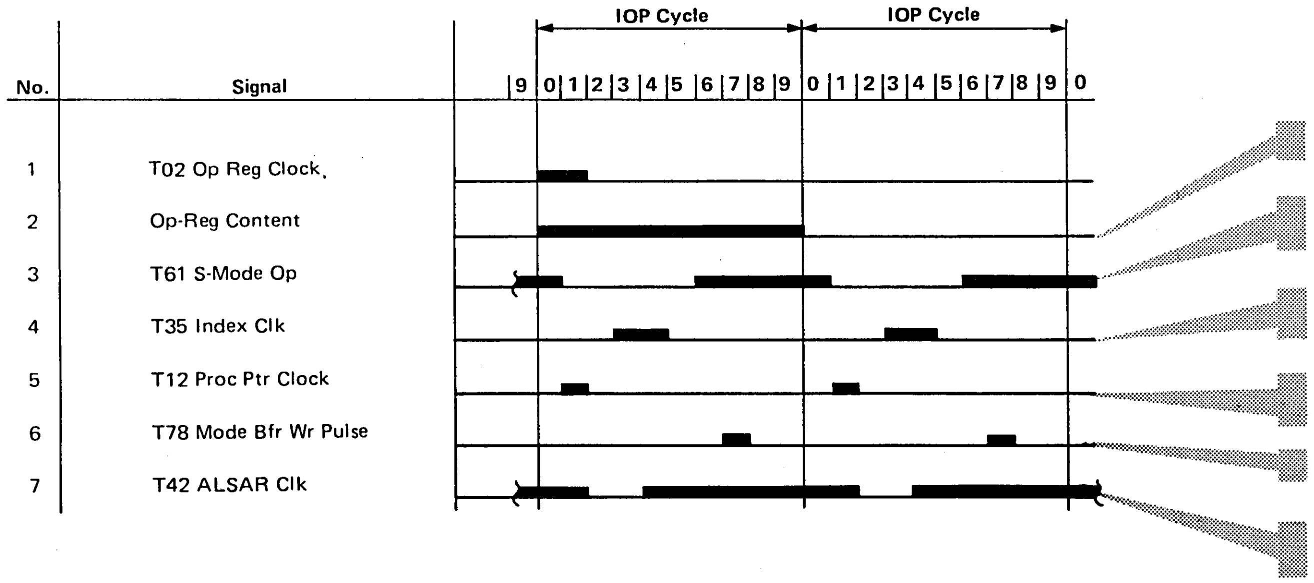
'Suffix U' Microinstructions



The microinstruction groups that allow the IARs to change are ANDed with Y-bus bit 0 (which is also called "suffix U" bit) to generate the 'mode bfr wr gate' signal.

'Mode bfr wr gate' is XORed with the existing mode to generate 'set SR-mode'. Thus, if SR-mode does not already exist, 'mode bfr wr gate' switches to SR-mode; if SR-mode already exists, 'mode bfr wr gate' resets SR-mode.

The contents of the mode buffer are set into the ALSAR via the access mode register and the SR bit "on" or "off" causes a change of IARs by addressing different locations in ALS.



Microinstruction in op-register

At the beginning of an access cycle the "mode" is analyzed.

During this time the mode that was found is set into Access Mode Register

During this time the mode is set into the process pointer register

During this time the mode is set into the mode buffer

During this time the mode is set into the ALSAR to address the desired IAR in ALS

Access Cycle for all Instructions

All microinstructions of the IOP microprograms (except those for data load/store operations) are performed in two cycles:

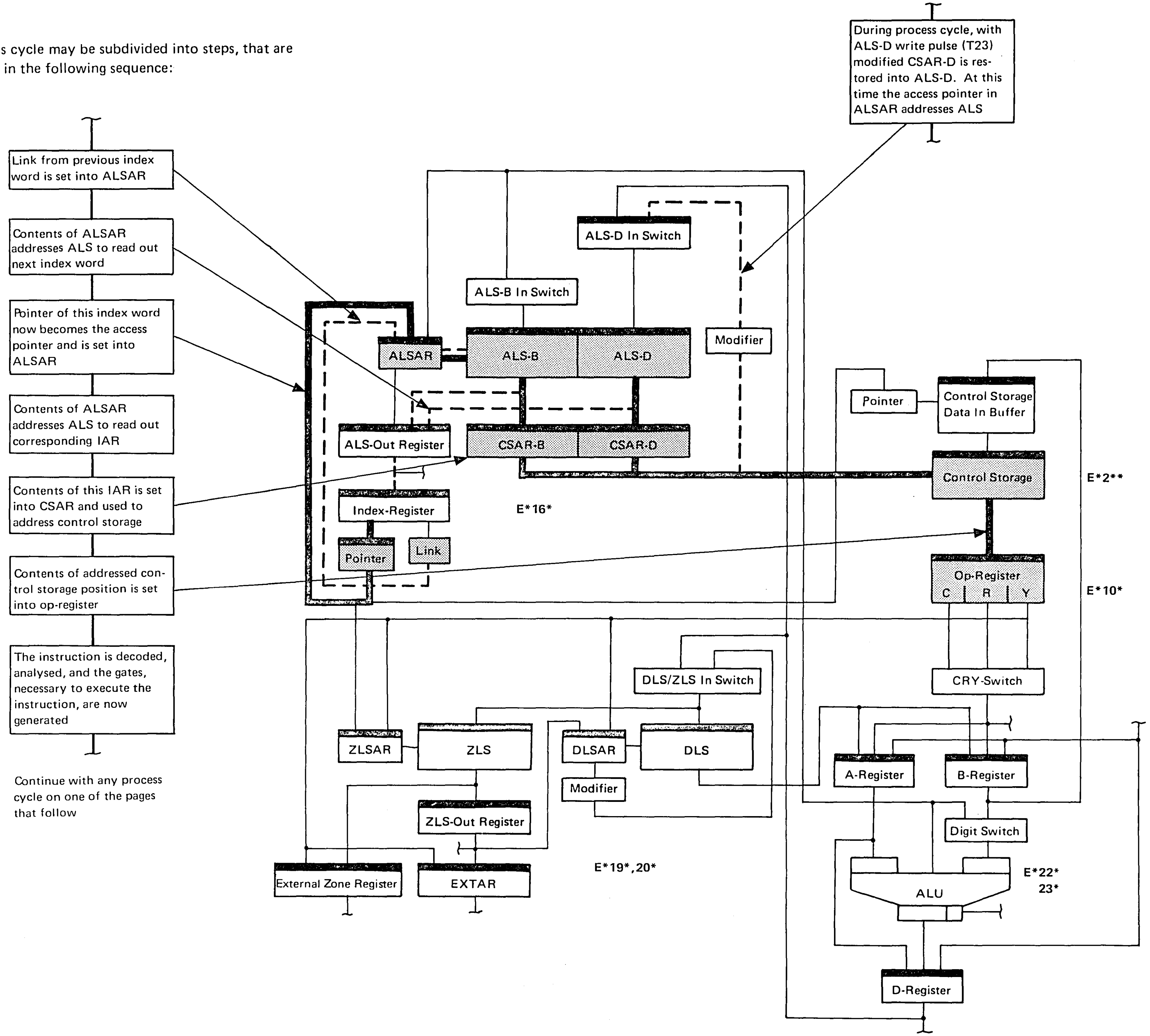
- The access cycle, during which an instruction is fetched from control storage
- The process cycle, during which an instruction is actually executed.

Because the access cycle is the same for all instructions, it is documented only once.

The representation of this access cycle may be used in connection with the documentation of process cycles of all instructions or instruction groups on the pages that follow.

Access cycle/process cycle principle and timing is shown on page 3-055.

- Each access cycle may be subdivided into steps, that are performed in the following sequence:



Link from previous index word is set into ALSAR

Contents of ALSAR addresses ALS to read out next index word

Pointer of this index word now becomes the access pointer and is set into ALSAR

Contents of ALSAR addresses ALS to read out corresponding IAR

Contents of this IAR is set into CSAR and used to address control storage

Contents of addressed control storage position is set into op-register

The instruction is decoded, analysed, and the gates, necessary to execute the instruction, are now generated

Continue with any process cycle on one of the pages that follow

During process cycle, with ALS-D write pulse (T23) modified CSAR-D is restored into ALS-D. At this time the access pointer in ALSAR addresses ALS

Microinstructions (continued)

Access and Process Cycle

Access and process cycles overlap. This leads to the impression that the instructions are performed in only one cycle.

The overlapping is implemented according to the following scheme:

An	An+1	
Pn-1	Pn	

The sequence in which the microprogram is executed, is controlled by index words, which consist of a pointer and a link portion.

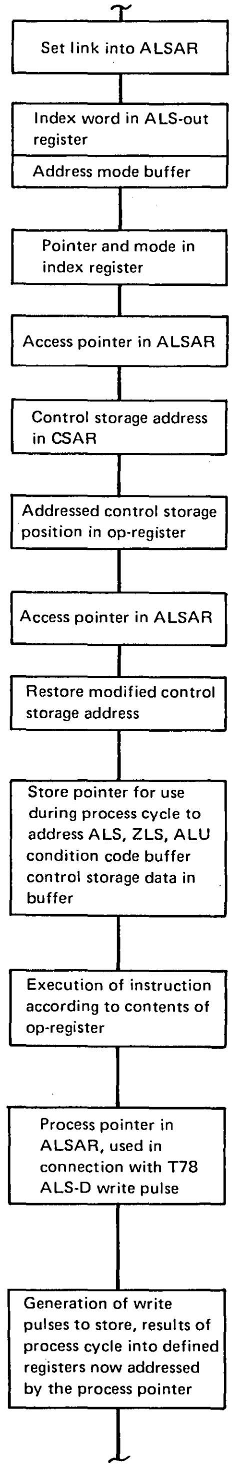
The pointer indicates an IAR, that holds the control storage address of the instruction to be executed, while the link leads to the next index word.

On the right, the flowcharts show the steps or actions performed in an access and process cycle, while the timing chart shows when these steps or actions are performed within the appropriate cycle.

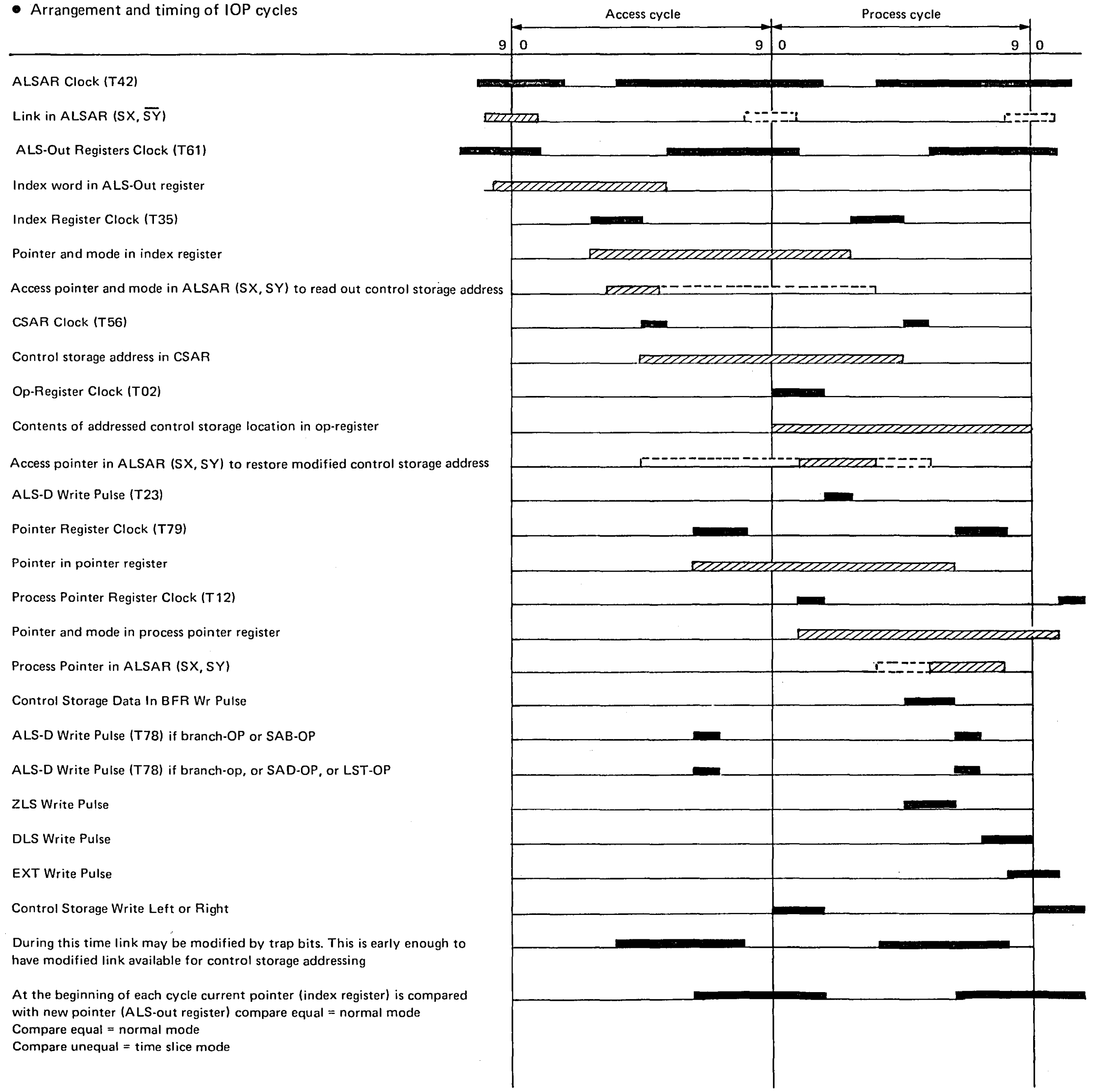
Details of process cycles for individual microinstructions are shown on the pages that follow.

See also pages 3-500 to 3-550 for microprogram control with and without time slicing and trapping.

- Actions during access and process cycles:



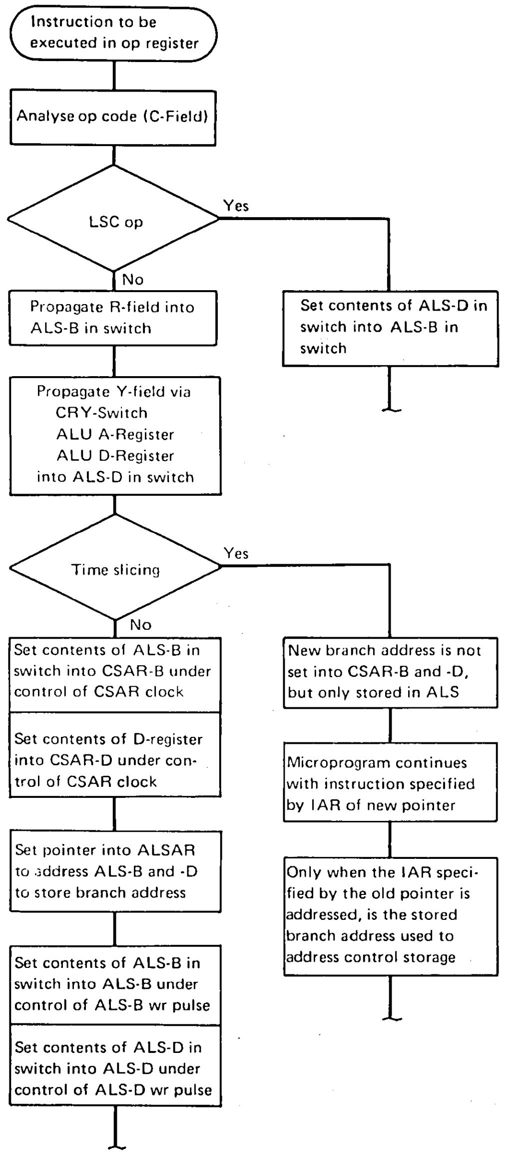
- Arrangement and timing of IOP cycles



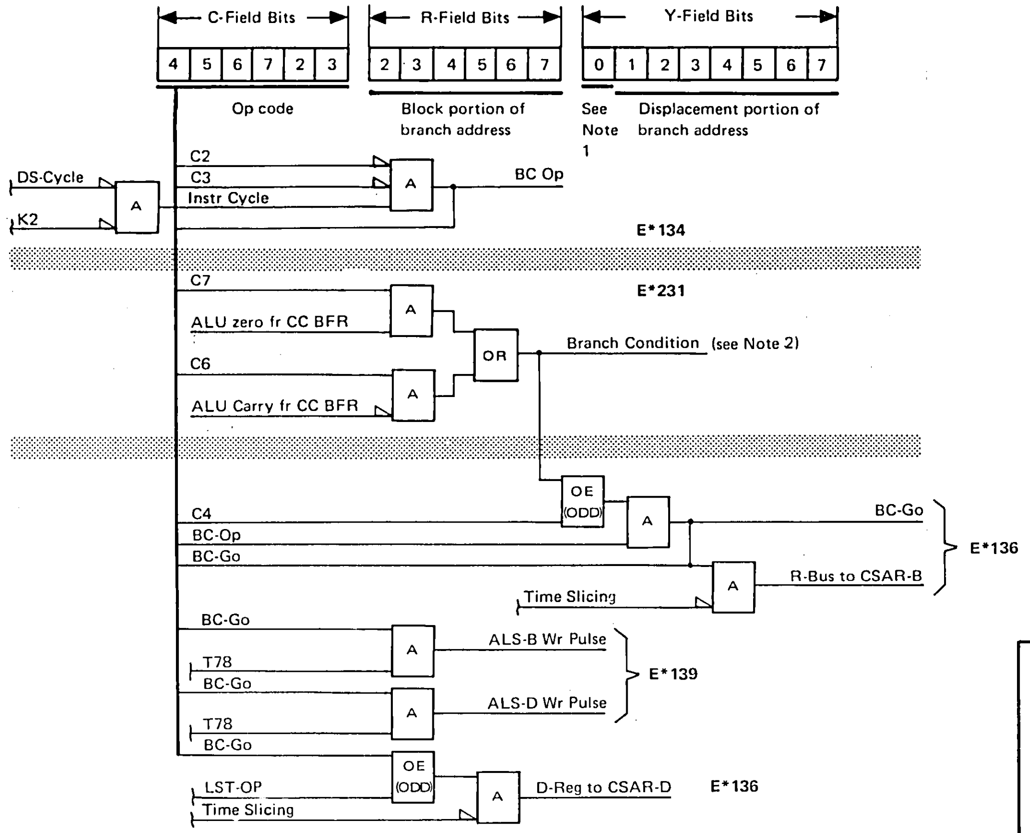
BNZ, BCY, BCN, BZ, BNC, BZN Process Cycle

- Group 1: Process cycle of BC (Branch on Condition) instructions. The following branch conditions are possible:

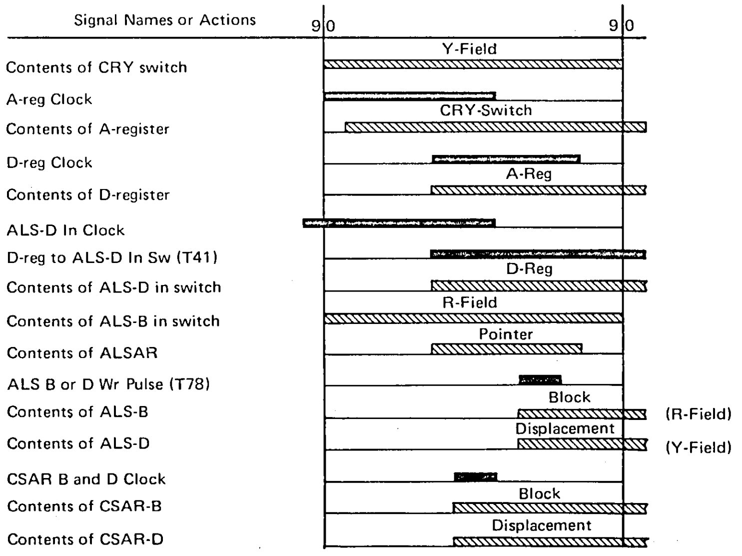
Mnemonic Op Code	C-Field Bits 4 5 6 7 2 3	Branch Condition
BNZ	0 0 0 1 0 0	Not zero
BCY	0 0 1 0 0 0	Carry
BCN	0 0 1 1 0 0	Carry and not zero
BZ	1 0 0 1 0 0	Zero
BNC	1 0 1 0 0 0	No carry
BZN	1 0 1 1 0 0	Zero or no carry



- Op-Register Layout



- Timing



- Control Signals for:

CRY Switch (E*10*) are:
CRY Switch SX 0 to 5 – Inactive
CRY Switch SX 6, 7 – Inactive
CRY Switch SY – Active

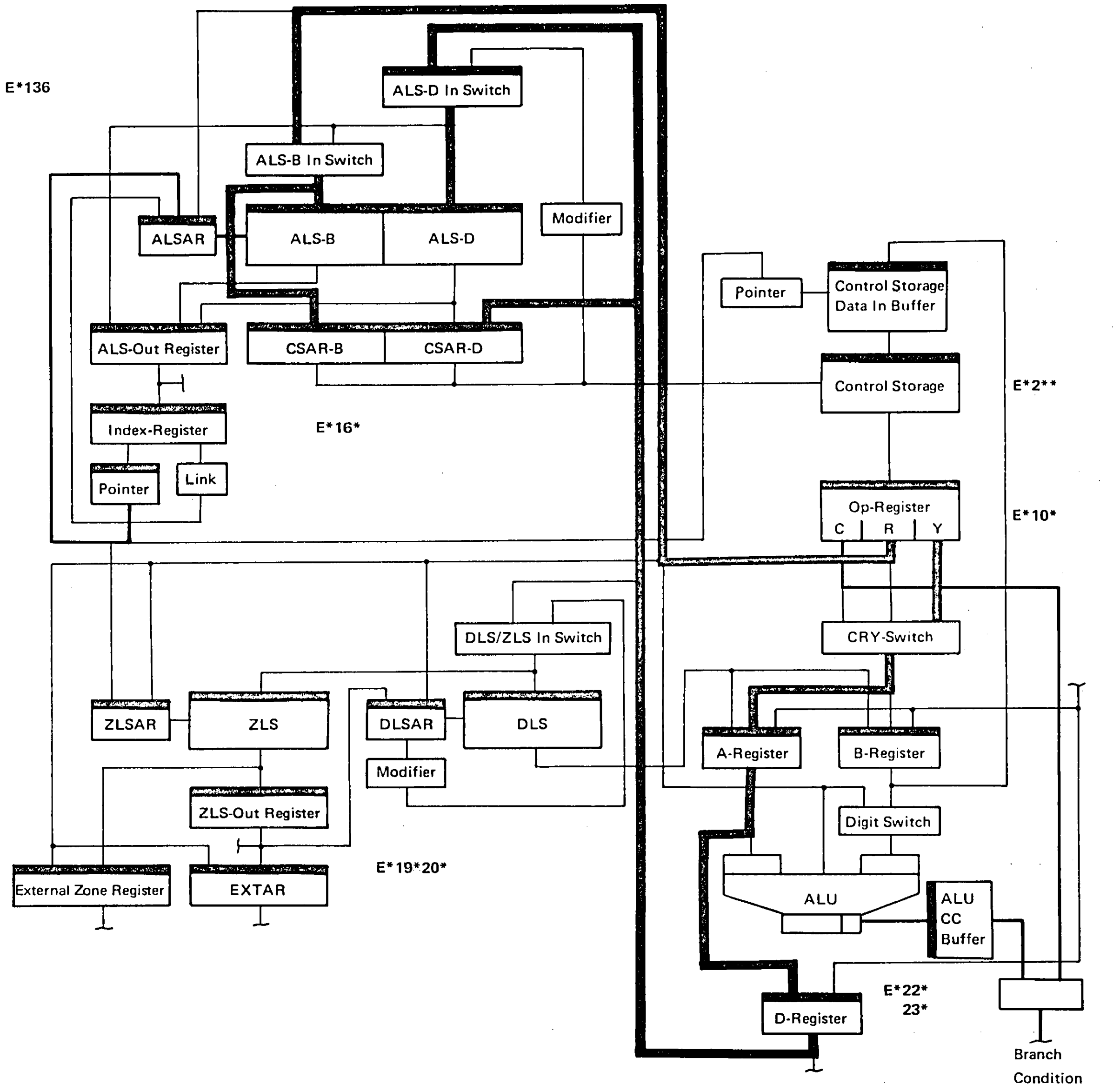
ALU A-Register (E*13*) are:
A-Reg SX – Active
A-Reg SY – Inactive

ALU D-Register (E*13*) are:
D-Reg SX – Inactive
D-Reg SY – Active

ALSAR (E*16*) are:
ALSAR SX – Inactive
ALSAR SY – Active.

Note 1: Suffix U bit; if on, it allows the changing of IARs in case of a successful branch.

Note 2: Active level of the line branch condition is plus.



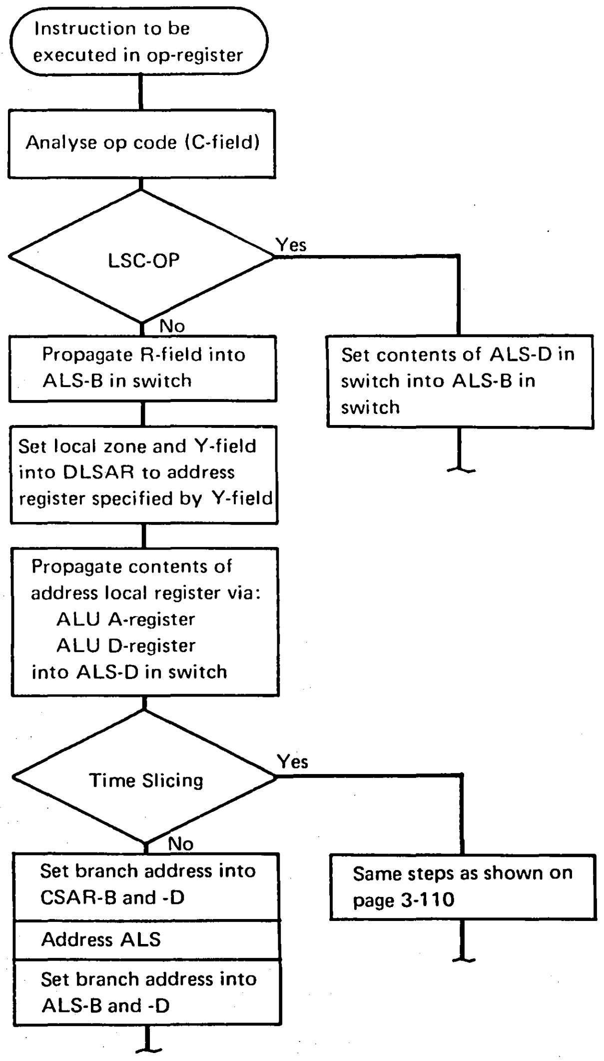
Microinstructions (continued)

BNZR, BCYR, BCNR, BZR, BNCR, BZNR Process Cycle

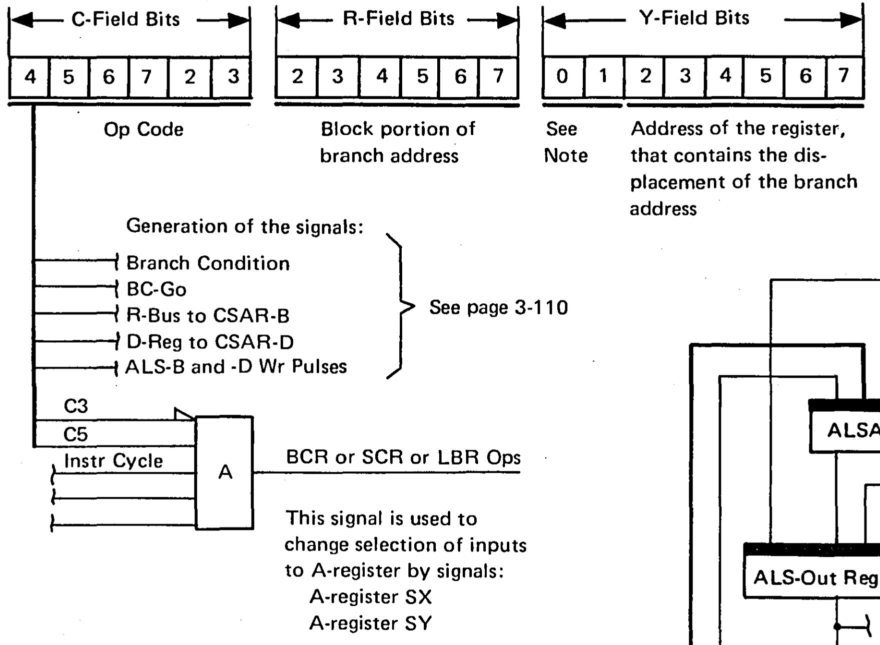
(Local register holds displacement portion of branch address)

- Group 1: Process cycle of BCR (Branch on Condition) instructions.
  The following branch conditions are possible:

Mnemonic Op Code	C-Field Bits 4 5 6 7 2 3	Branch Condition
BNZR	0 1 0 1 0 0	Not zero
BCYR	0 1 1 0 0 0	Carry
BCNR	0 1 1 1 0 0	Carry and not zero
BZR	1 1 0 1 0 0	Zero
BNCR	1 1 1 0 0 0	No carry
BZNR	1 1 1 1 0 0	Zero or no carry



- Op-Register Layout

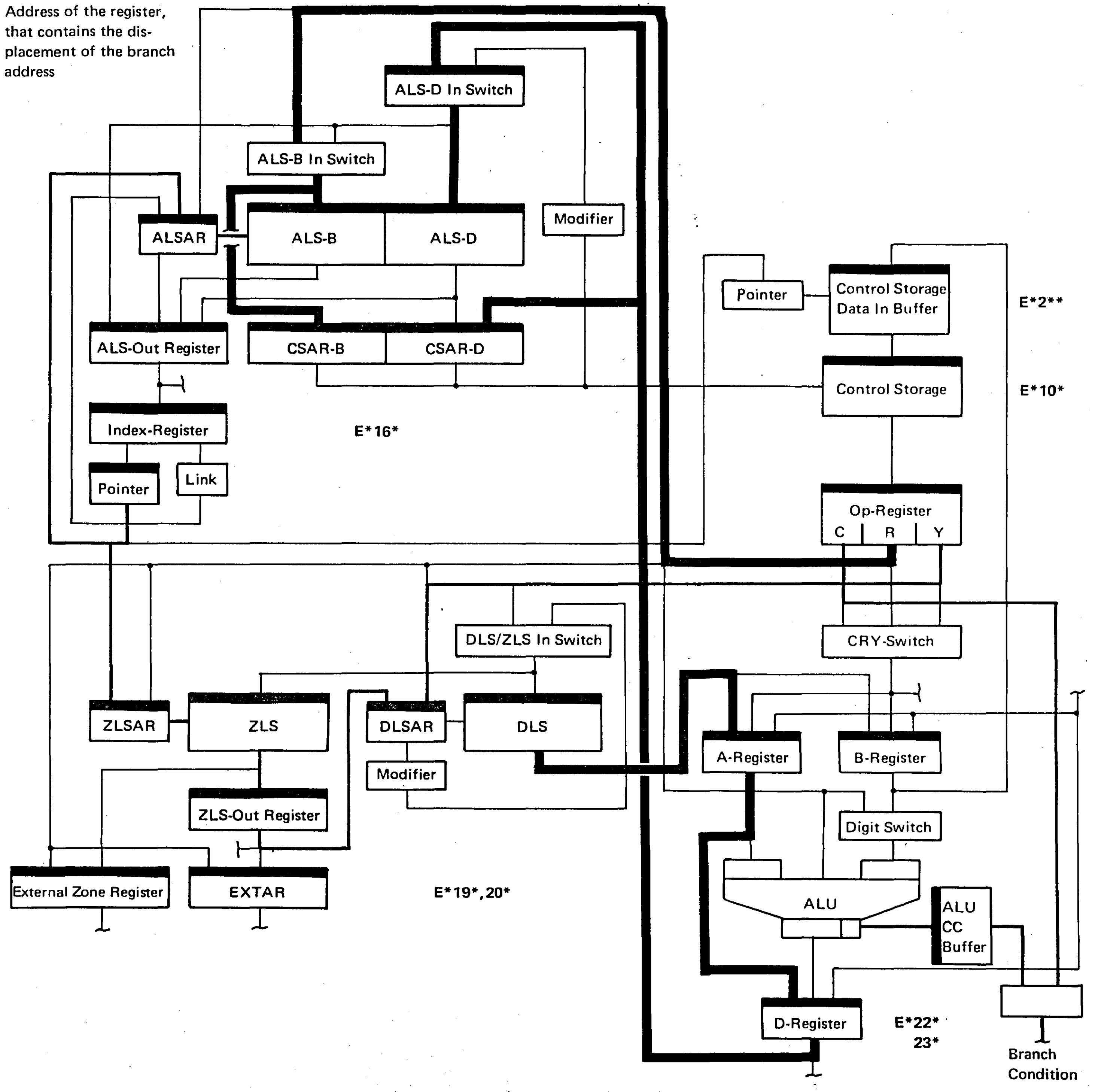


Note:
Y0 – Suffix U bit
Y1 – Has to be zero

- Timing (see page 3-110)
  Same as if Y-field represents displacement portion of branch address, except that A-register is loaded from DLS.

- Control Signals for:
  ALU A-Register (E*13*) are:
  A-Reg SX – Inactive
  A-Reg SY – Inactive

  ALU D-Register
  ALSAR
  See page 3-110.

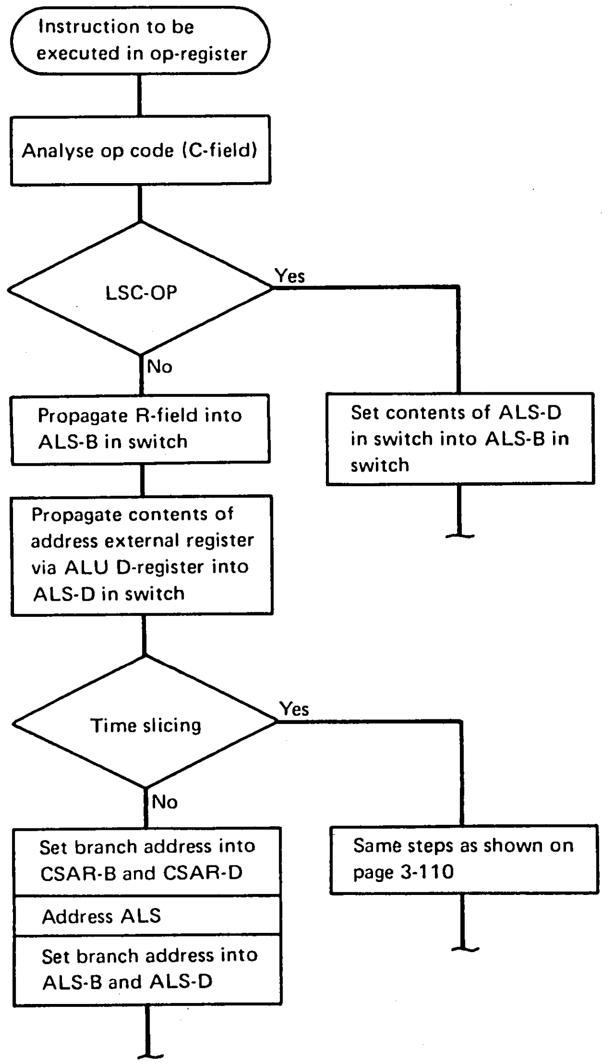


BNZR, BCYR, BCNR, BZR, BNCR, BZNR Process Cycle

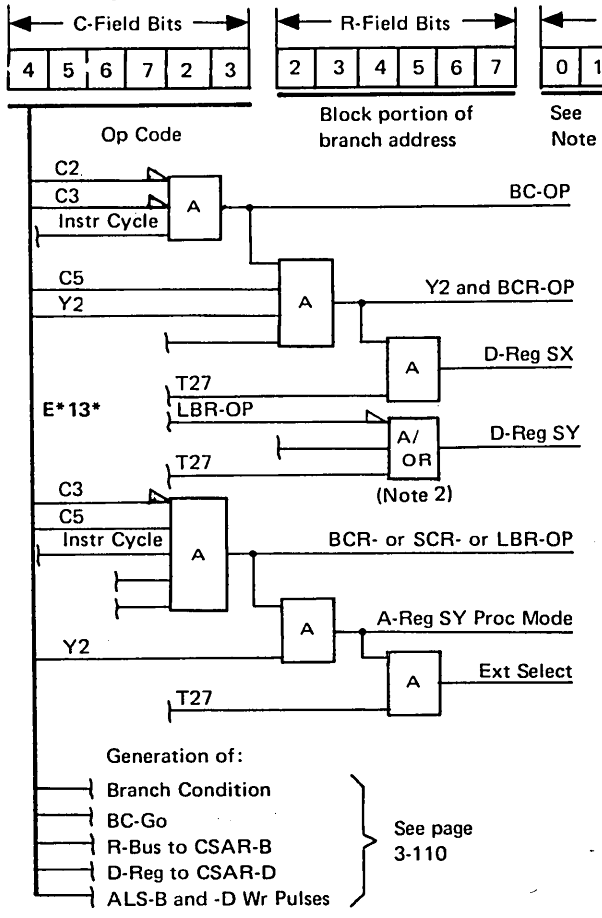
(External register holds displacement portion of branch address)

- Group 1: Process cycles of BCR (Branch on Condition) instructions.

See page 3-110 for branch conditions



- Op-Register Layout



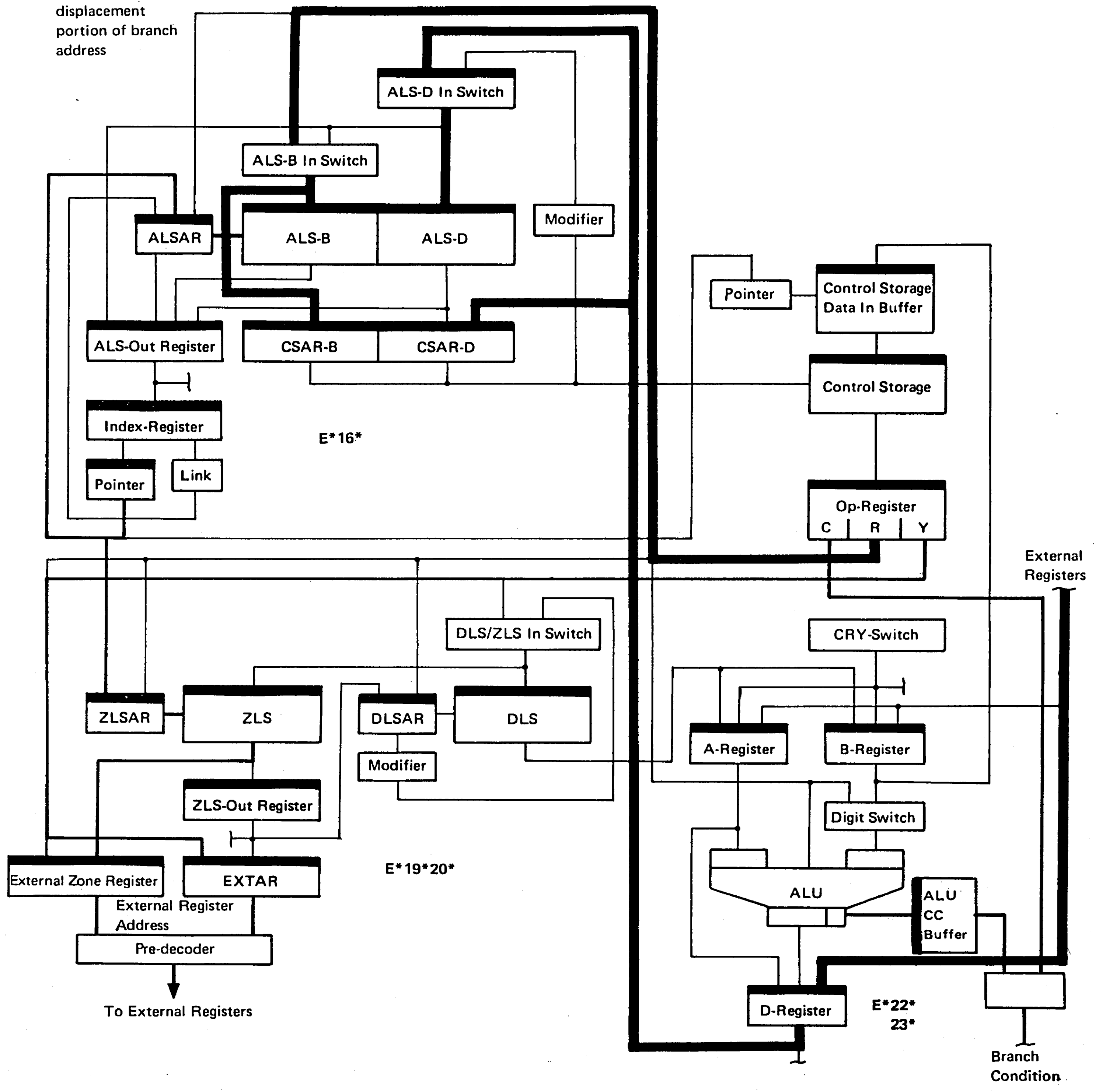
Notes:

1: Y0 = Suffix U bit
   Y1 = Has to be zero

2: This box represents a number of AND/OR blocks

- Timing (See page 3-110)
  Same as if Y-field represents displacement portion of branch address except that displacement from external register is directly set into D-register

- Control Signals for ALSAR. See page 3-110.

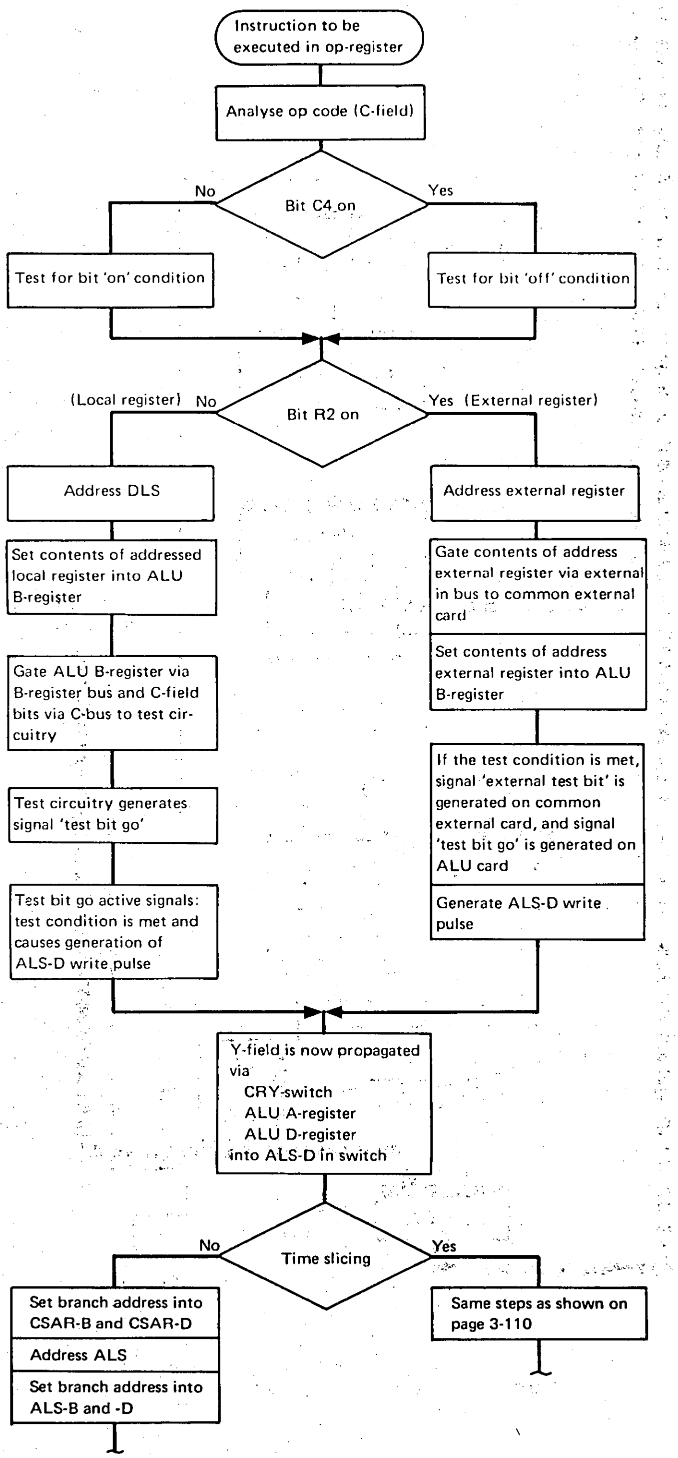


Microinstructions (continued)

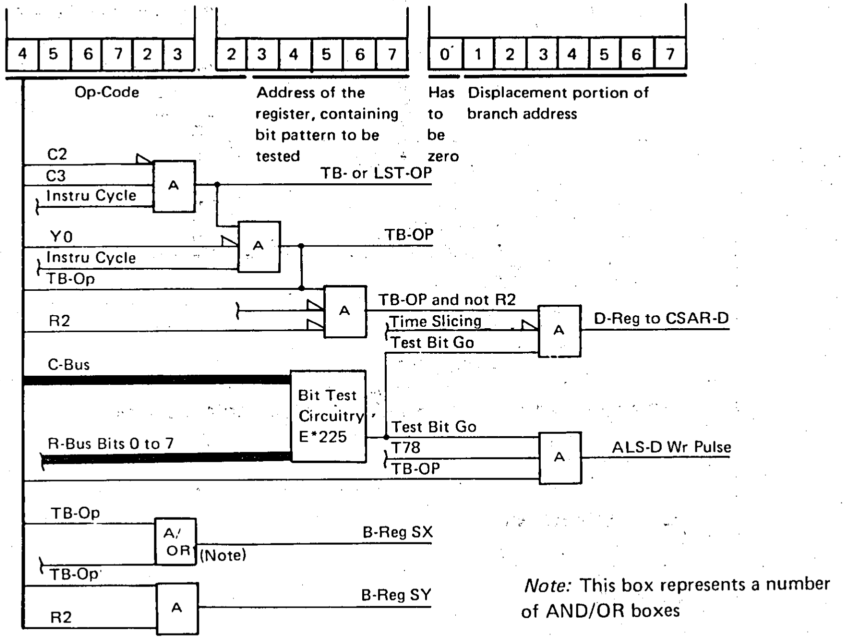
TB Process Cycle

(TB on bits in local registers).

- Group 1: Process cycle of TB (Test Bit and Branch) instructions
  Bit 'on' or bit 'off' conditions are tested, depending upon bit C4.
  If the condition to be tested is met, displacement portion (Y-Field) is stored

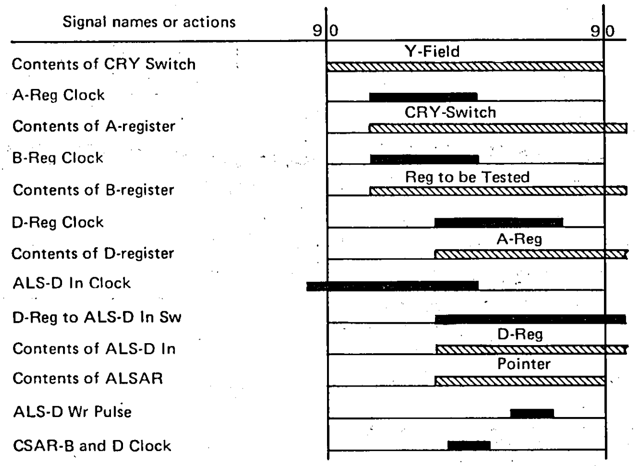


- Op-Register Layout and Control Signal Generation



Note: This box represents a number of AND/OR boxes

- Timing



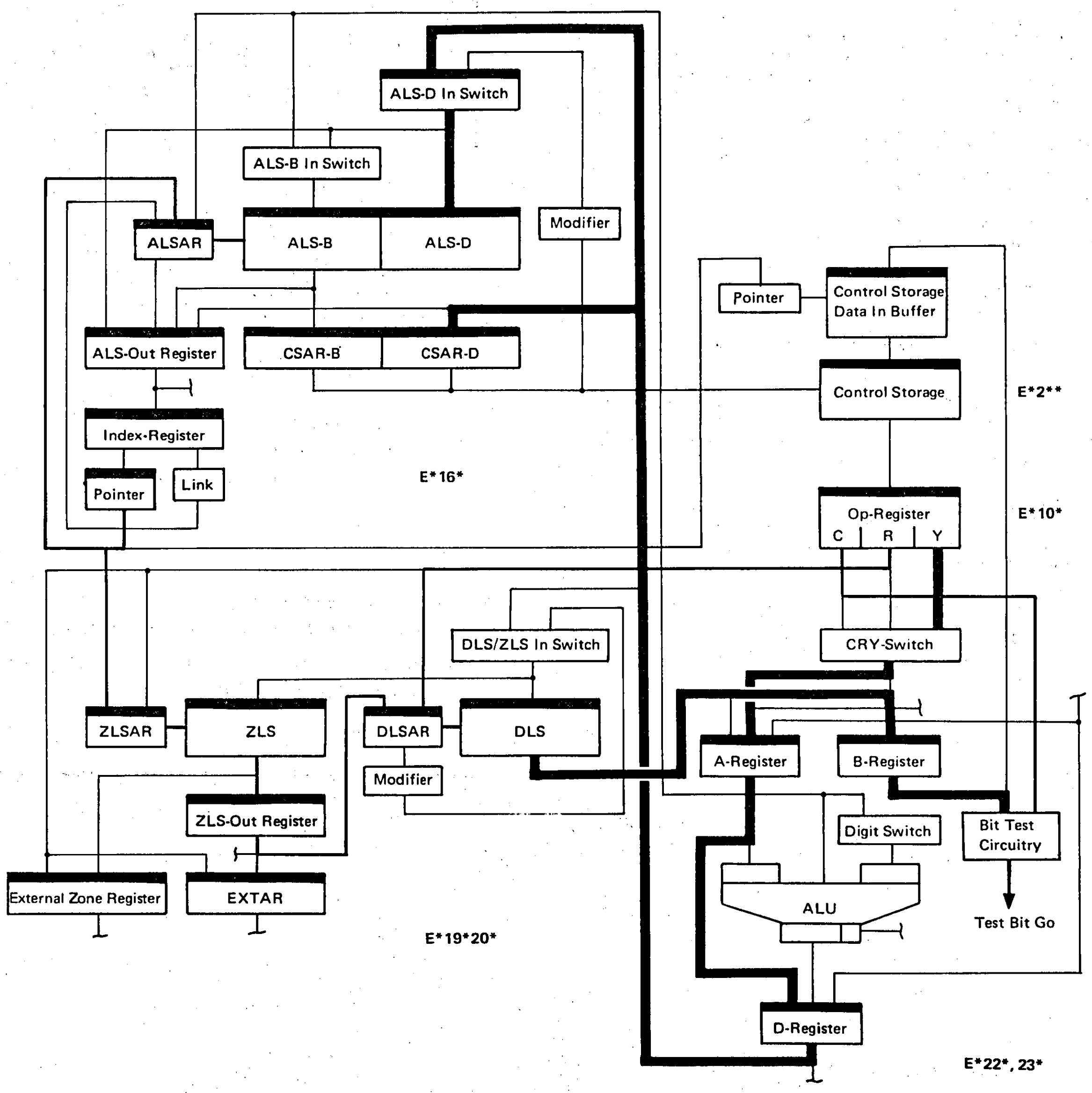
- Control Signals for:
  CRY Switch (E*10*) are:
  CRY Switch SX 0 to 5 – Inactive
  CRY Switch SX 6, 7 – Inactive
  CRY Switch SY – Active

  ALU A-Register (E*13*) are:
  A-Reg SX – Active
  A-Reg SY – Inactive

  ALU B-Register (E*13*) are:
  B-Reg SX – Active
  B-Reg SY – Inactive

  ALU D-Register (E*13*) are:
  D-Reg SX – Inactive
  D-Reg SY – Active

  ALSAR (E*16*) are:
  ALSAR SX – Inactive
  ALSAR SY – Active

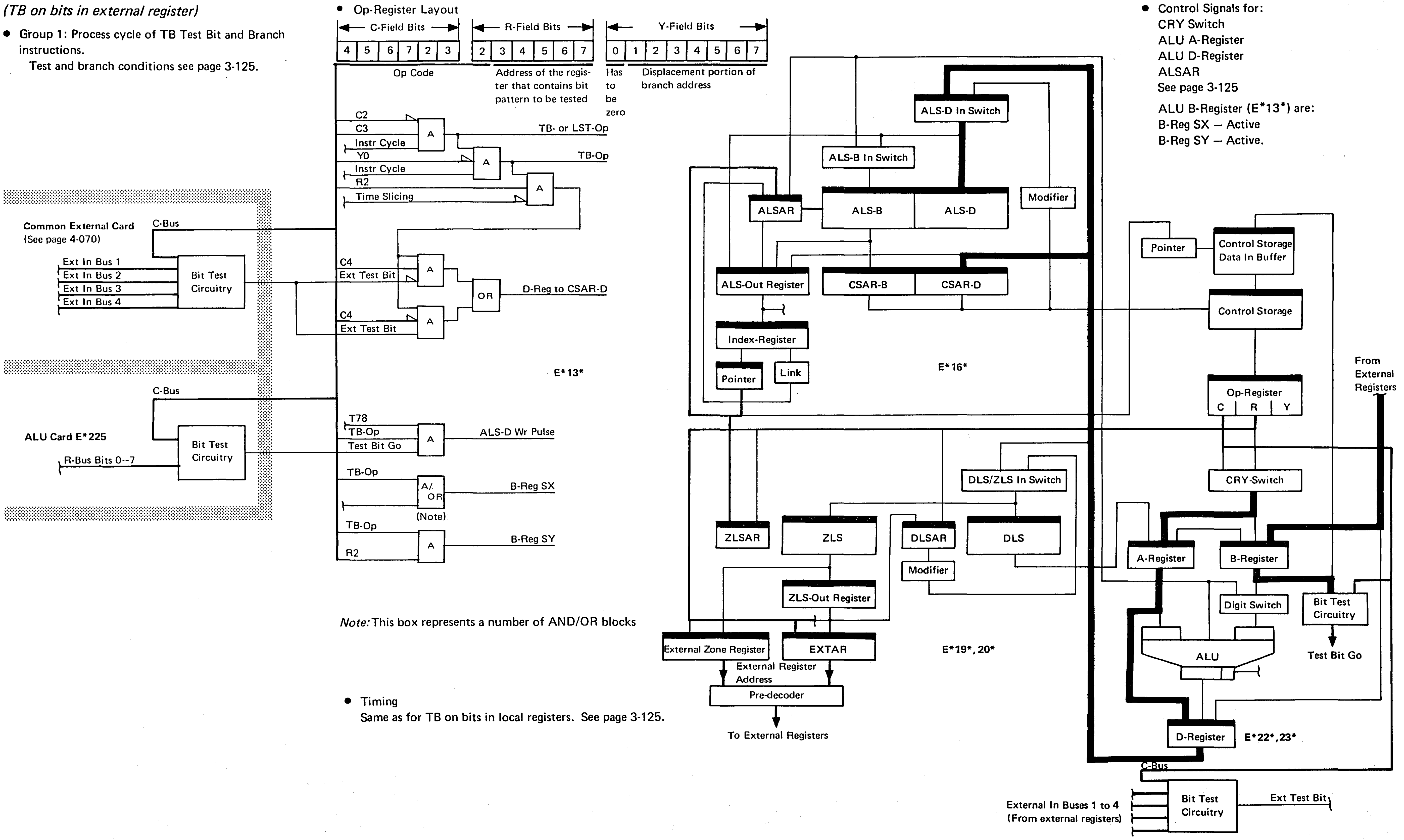


TB Process Cycle (continued):

(TB on bits in external register)

- Group 1: Process cycle of TB Test Bit and Branch instructions.
  Test and branch conditions see page 3-125.

- Control Signals for:
  CRY Switch
  ALU A-Register
  ALU D-Register
  ALSAR
  See page 3-125
  ALU B-Register (E*13*) are:
  B-Reg SX – Active
  B-Reg SY – Active.



Note: This box represents a number of AND/OR blocks

- Timing
  Same as for TB on bits in local registers. See page 3-125.

Microinstructions (continued)

Store (SDEC, SINC), Load (LDEC,LINC)

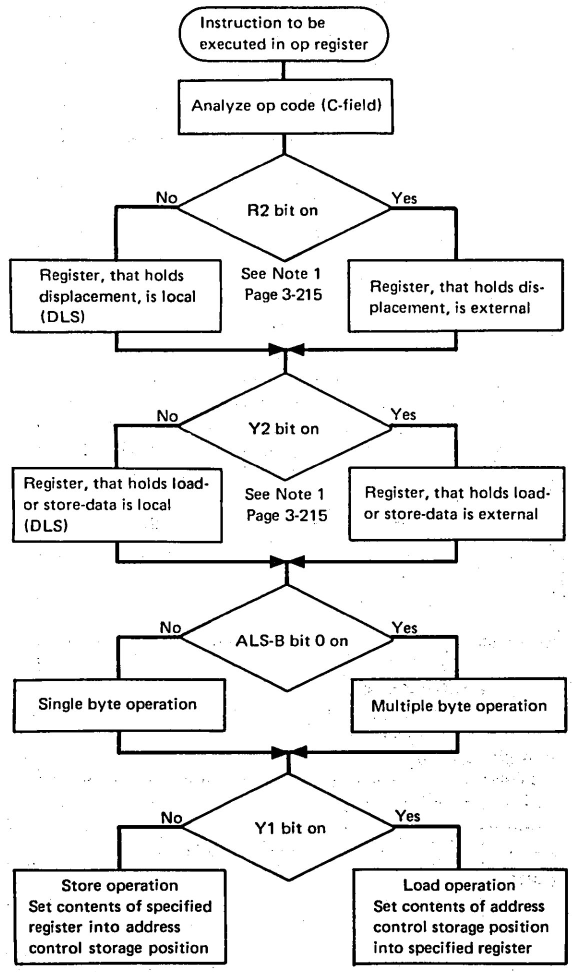
(Generation of control signals used during process cycles.)

- Group 2 Process Cycles of Store and Load
  - Store (data from register into control storage)
  - Load (data from control storage into register) instructions.

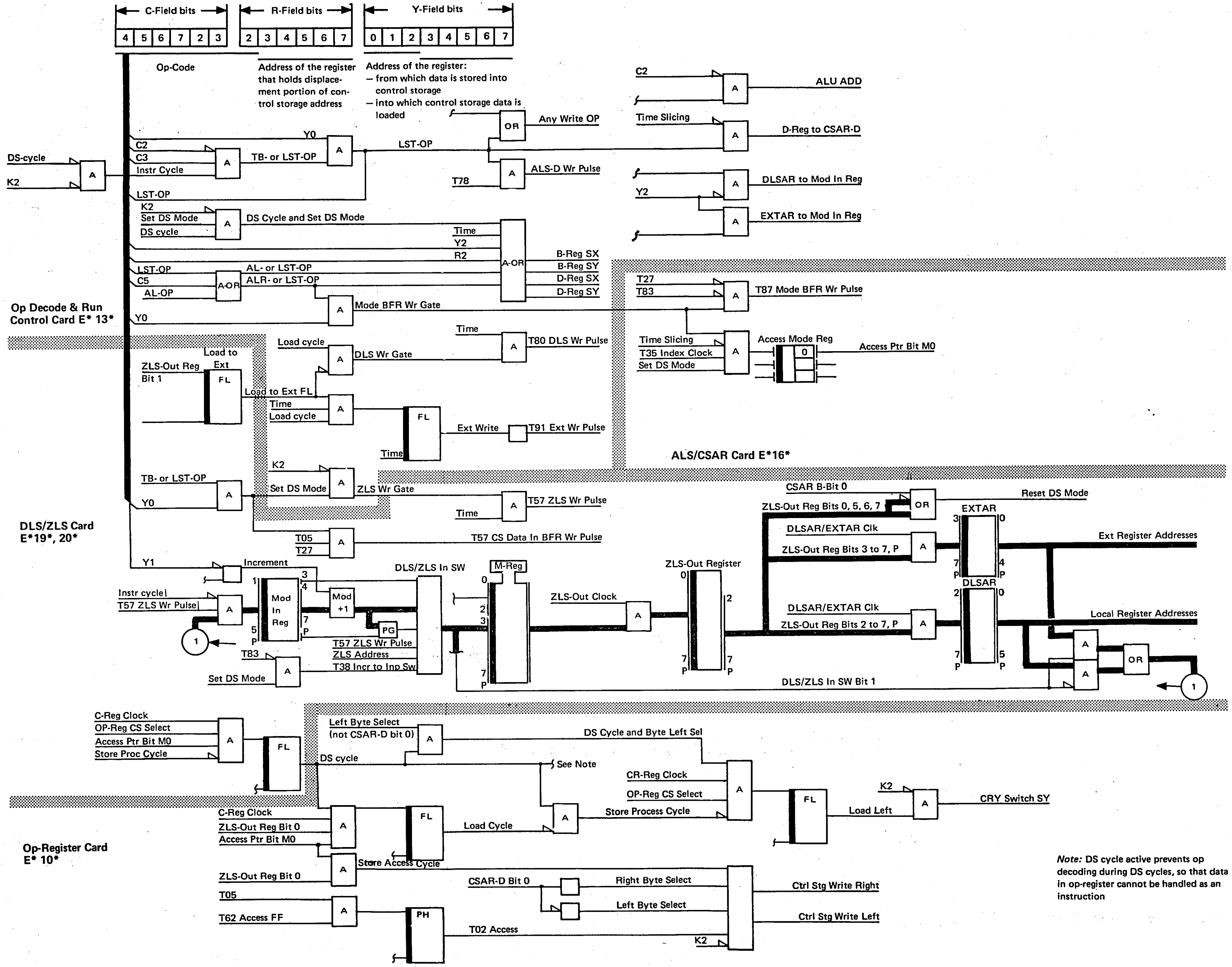
It is possible to increment or decrement, simultaneously, the contents of the register specified by the R-field. If so, the contents of that register may be modified from ±0 to ±8, as specified by the C-field bits 4 to 7. The result of this modification is restored into that register.

This modification has nothing to do with the modification of the register address, specified by the Y-field, during a multiple byte operation.

Modification of the displacement takes place in the first cycle following the access cycle of the instruction.



- Op-Register Layout



Note: DS cycle active prevents op decoding during DS cycles, so that data in op-register cannot be handled as an instruction

Store, Load Process Cycles

- Group 2:
  Store (single byte) operation
  Load (single byte) operation

The flowchart shows typical actions of Store process cycles and DS cycles.

All actions simultaneously performed by the IOP, but not belonging to the actual LST operation, are not shown.

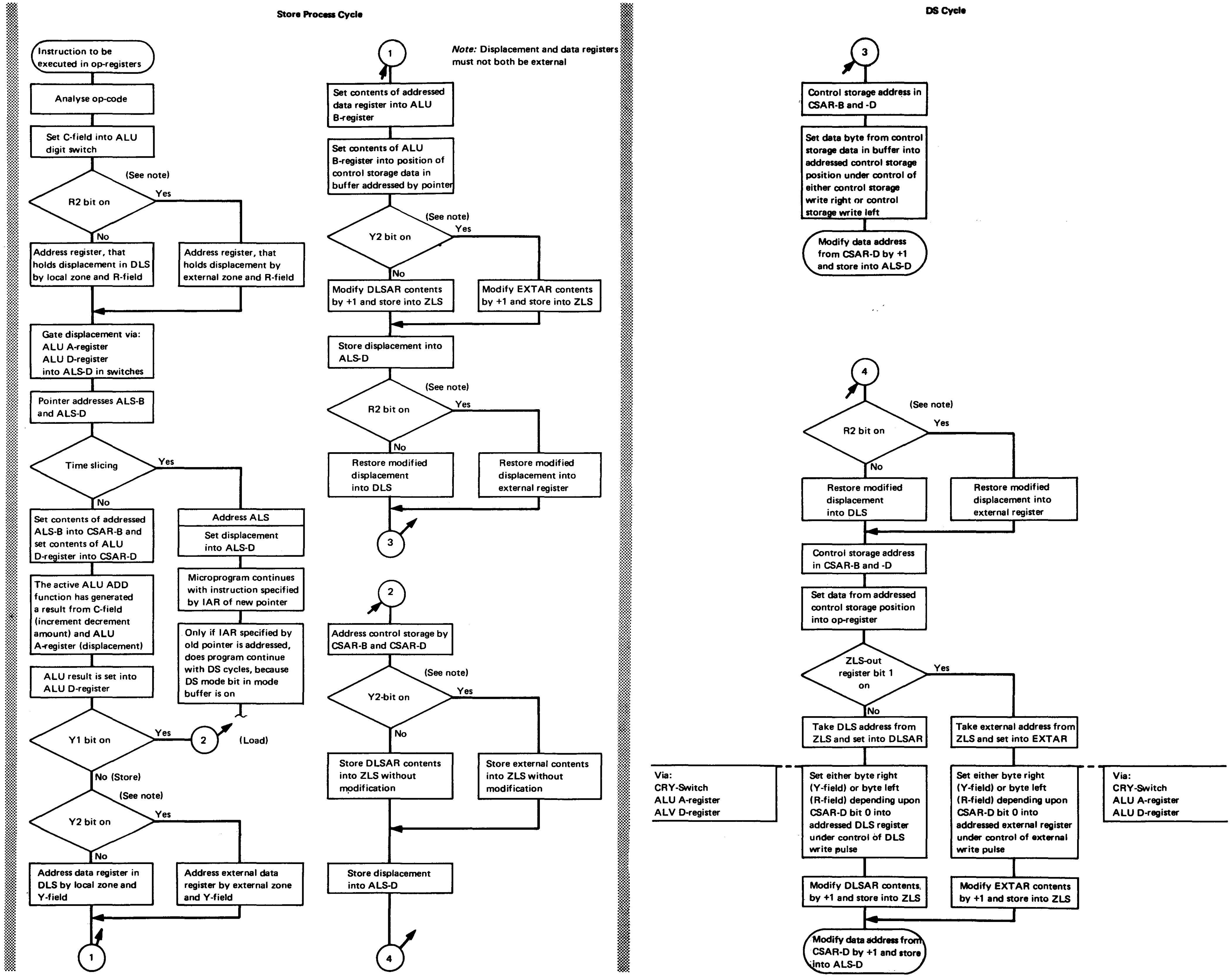
The cycle sequence is as follows:

Store Access Cycle	Store Process Cycle	DS-Cycle
Fetch store instruction	Set contents of register, specified by Y-field into control storage data in buffer	Set data from control storage data in buffer into control storage
Process previous instruction	Prepare control storage address	Fetch next instruction
Fetch load instruction	Prepare control storage address and address control storage location from which data will be taken	Set data from addressed control storage location into op-register
Process previous instruction		Address data register specified by Y-field
		Set data into addressed register
		Fetch next instruction

- Number of cycles, required for store and load operations are the number of bytes to be handled +1.
- References:
  - Store (single byte) operations are continued on Pages 3-220 to 3-230
  - Load (single byte) operations are continued on Pages 3-235 to 3-245
  - Multiple byte operations, see Pages 3-250 and 3-255
  - Op-register layout and
  - Generation of control signals see Page 3-210.

Store Process Cycle

DS Cycle



Microinstructions (continued)

Store

(Single byte, with both registers in DLS)

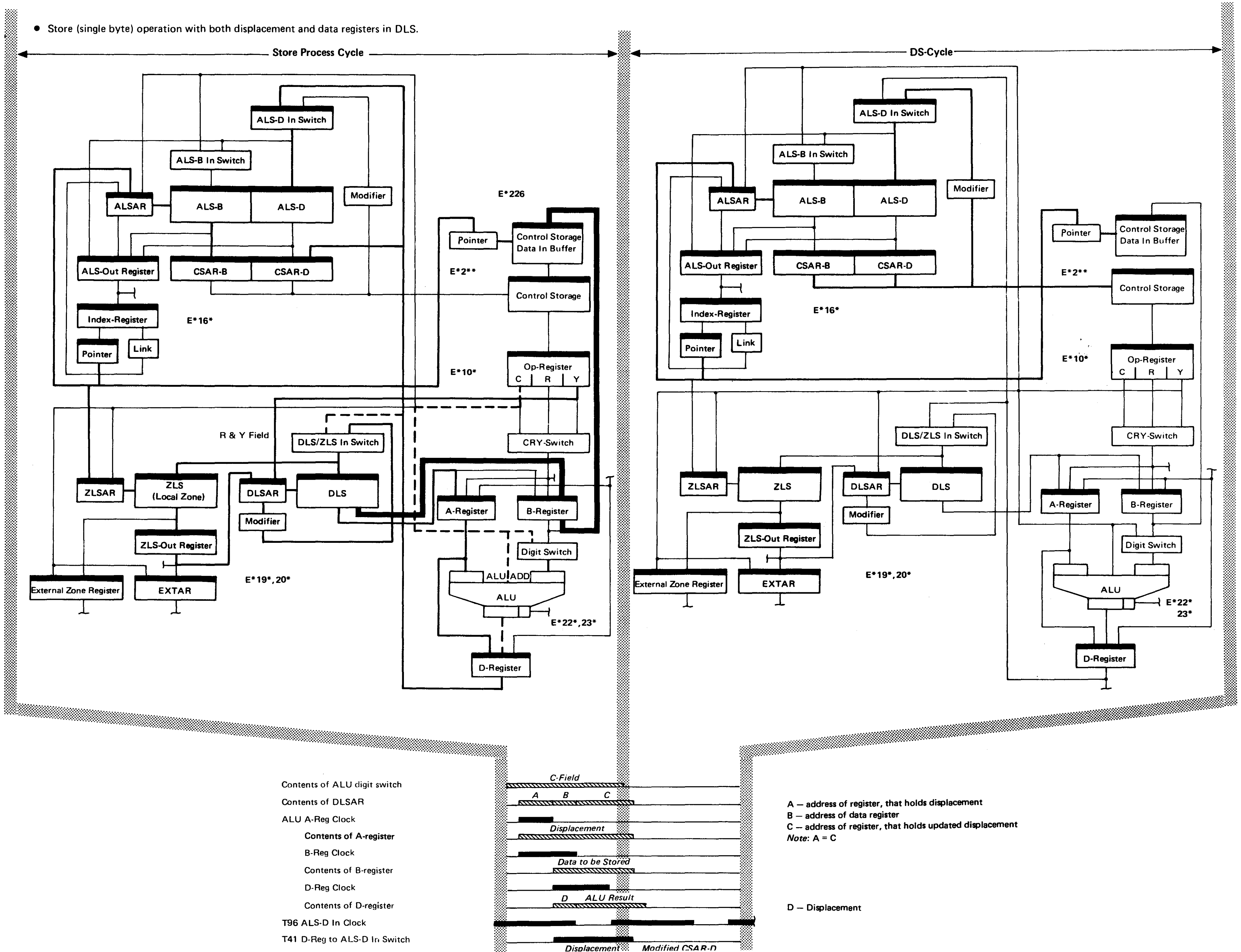
- Group 2: The cycles of a store (single byte) operation, with both displacement and data registers in DLS, are executed according to the following scheme:

Store Access Cycle	Store Process Cycle	DS-Cycle
Fetch instruction from control storage and set into op-register	**Analyze instruction:** R2 bit – off Y1 bit – off Y2 bit – off ALS-B bit 0 – off Prepare control storage address Add increment/decrement amount to displacement (see dotted line) Address register that contains data byte Set contents of data register into control storage data in buffer Modify data register address and store into ZLS	Set data from control storage data in buffer into addressed control storage position under control of 'control storage write right' or 'control storage write left'
Process cycle of previous instruction, store modified address into ALS (MIAR or SIAR)	Store displacement into ALS-D Restore modified displacement into DLS ALU controls: DLS to A-register A-register SX = 0 A-register SY = 0 A-register to D-register D-register SX = 0 D-register SY = 1 Result to D-register D-register SX = 1 D-register SY = 0 DLS to B-register B-register SX = 1 B-register SY = 0	Store modified data address from CSAR-D into ALS-D Access cycle of next instruction

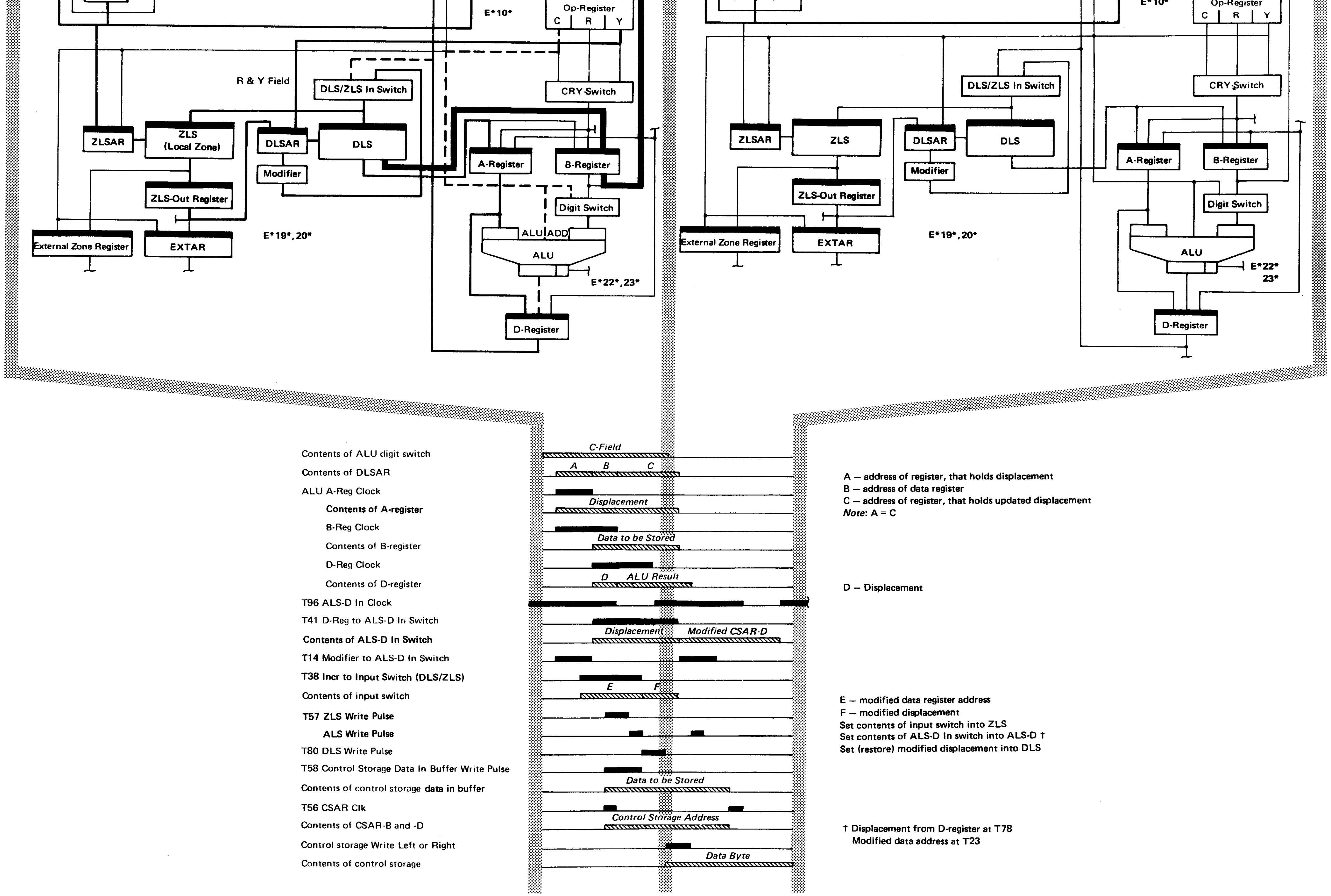
- References:

Subject	Pages
Load (single byte) ops	3-235 to 3-245
Multiple byte ops	3-250, 3-255
Op-register layout	3-210
Generation of control signals	3-210
Store/load flowchart	3-215

- Store (single byte) operation with both displacement and data registers in DLS.



Process cycle of previous instruction, store modified address into ALS (MIAR or SIAR)	Store displacement into ALS-D Restore modified displacement into DLS ALU controls: DLS to A-register A-register SX = 0 A-register SY = 0 A-register to D-register D-register SX = 0 D-register SY = 1 Result to D-register D-register SX = 1 D-register SY = 0 DLS to B-register B-register SX = 1 B-register SY = 0	Store modified data address from CSAR-D into ALS-D Access cycle of next instruction



A – address of register, that holds displacement
B – address of data register
C – address of register, that holds updated displacement
Note: A = C

D – Displacement

E – modified data register address
F – modified displacement
Set contents of input switch into ZLS
Set contents of ALS-D In switch into ALS-D †
Set (restore) modified displacement into DLS

† Displacement from D-register at T78
Modified data address at T23

- References:

Subject	Pages
Load (single byte) ops	3-235 to 3-245
Multiple byte ops	3-250, 3-255
Op-register layout	3-210
Generation of control signals	3-210
Store/load flowchart	3-215

Store

(Single byte with external data register and displacement in DLS)

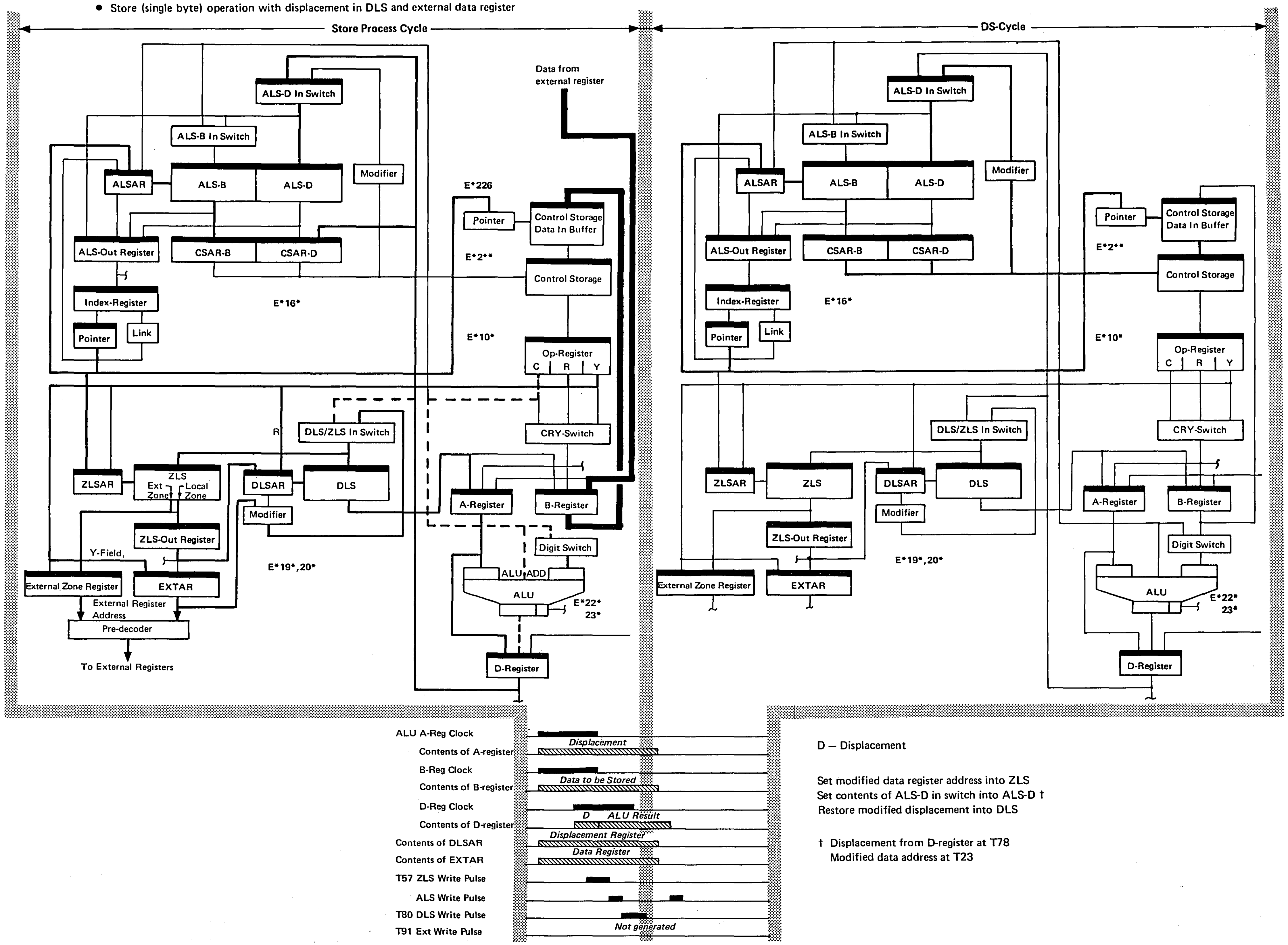
- Group 2: The cycles of a store (single byte) operation, with external data register and displacement in DLS, are executed according to the following scheme:

Store Access Cycle	Store Process Cycle	DS-Cycle
Fetch instruction from control storage and set into op-register	Analyse instruction: R2 bit – off Y1 bit – off Y2 bit – on ALS-B bit 0 – off Prepare control storage address Add increment/ decrement amount to displacement (see dotted line) Address register that contains data byte Set contents of data register into control storage data in buffer Modify data register address and store into ZLS	Set data from control storage data in buffer into addressed control storage position under control of either 'control storage write right' or 'control storage write left'
Process cycle of previous instruction, store modified address into ALS (MIAR or SIAR)	Store displacement into ALS-D Restore modified displacement into DLS ALU controls: DLS to A-Register A-Register to D-Register Result to D-Register see page 3-220 External to B-Register B-Register SX = 1 B-Register SY = 1	Store modified data address from CSAR-D into ALS-D Access cycle of next instruction

- References:

Subject	Pages
Load (single byte) ops	3-235 to 3-245
Multiple byte ops	3-250, 3-255
Op register layout	3-210
Generation of control signals	3-210
Store/load flowchart	3-215

- Store (single byte) operation with displacement in DLS and external data register



D – Displacement

Set modified data register address into ZLS
Set contents of ALS-D in switch into ALS-D †
Restore modified displacement into DLS

† Displacement from D-register at T78
Modified data address at T23

Microinstructions (continued)

Store

(Single byte with data register in DLS and displacement in external register)

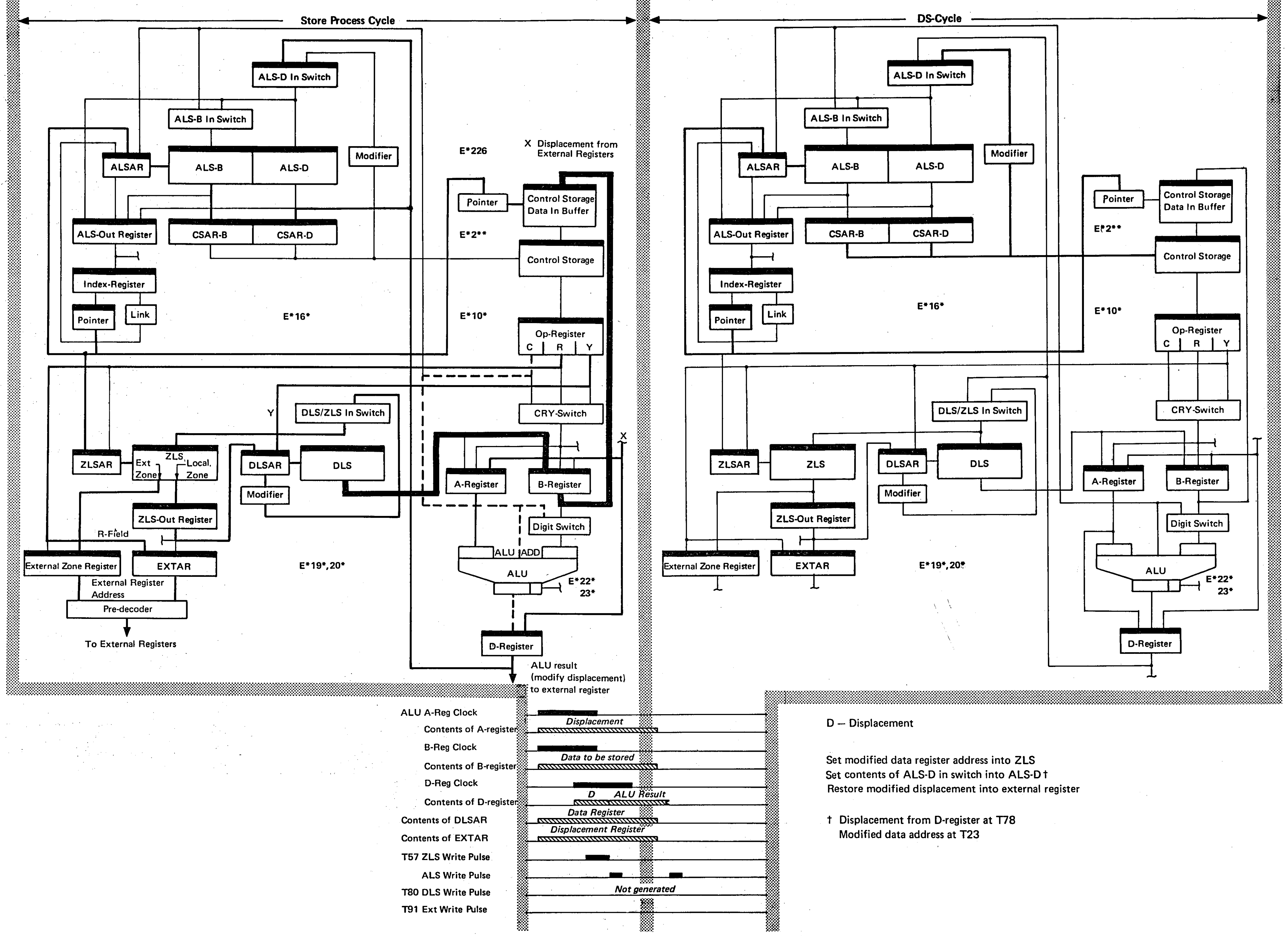
- Group 2: The cycles of a store (single byte) operation, with displacement in external register and data register in DLS, are executed according to the following scheme:

Store Access Cycle	Store Process Cycle	DS-Cycle
Fetch instruction from control storage and set into Op register	Analyse instruction R2 bit – on Y1 bit – off Y2 bit – off ALS-B bit 0 – off Prepare control storage address Add increment/ decrement amount to displacement (see dotted line) Address register that contains data byte Set contents of data register into control storage data in buffer Modify data register address and store into ZLS	Set data from control storage data in buffer into addressed control storage position under control of either 'control storage write right' or 'control storage write left'
Process cycle of previous instruction, store modified address into ALS (MIAR or SIAR)	Store displacement into ALS-D Restore modified displacement into external register ALU controls: External to A-Register A-register SX = 0 A-register SY = 1 External to D-register D-register SX = 1 D-register SY = 1 Result to D-register DLS to B-register see page 3-220	Store modified data address from CSAR-D into ALS-D Access cycle of next instruction

- References:

Subject	Pages
Load (single byte) ops	3-235 to 3-245
Multiple byte ops	3-250, 3-255
Op-register layout	3-210
Generation of control signals	3-210
Store/load flowchart	3-215

- Store (single byte) operation with displacement in external register data register in DLS



D – Displacement

Set modified data register address into ZLS
Set contents of ALS-D in switch into ALS-D†
Restore modified displacement into external register

† Displacement from D-register at T78
Modified data address at T23

Load

(Single byte with both registers in DLS)

- Group 2: The cycles of a load (single byte) operation, with both displacement and data registers in DLS, are executed according to the following scheme:

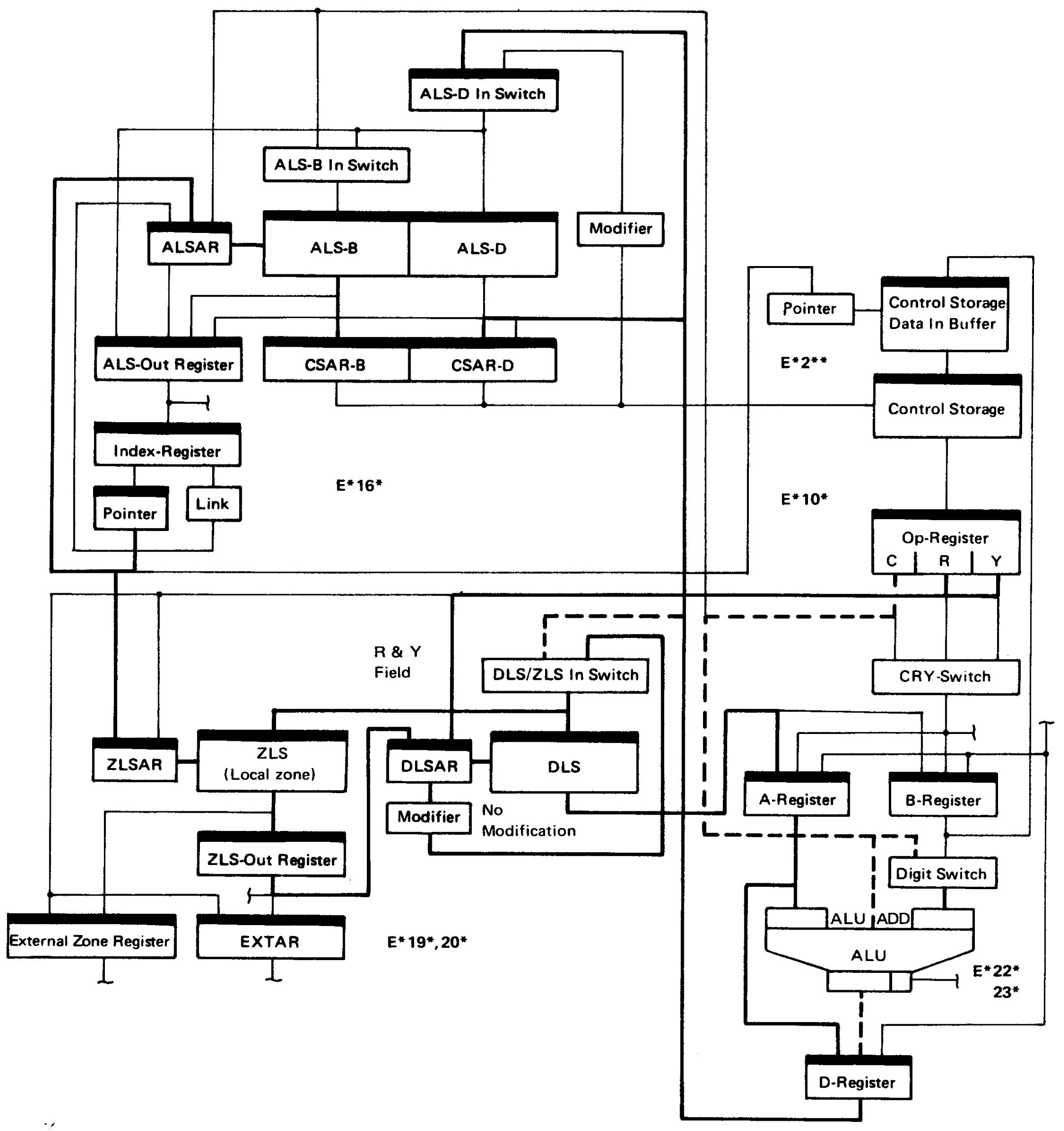
Store Access Cycle	Store Process Cycle	DS-Cycle
Fetch instruction from control storage and set into op-register	Analyse instruction R2 bit – off Y1 bit – on Y2 bit – off ALS-B bit 0 – off Prepare control storage address	Set data from address control storage location into op-register Address data register
	Add increment/decrement amount to displacement (see dotted line) Set control storage address into CSAR-B and CSAR-D	Set either byte right (Y) or byte left (R) depending upon CSAR-D bit 0 into addressed data register under control of respective write pulse
	Store data register address into ZLS (modification is suppressed by bit Y1 – on)	Modify data register address and store into ZLS
Process cycle of previous instruction, store modified address into ALS (MIAR or SIAR)	Store displacement into ALS-D Restore modified displacement into DLS	Store modified data address into ALS-D Access cycle of next instruction
	ALU controls: DLS to A-register A-register SX = 0 A-register SY = 0 A-register to D-register D-register SX = 0 D-register SY = 1 ALU result to D-register D-register SX = 1 D-register SY = 0	ALU controls: CRY switch to A-register A-register SX = 1 A-register SY = 0 A-register to D-register D-register SX = 0 D-register SY = 1

- References

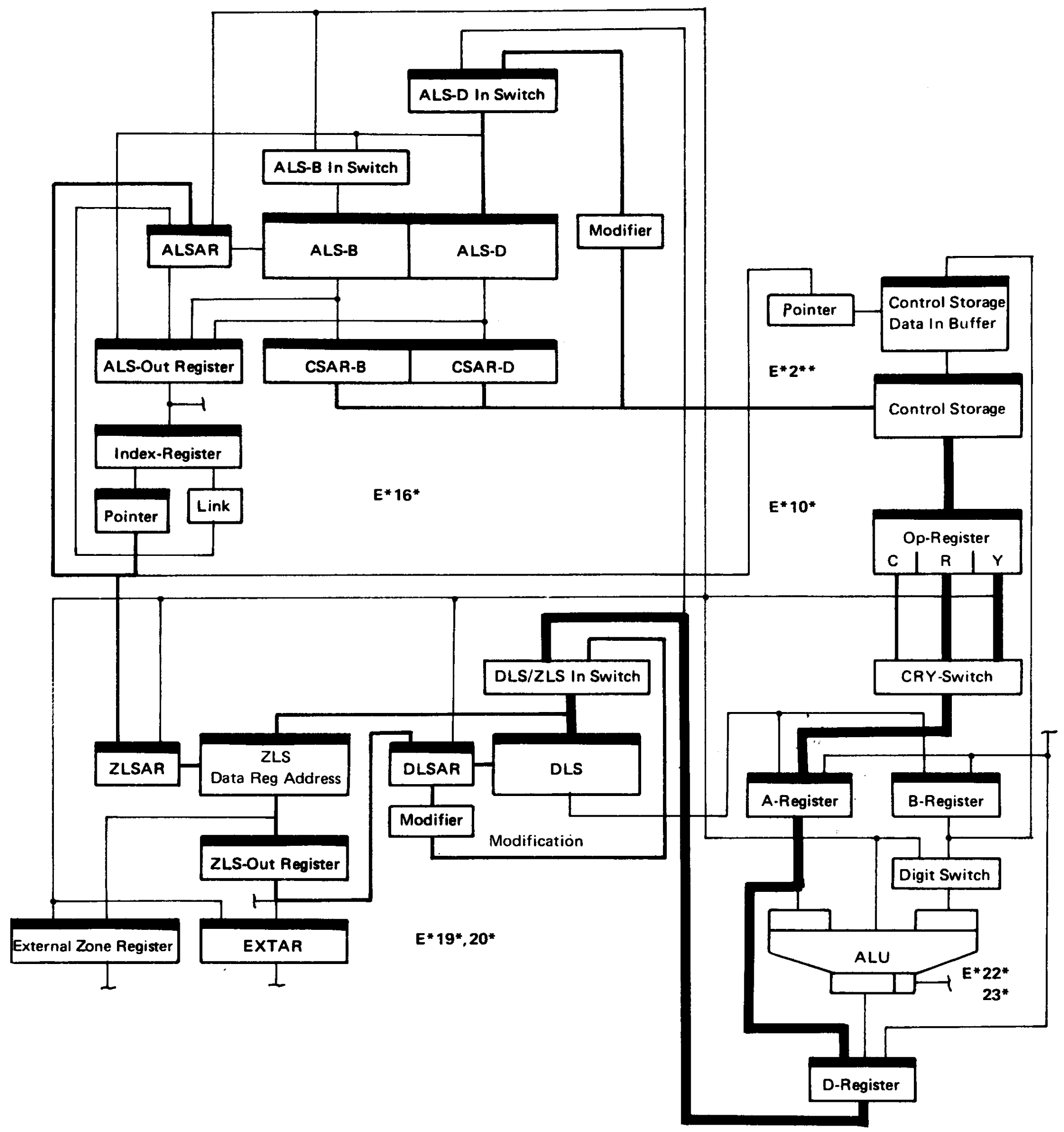
Subject	Pages
Store (single byte) ops	3-220 to 3-230
Multiple byte ops	3-250, 3-255
Op-register layout	3-210
Generation of control signals	3-210
Store/load flowchart	3-215

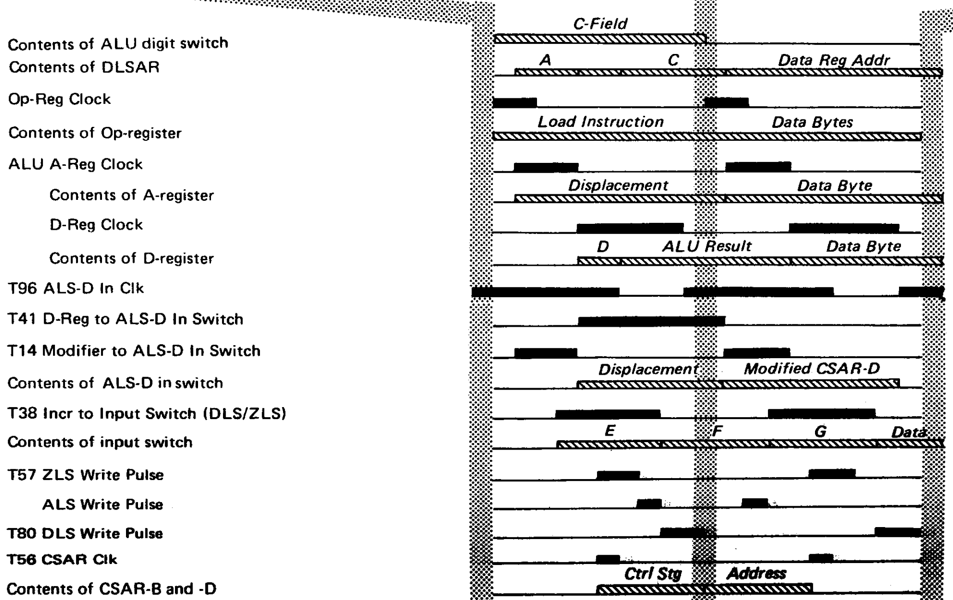
- Load (single byte) operation with both displacement and data registers in DLS.

Store Process Cycle



DS-Cycle





A – Address of register, that holds desplacement
C – Address of register, that holds updated displacement
Note: A = C

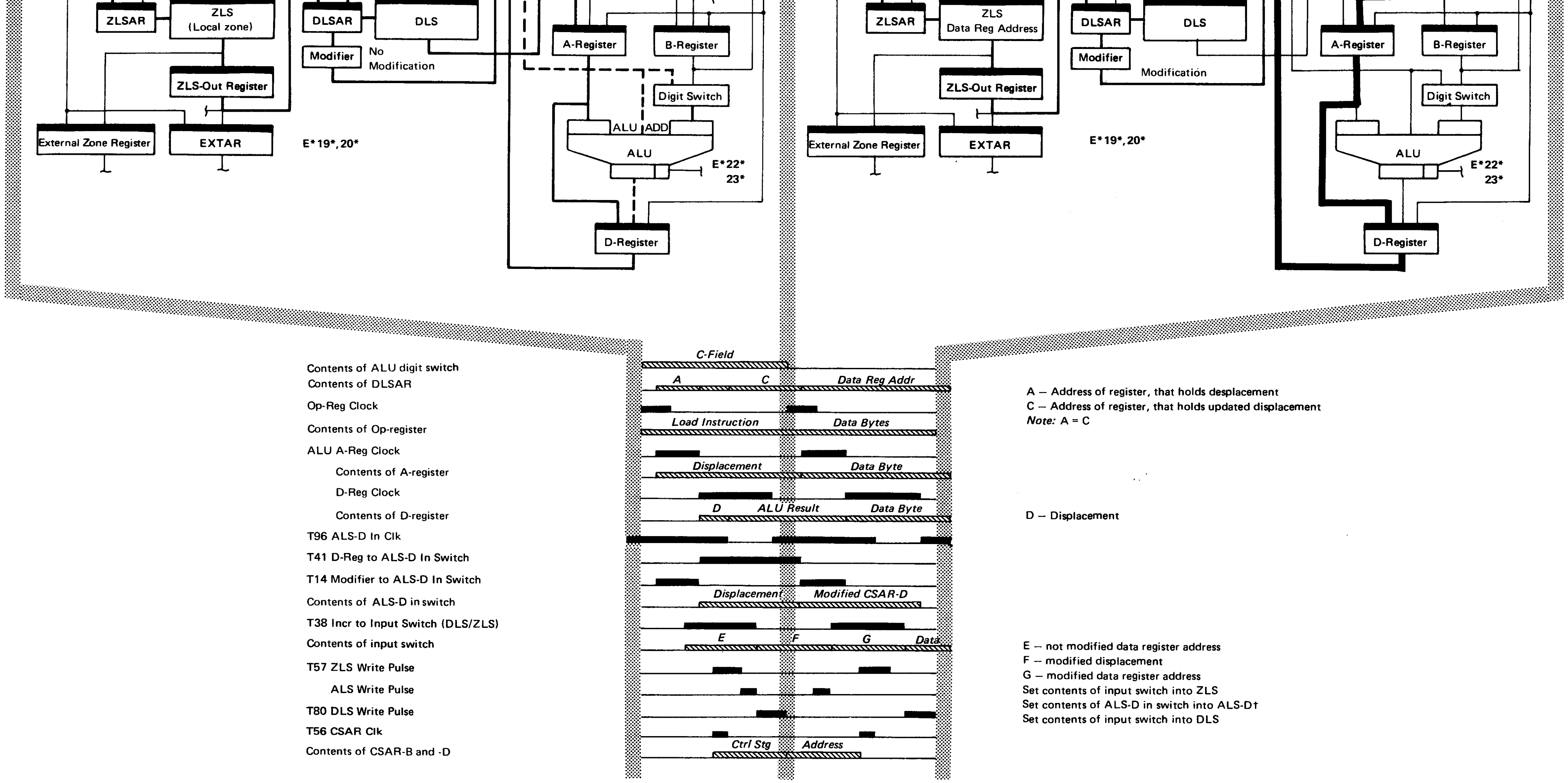
D – Displacement

E – not modified data register address
F – modified displacement
G – modified data register address
Set contents of input switch into ZLS
Set contents of ALS-D in switch into ALS-D†
Set contents of input switch into DLS

	A-register to D-register D-register SX = 0 D-register SY = 1 ALU result to D-register D-register SX = 1 D-register SY = 0	A-register SY = 0 A-register to D-register D-register SX = 0 D-register SY = 1

- References

Subject	Pages
Store (single byte) ops	3-220 to 3-230
Multiple byte ops	3-250, 3-255
Op-register layout	3-210
Generation of control signals	3-210
Store/load flowchart	3-215



A – Address of register, that holds displacement
C – Address of register, that holds updated displacement
Note: A = C

D – Displacement

E – not modified data register address
F – modified displacement
G – modified data register address
Set contents of input switch into ZLS
Set contents of ALS-D in switch into ALS-D†
Set contents of input switch into DLS

† Displacement from D-register at T78
Modified data address at T23

Microinstructions (continued)

Load

(Single byte with data external register and displacement in DLS)

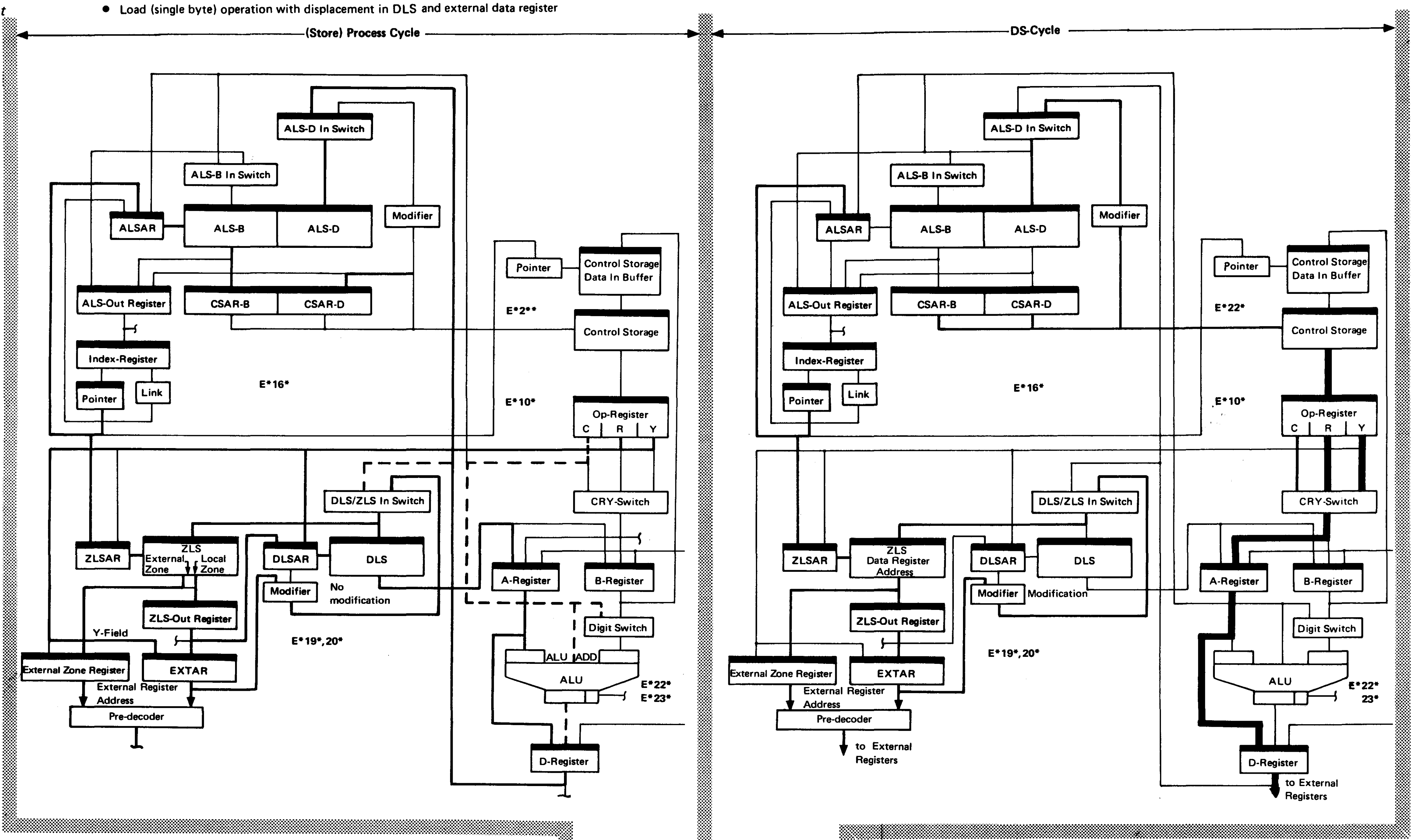
- Group 2: The cycles of a load (single byte) operation, with external data register and displacement in DLS,are executed according to the following scheme:

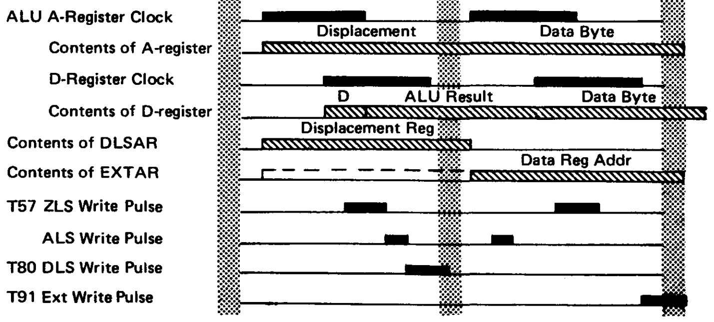
(Store) Access Cycle	(Store) Process Cycle	DS-Cycle
Fetch instruction from control storage and set into op-register	Analyse instruction R2 bit -- off Y1 bit – on Y2 bit -- on ALS-B bit 0 - off	Set data from addressed control storage position into Op-register Address data register
	Prepare control storage address	Set either byte right (Y-field) or byte left (R-field) depending upon CSAR-D bit 0 into addressed data register under control of respective write pulse
	ADD increment/ decrement amount to displacement. See dotted line	
	Set control storage address into CSAR-B and CSAR-D	Modify data register address and store into ZLS
	Store data register address into ZLS (modification is suppressed by bit Y1 – on)	Store modified data address into ALS-D (from CSAR-D) Access cycle of next instruction
Process cycle of previous instruction, store modified address into ALS-D (MIAR or SIAR)	Store displacement into ALS-D	ALU-controls: CRY Switch to A-register A-register to D-register see page 3-235
	Restore modified displacement into DLS	
	ALU controls: DLS to A-register A-register to D-register Result to D-register see page 3-235	

- References

Subject	Pages
Store (single byte) ops	3-220 to 3-230
Multiple byte ops	3-250, 3-255
Op-register layout	3-210
Generation of control signals	3-210
Store/load flowchart	3-215

- Load (single byte) operation with displacement in DLS and external data register





D – Displacement

Set contents of input switch (DLS/ZLS) into ZLS
Set contents of ALS-D in switch into ALS-D †
Restore modified displacement into DLS
Set data byte into addressed external register

† Displacement from D-register at T78
Modified data address at T23

Load

(Single byte with data register in DLS and displacement in external register)

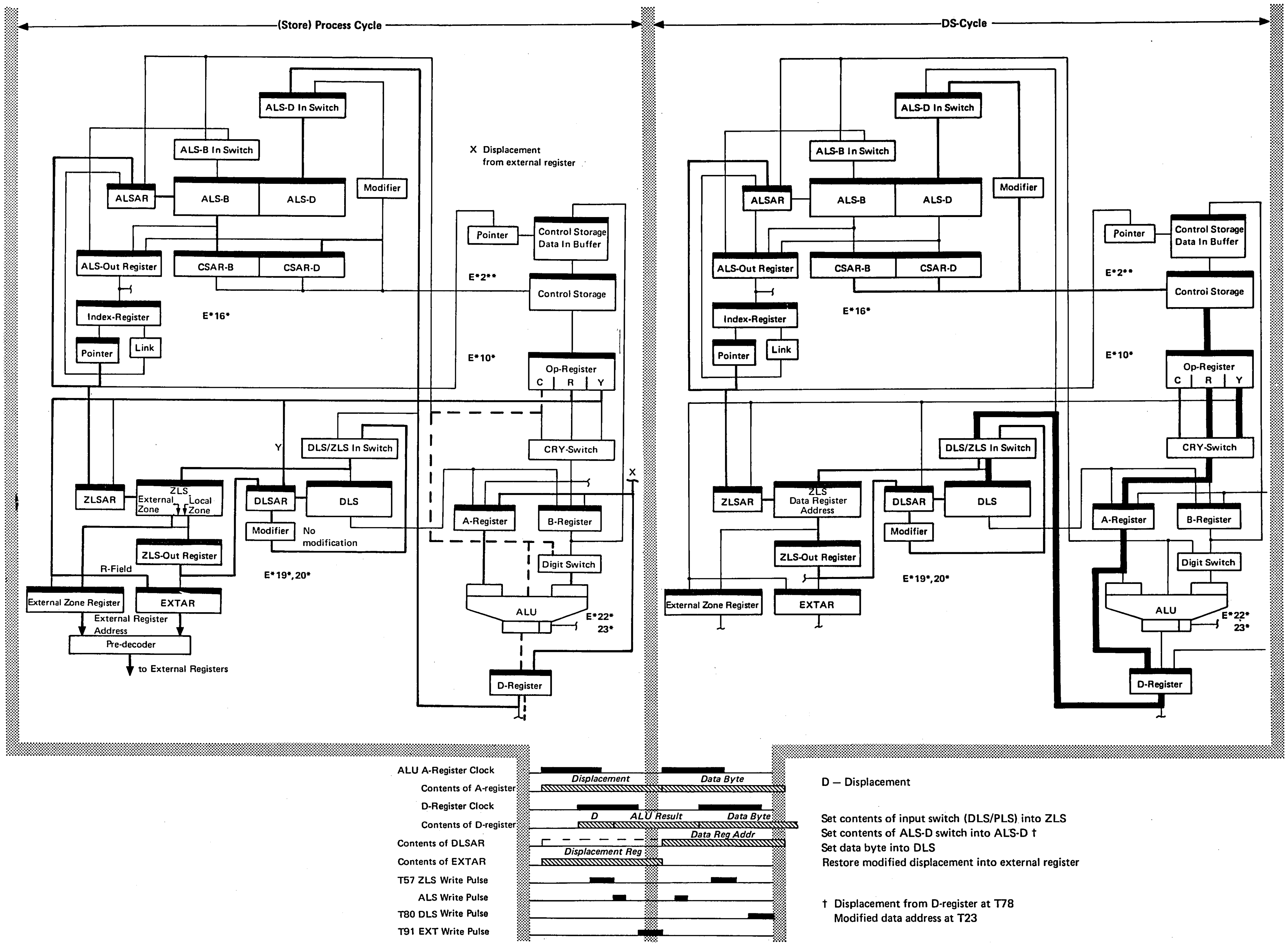
- Group 2: The cycles of a load (single byte) operation with displacement in external register and data register in DLS, are executed according to the following scheme:

(Store) Access Cycle	(Store) Process Cycle	DS-Cycle
Fetch instruction from control storage and set into Op-register	Analyse instruction R2 bit – on Y1 bit – on Y2 bit – off ALS-B bit 0 – off Prepare control storage address Add increment/ decrement amount to displacement (see dotted line) Set control storage address into CSAR-B and CSAR-D Store data register address into ZLS (modification is suppressed by bit Y1 – on)	Set data from addressed control storage location into op register Address data register Set either byte right (Y) or byte left (R-field) depending upon CSAR-D bit 0 into addressed data register under control of respective write pulse Modify data register address and store into ZLS Store modified data address into ALS-D (from CSAR-D)
Process cycle of previous instruction, store modified address into ALS (MIAR or SIAR)	Store displacement into ALS-D Restore modified displacement into external register ALU controls: External to A-Register A-Register SX A-Register SY External to D-Register D-Register SX D-Register SY Result to D-Register see page 3-235	Access cycle of next instruction ALU-controls: CRY Switch to A-Register A-Register to D-Register see page 3-235

- References

Subject	Pages
Store (single byte) ops	3-220 to 3-230
Multiple byte ops	3-250, 3-255
Op-register layout	3-210
Generation of control signals	3-210
Store/load flowchart	3-215

- Load (single byte) operation with displacement in external register and data register in DLS



D – Displacement

Set contents of input switch (DLS/PLS) into ZLS
Set contents of ALS-D switch into ALS-D †
Set data byte into DLS
Restore modified displacement into external register

† Displacement from D-register at T78
Modified data address at T23

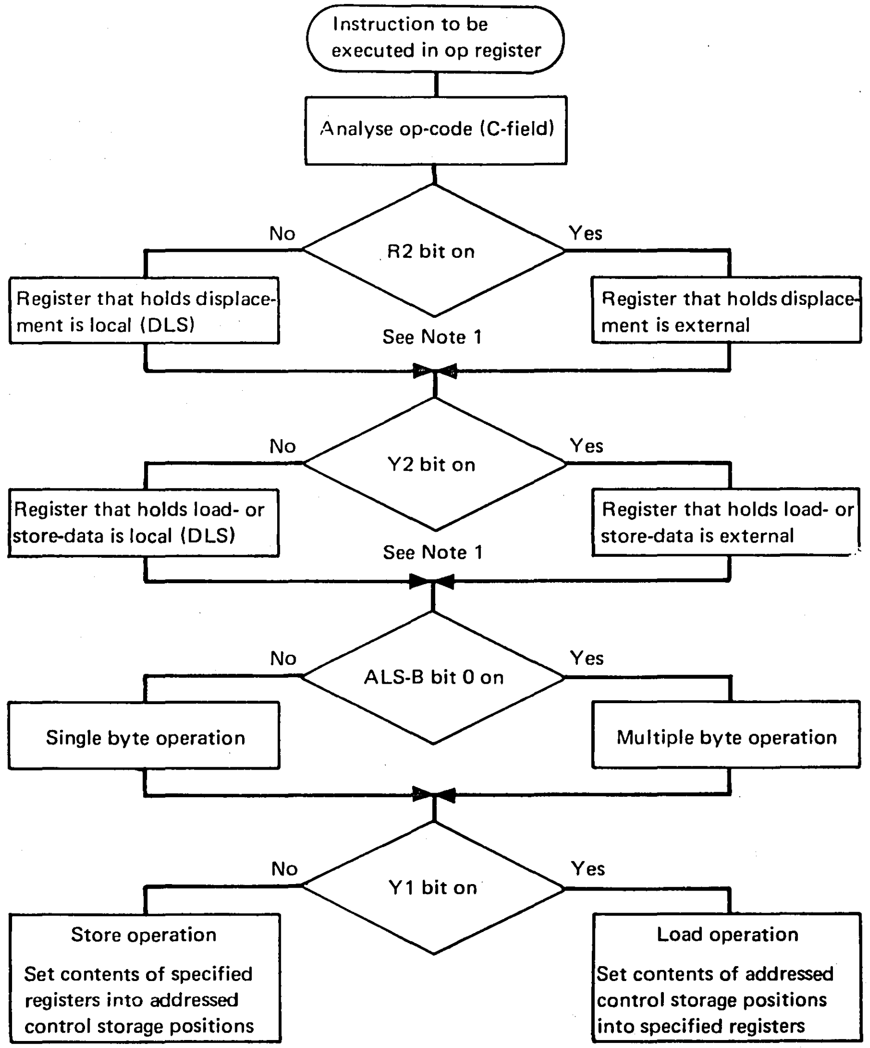
Microinstructions (continued)

Store, Load (Multiple Byte) Operations

Multiple byte operations in principle are executed in the same way as single byte operations with the exception that the same number of DS cycles take place, as bytes handled.

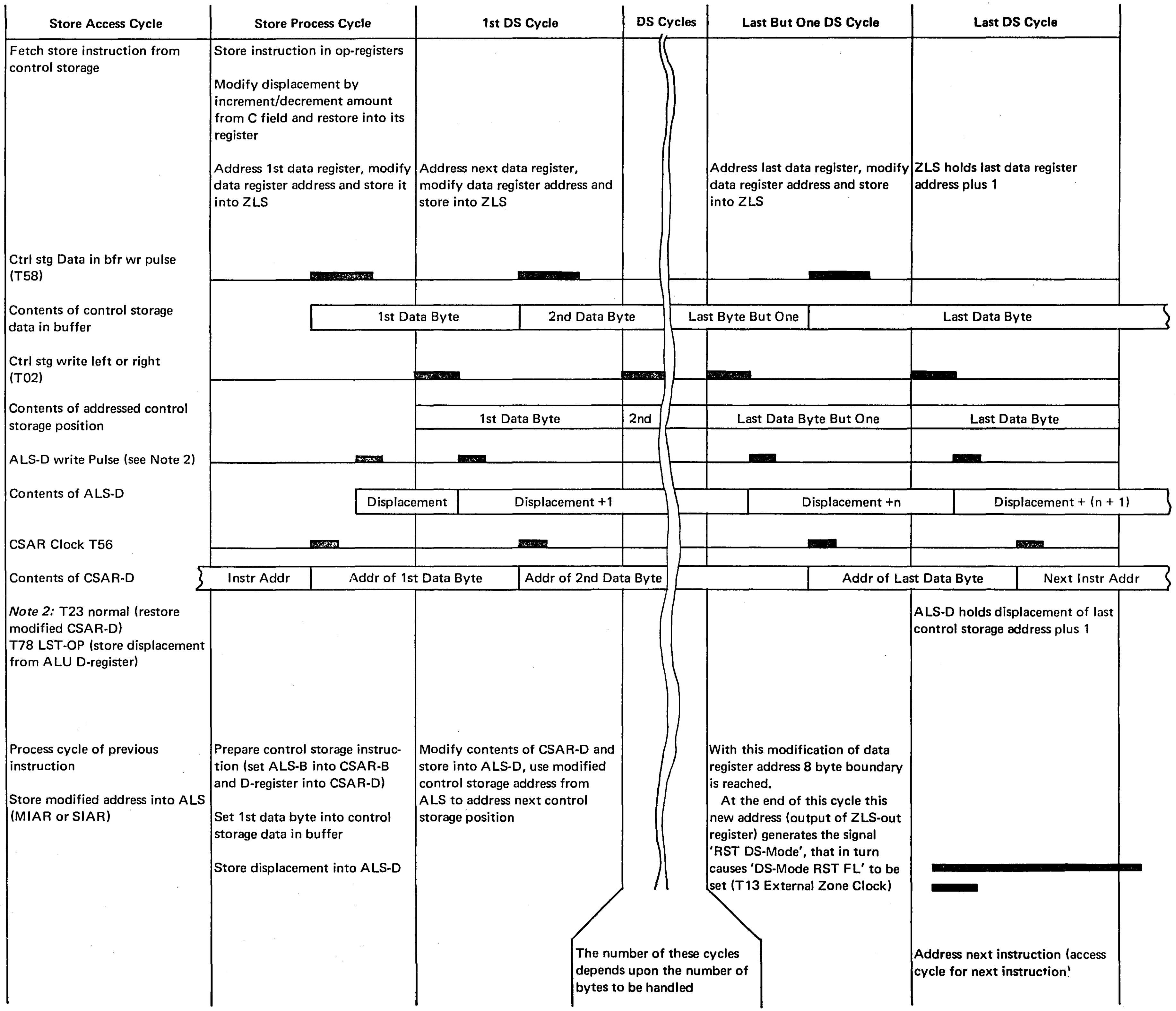
A maximum of eight bytes may be handled with one instruction. A multiple byte operation is terminated as soon as a data register address with all the three low order bits on or off is reached (8 byte boundary). This means the number of bytes that is handled is defined by the start address, defined by the Y-field in the instruction.

Multiple byte operations are executed according to the schemes shown here for a store-op, and on Page 3-255 for a load-op.



Note 1: Registers displacement and data must not both be external.

STORE (Multiple byte) Operation



Store Access Cycle: Fetch store instruction from control storage

Store Process Cycle: Store instruction in op-registers

Modify displacement by increment/decrement amount from C field and restore into its register

Address 1st data register, modify data register address and store it into ZLS

1st DS Cycle: Address next data register, modify data register address and store into ZLS

Last But One DS Cycle: Address last data register, modify data register address and store into ZLS

Last DS Cycle: ZLS holds last data register address plus 1

Note 2: T23 normal (restore modified CSAR-D)
T78 LST-OP (store displacement from ALU D-register)

ALS-D holds displacement of last control storage address plus 1

Process cycle of previous instruction

Store modified address into ALS (MIAR or SIAR)

Prepare control storage instruction (set ALS-B into CSAR-B and D-register into CSAR-D)

Set 1st data byte into control storage data in buffer

Store displacement into ALS-D

Modify contents of CSAR-D and store into ALS-D, use modified control storage address from ALS to address next control storage position

With this modification of data register address 8 byte boundary is reached.

At the end of this cycle this new address (output of ZLS-out register) generates the signal 'RST DS-Mode', that in turn causes 'DS-Mode RST FL' to be set (T13 External Zone Clock)

The number of these cycles depends upon the number of bytes to be handled

Address next instruction (access cycle for next instruction'

The way in which a load multiple byte operation is executed is shown on this page.

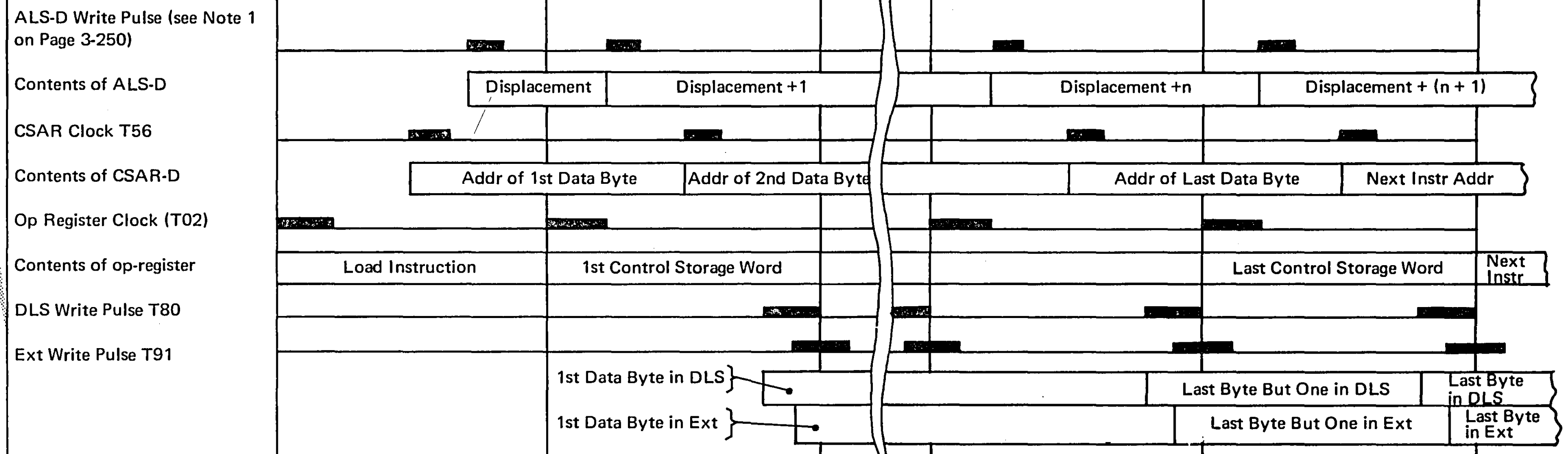
With either DLS or external write pulse, data from addressed control storage position is set into addressed data register. These write pulses are active exclusively with the two signals:

- Left byte select (CSAR-D bit 0 = 0)
- Right byte select (CSAR-D bit 0 = 1)

R-field or Y-field from op-register are set into CRY-switch. From here, selected data byte is gated via ALU circuitry into addressed register.

LOAD (Multiple byte) Operation

(Store) Access Cycle	(Store) Process Cycle	1st DS Cycle	DS Cycles	Last But One DS Cycle	Last DS Cycle
Fetch load instruction from control storage	Load instruction in op-register Modify displacement by increment/decrement amount from C-field and restore into its register Store data register address into ZLS with *no* modification	Address 1st data register, modify data register address and store into ZLS		Address last data register, modify data register address and store into ZLS	ZLS holds last data register address plus 1

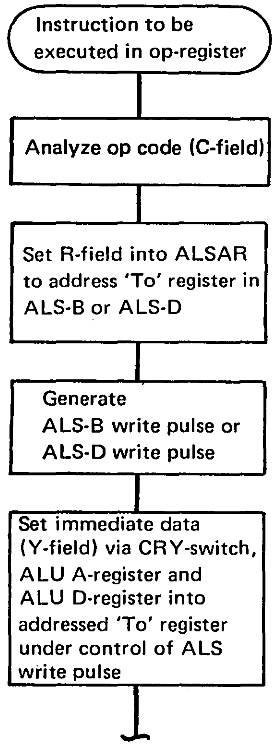


(Store) Access Cycle	(Store) Process Cycle	1st DS Cycle	DS Cycles	Last But One DS Cycle	Last DS Cycle
					ALS-D holds displacement of last control storage address plus 1
Process cycle of previous instruction Store modified address into ALS (MIAR or SIAR)	Prepare control storage address (set ALS-B into CSAR-B and D-register into CSAR-D) Store displacement into ALS-D	Set 1st control storage word into op-register Modify contents of CSAR-D and store into ALS-D; use modified control storage address from ALS to address next control storage position With respective write pulses, data byte is set into addressed register, where it is available for further processing in the next cycle	The number of these cycles depends upon the number of bytes to be handled	With this modification of data register address on 8 byte boundary is reached From output of ZLS-out register 'RST DS-Mode' is generated, that in turn causes 'DS-Mode RST FL' to be set in the next cycle. See also Page 3-250	Address next instruction (access cycle for next instruction)

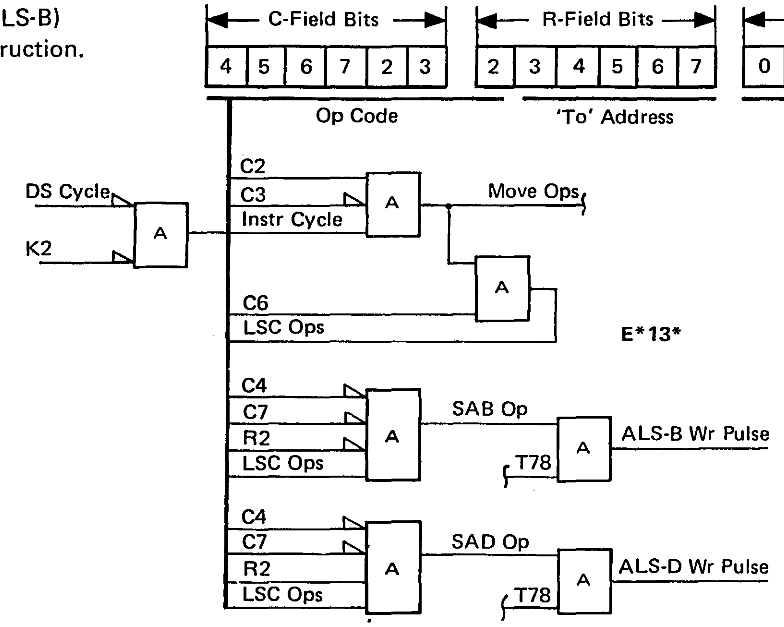
Microinstructions (continued)

SABI and SADI Process Cycle

- Group 3: Process cycle of SABI (Store into ALS-B) instruction and SADI (Store into ALS-D) instruction.



- Op Register Layout and ALS Write Pulse Generation



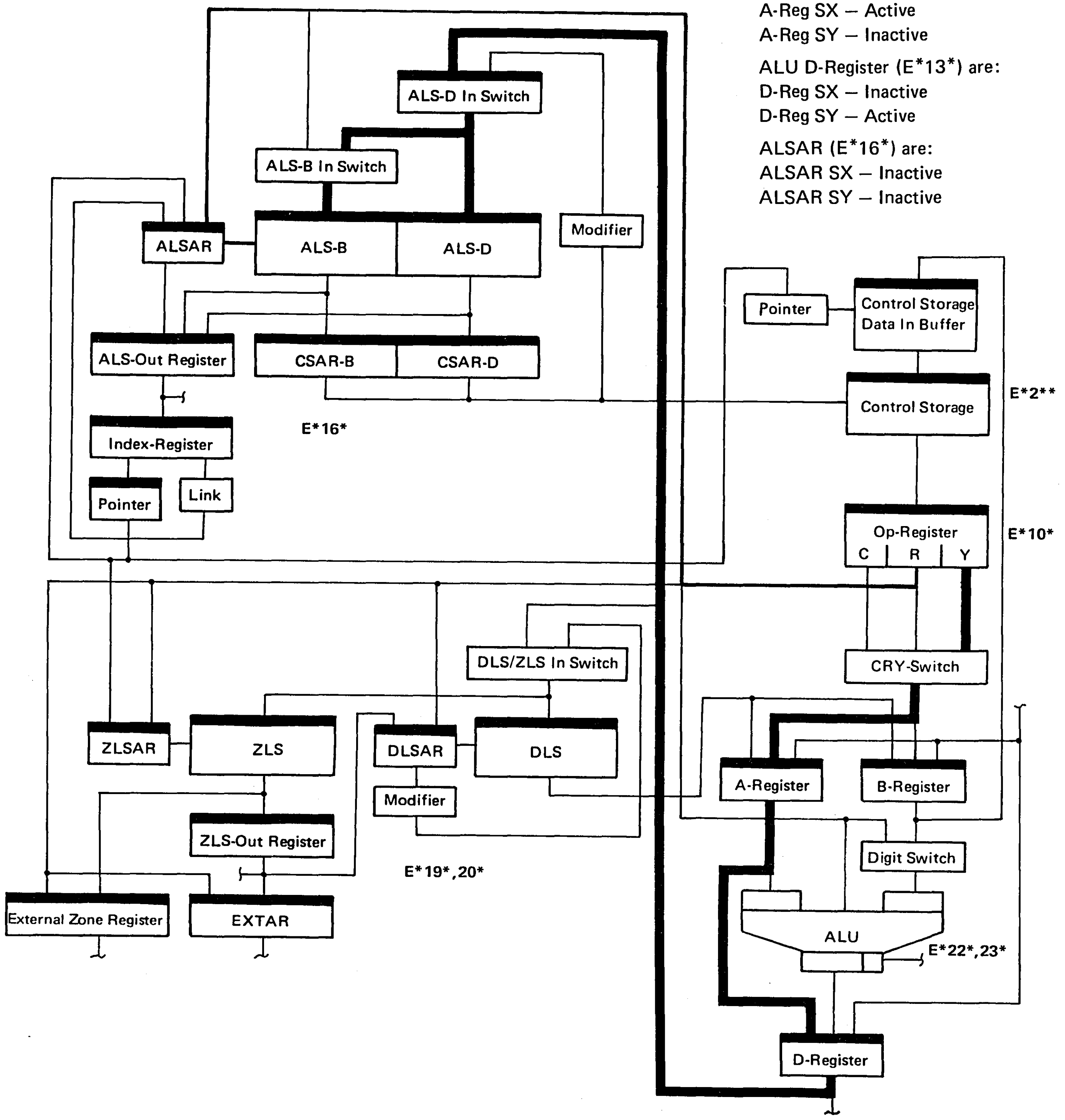
- Control Signals for:

CRY Switch (E*10*) are:
CRY Switch SX 0 to 5 – Inactive
CRY Switch SX 6, 7 – Inactive
CRY Switch SY – Active

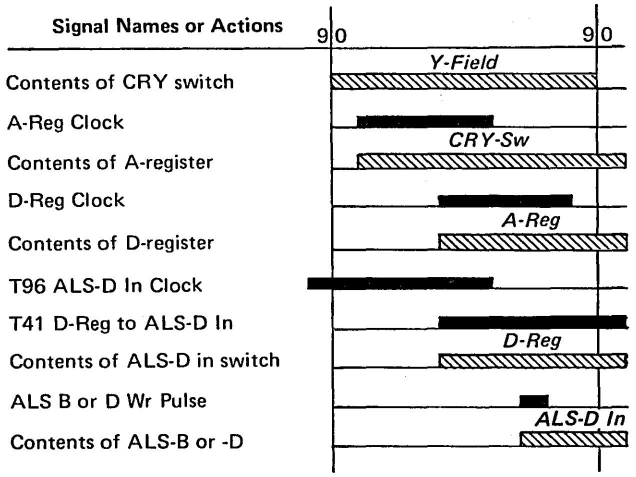
ALU A-Register (E*13*) are:
A-Reg SX – Active
A-Reg SY – Inactive

ALU D-Register (E*13*) are:
D-Reg SX – Inactive
D-Reg SY – Active

ALSAR (E*16*) are:
ALSAR SX – Inactive
ALSAR SY – Inactive

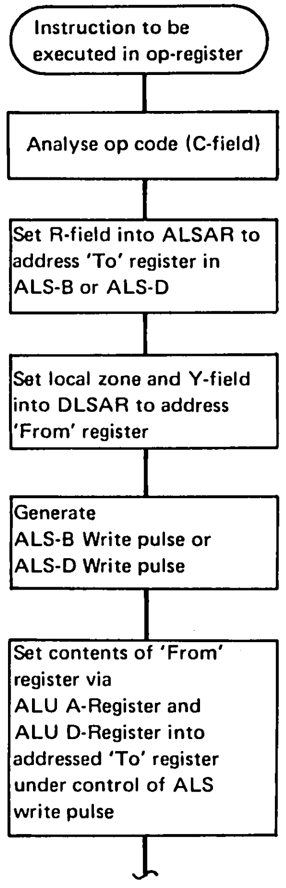


- Timing

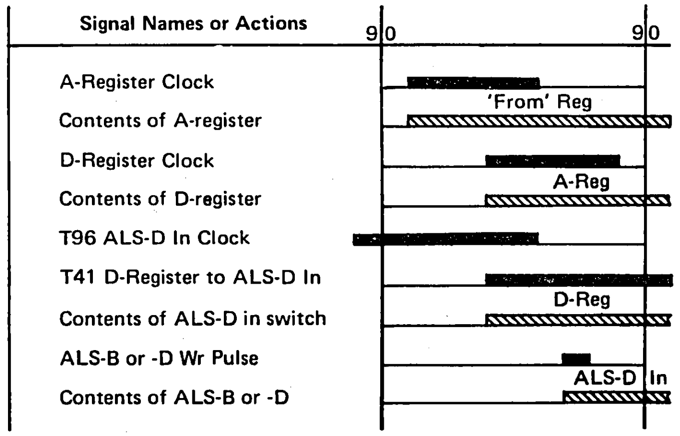


SABR and SADR Process Cycle (DLS register to ALS)

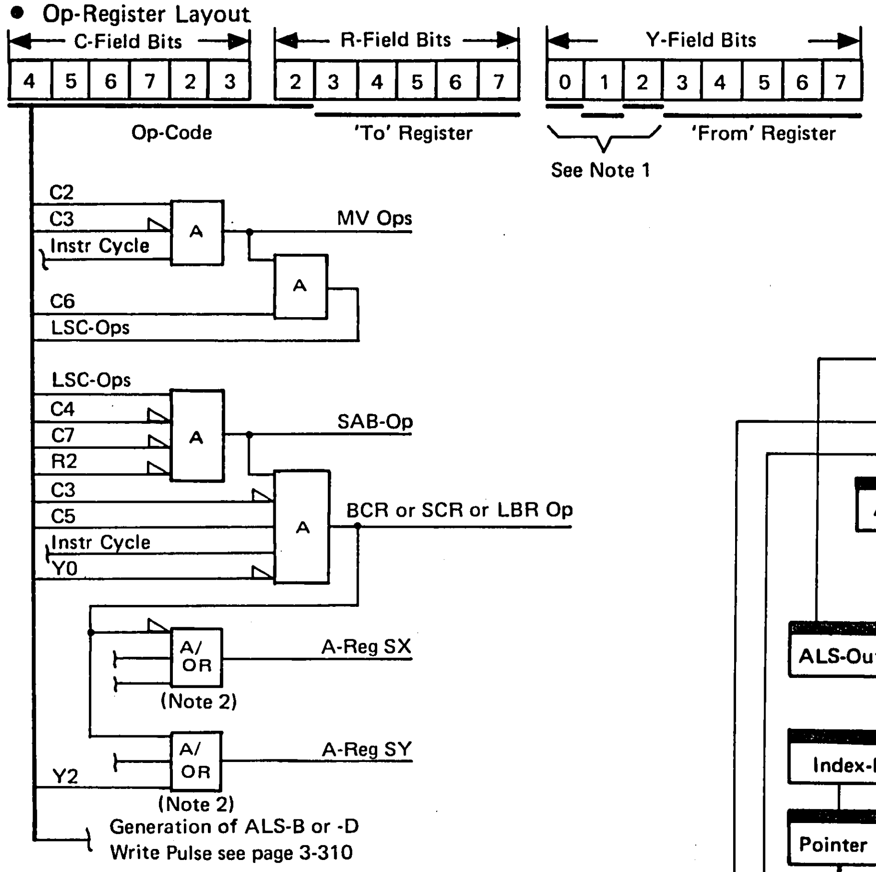
- Group 3: Process cycle of SABR (Store into ALS-B) instruction and SADR (Store into ALS-D) instruction.



- Timing



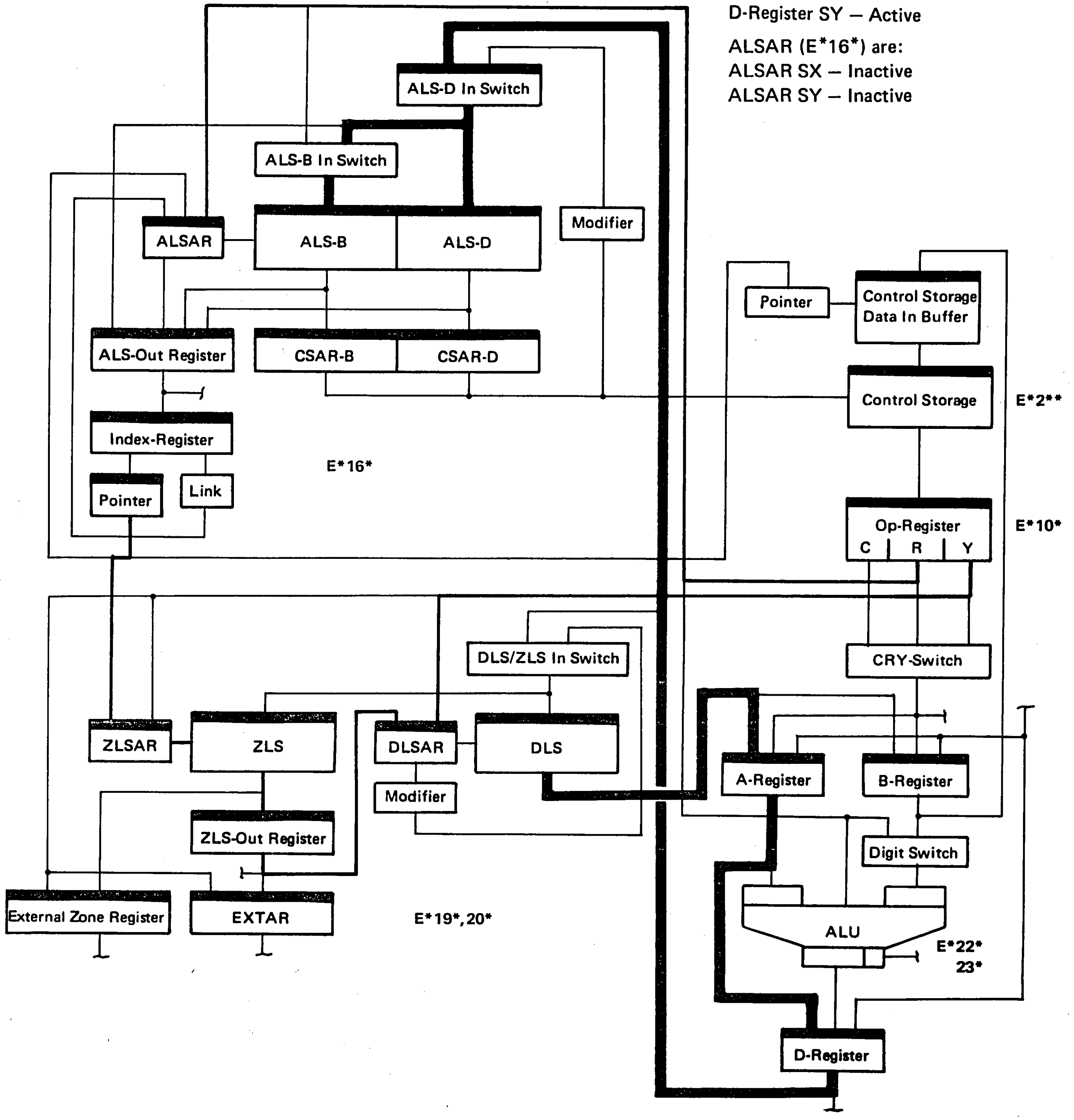
- Op-Register Layout



Note 1: Y0 – Has to be zero
Y1 – Has to be zero
Y2 if 'On' = external register
if 'Off' = DLS register

Note 2: This box represents a number of AND/OR blocks
See also notes on page 3-030

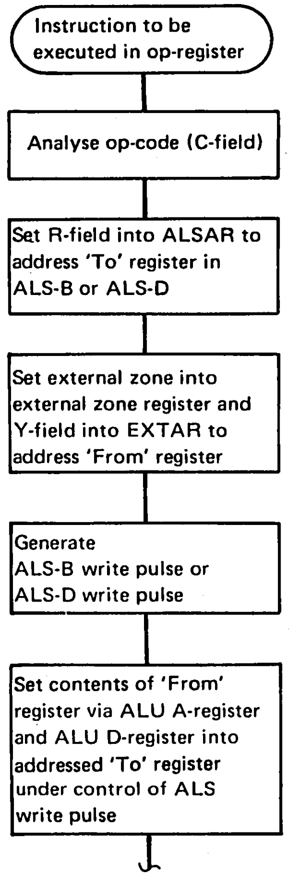
- Control Signals for:
  ALU A-register (E*13*) are:
  A-Register SX – Inactive
  A-Register SY – Inactive
  ALU D-register (E*13*) are:
  D-Register SX – Inactive
  D-Register SY – Active
  ALSAR (E*16*) are:
  ALSAR SX – Inactive
  ALSAR SY – Inactive



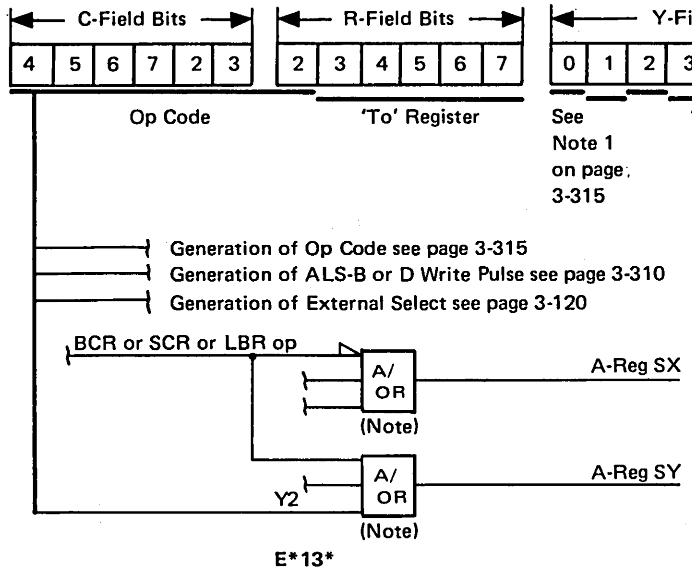
Microinstructions (continued)

SABR and SADR Process Cycle (External register to ALS)

- Group 3: Process cycle of SABR (Store into ALS-B) instruction and SADR (Store into ALS-D) instruction



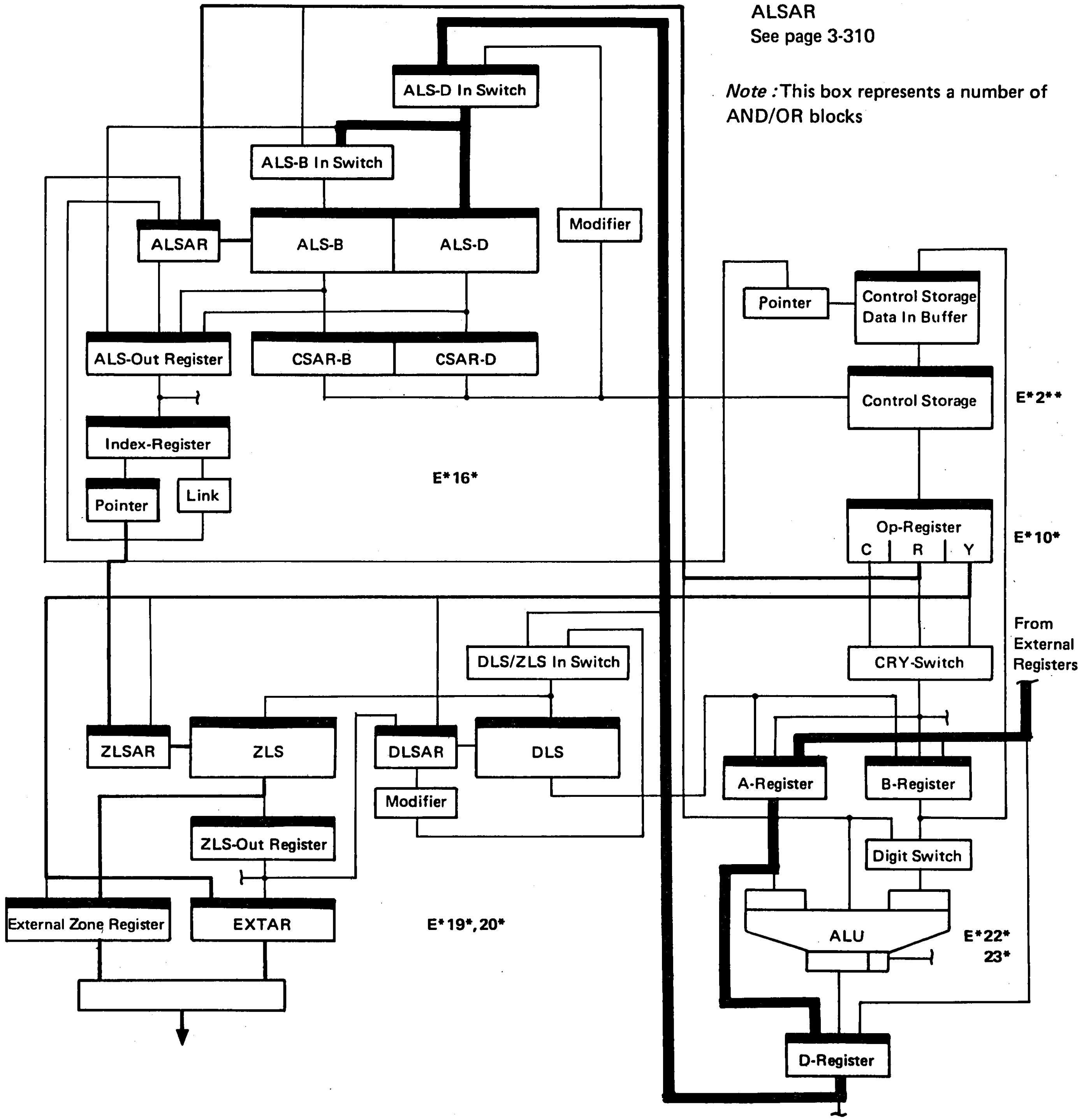
- OP-Register Layout



- Timing (see page 3-315)
  Same as for DLS to ALS except that external register is loaded into ALU A-register.

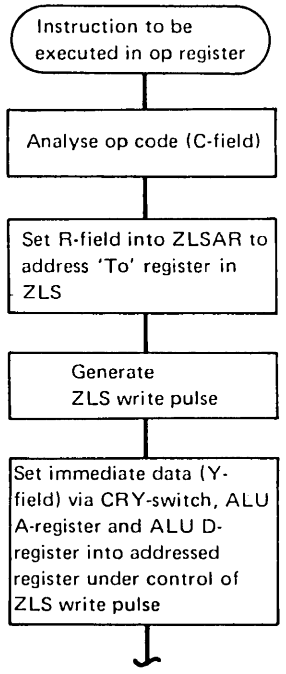
- Control Signals for:
  ALU A-register (E*13*) are:
  A-Register SX – Inactive
  A-Register SY – Active
  ALU D-Register
  ALSAR
  See page 3-310

Note : This box represents a number of AND/OR blocks

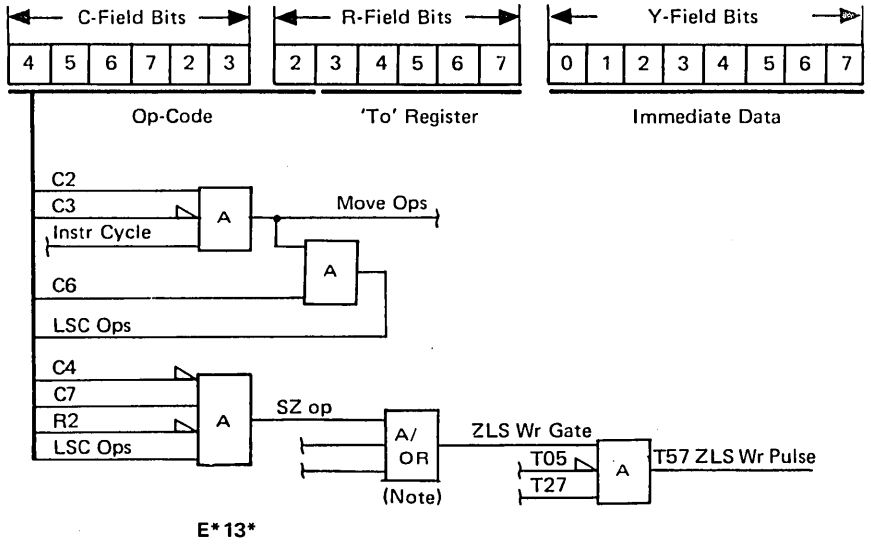


SZI Process Cycle

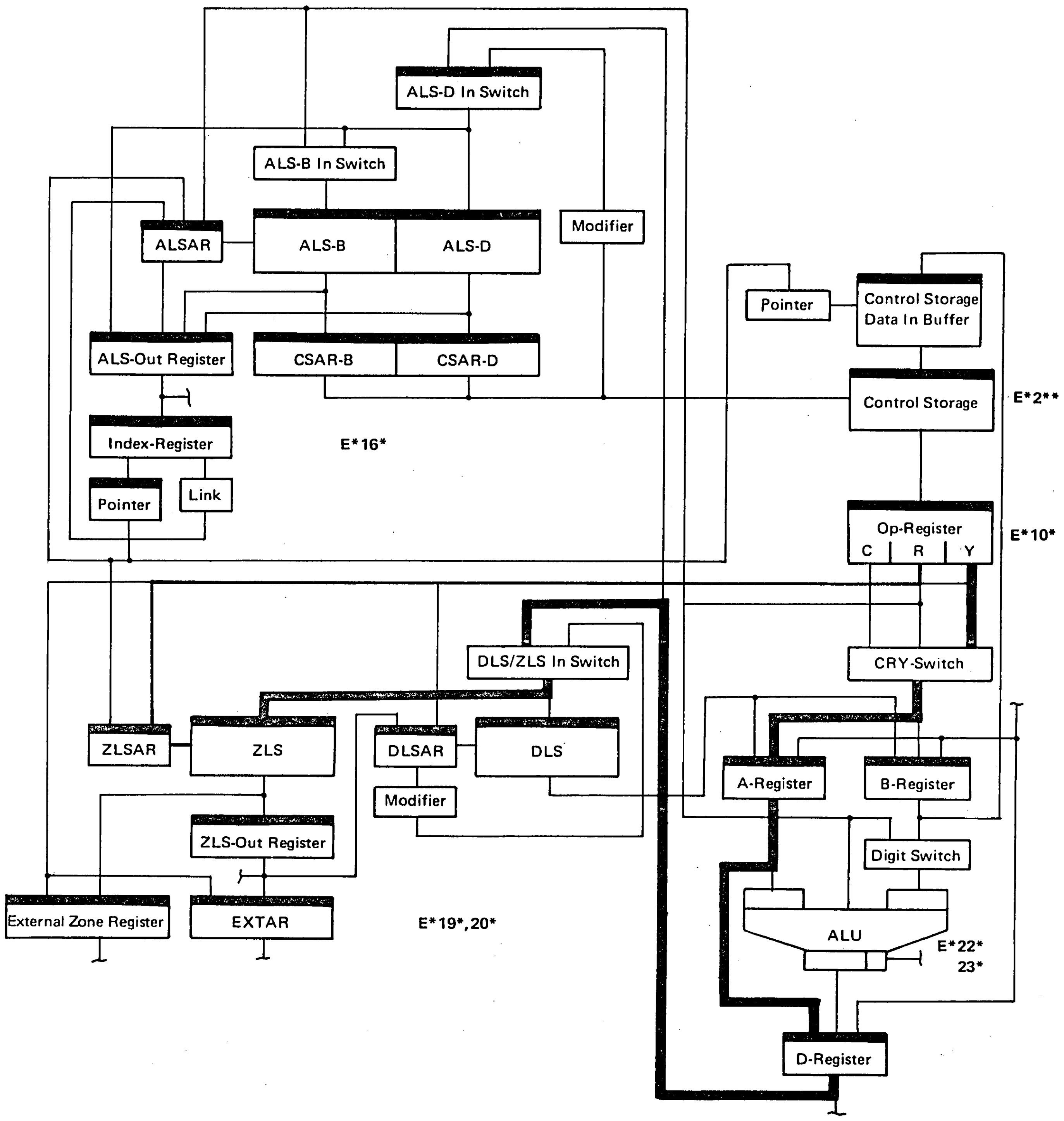
- Group 3: Process cycle of SZI (Store into ZLS) instruction.



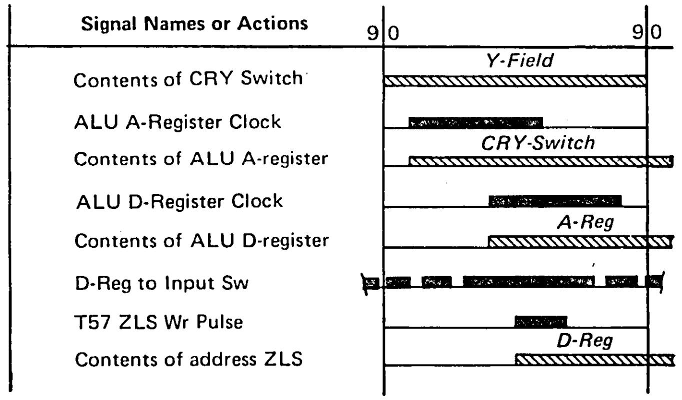
- Op-Register Layout and ZLS Write Pulse Generation



- Control Signals for:
CRY-Switch
ALU A-Register
ALU D-Register
See page 3-310
Note: This box represents a number of AND/OR blocks



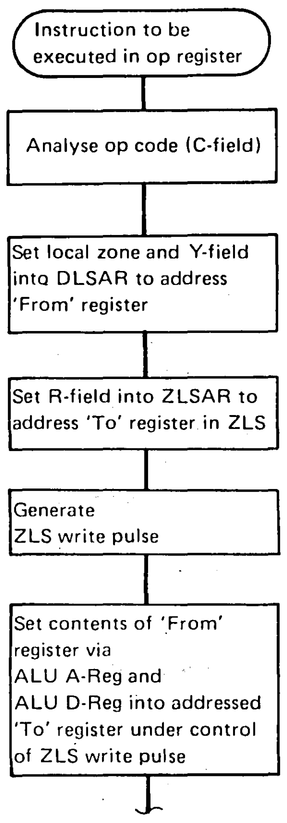
- Timing



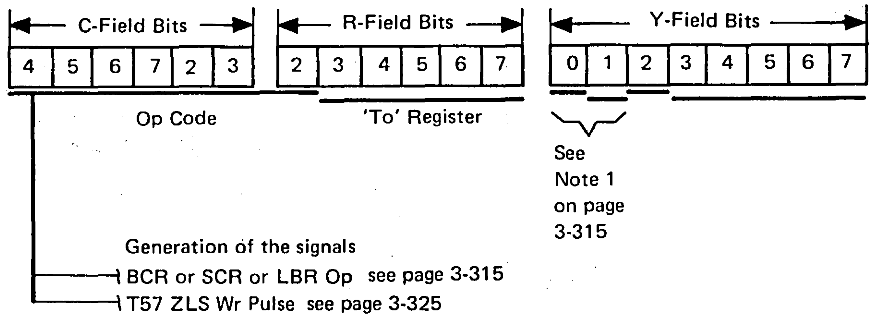
Microinstructions (continued)

SZR Process Cycle (DLS Register to ZLS)

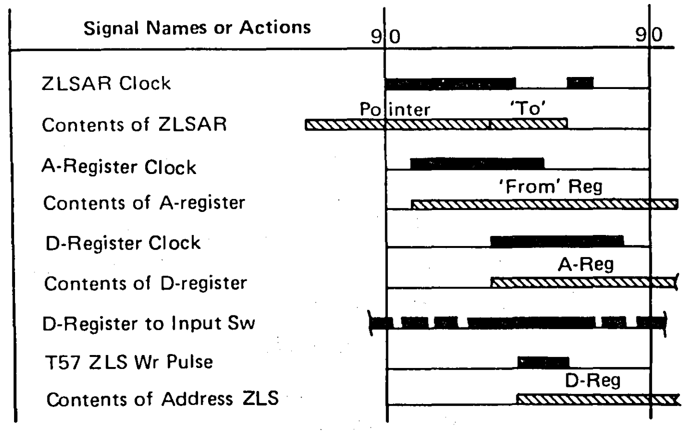
- Group 3: Process cycle of SZR (Store into ZLS) instruction.



- Op-Register Layout



- Timing



- Control Signals for:

  ALU A-Register (E*13*) are:

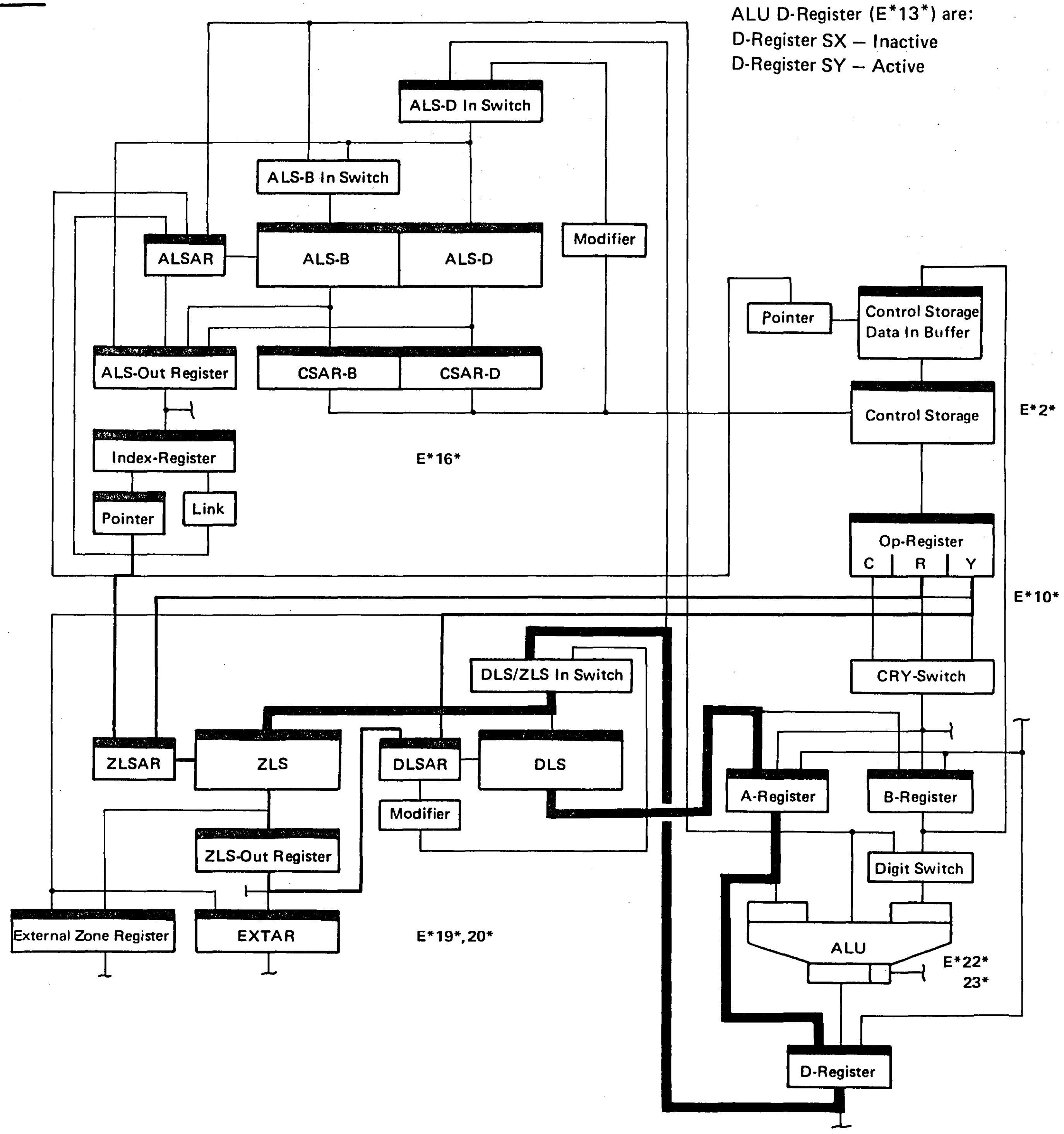
  A-Register SX – Inactive

  A-Register SY – Inactive

  ALU D-Register (E*13*) are:

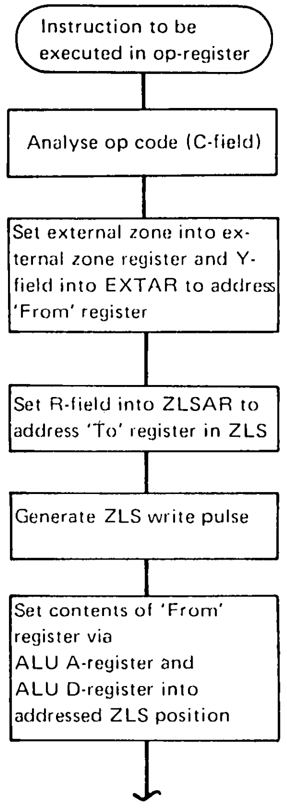
  D-Register SX – Inactive

  D-Register SY – Active

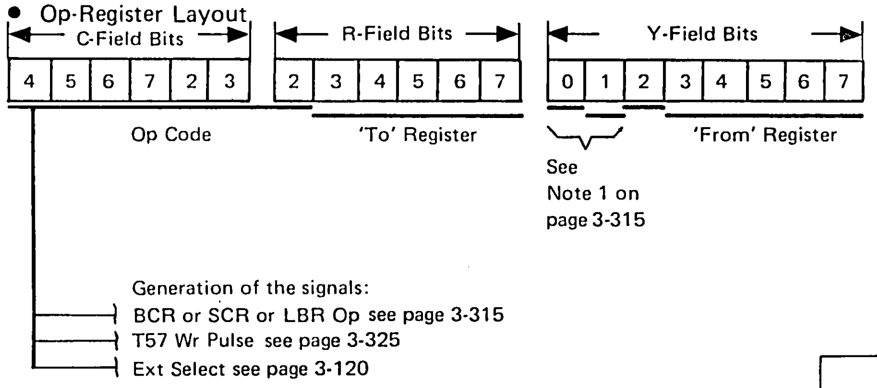


SZR Process Cycle (External Register to ZLS)

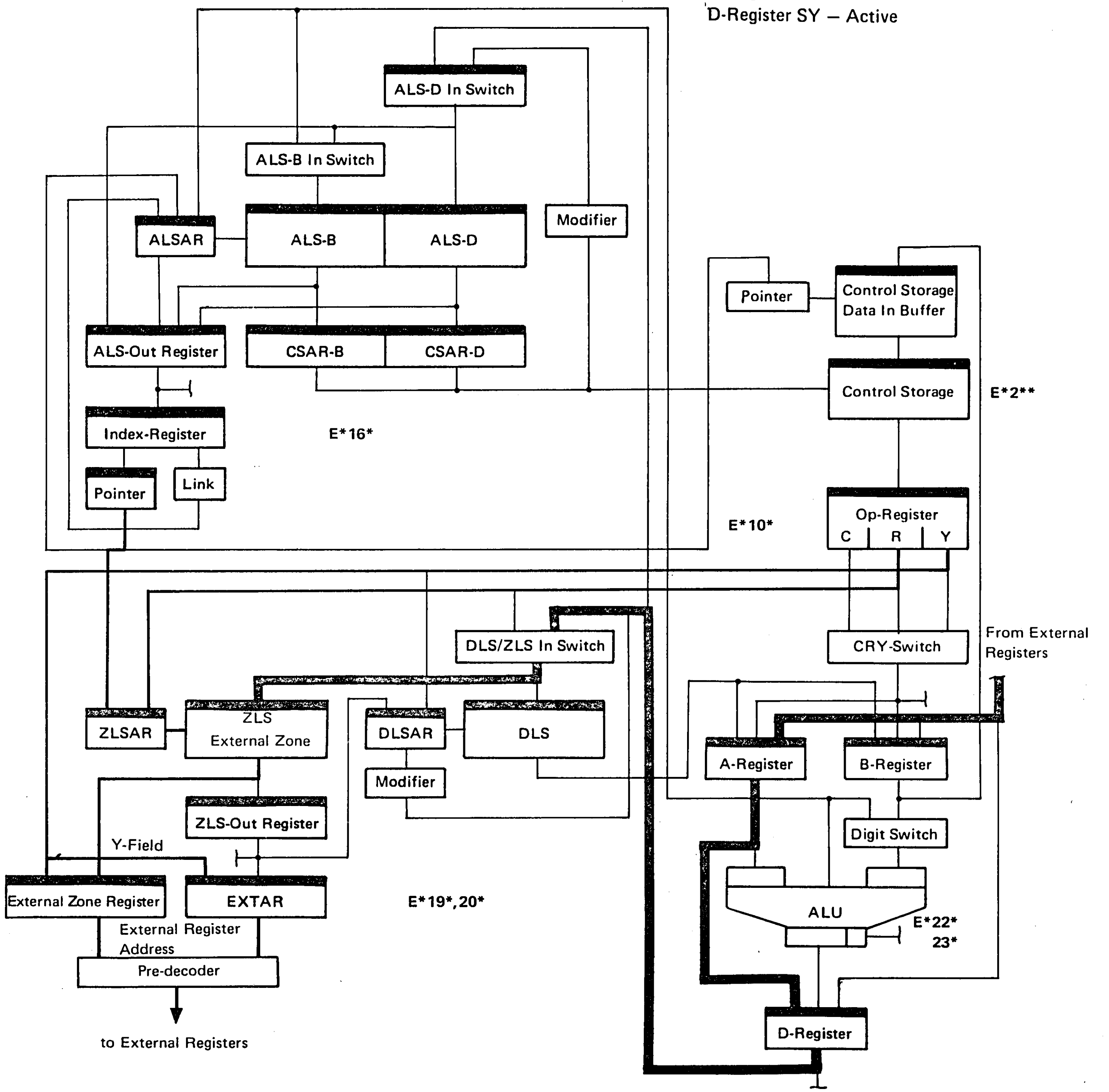
- Group 3: Process cycle of SZR (Store into ZLS) instruction.



- Op-Register Layout



- Control Signals for:
  ALU A-Register (E*13*) are:
  A-Register SX – Inactive
  A-Register SY – Active
  ALU D-Register (E*13*) are:
  D-Register SX – Inactive
  D-Register SY – Active

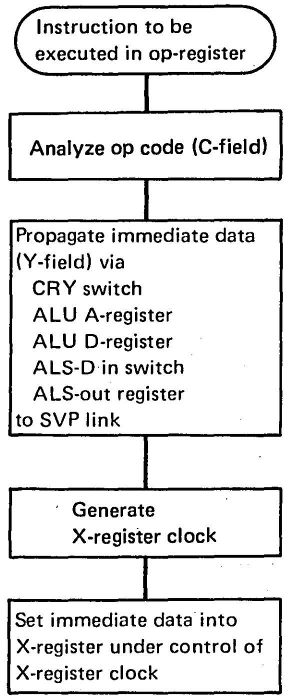


- Timing (see page 3-330)
  Same as for DLS to ZLS except that external register is loaded into ALU A-register.

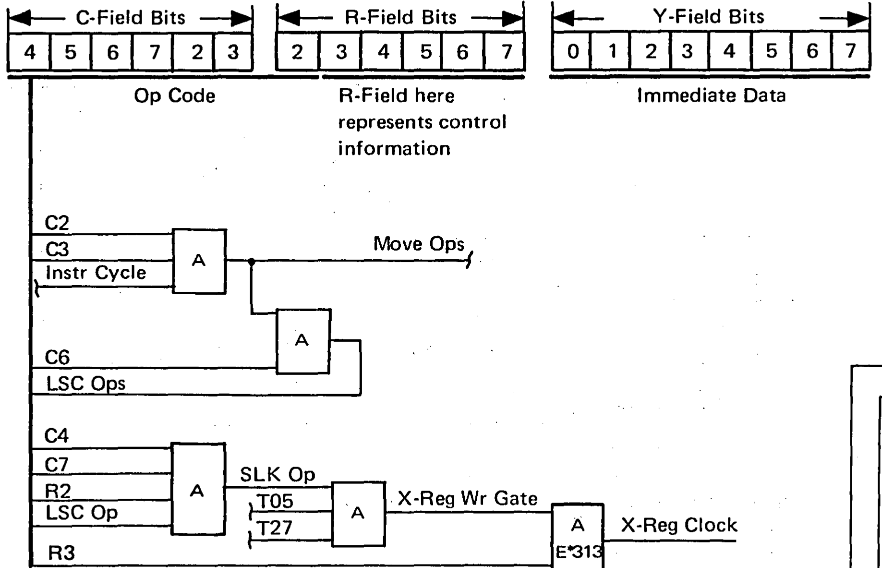
Microinstructions (continued)

SLKI Process Cycle

- Group 3: Process cycle of SLKI (Store into X-Register of SVP link card) instruction.



- Op-Register Layout and X-Register Clock Generation



R3 'on' Specified data is transferred to X-register (immediate data or contents of register specified by Y-Field)

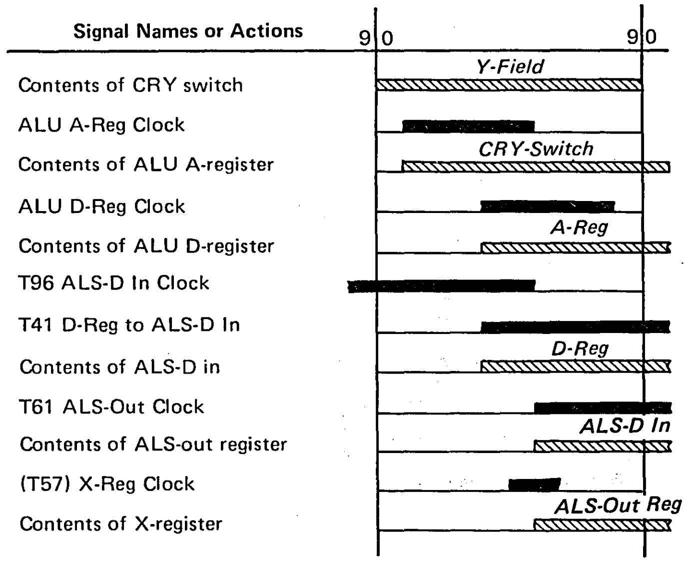
R4 'on' Set 'PCR' FL

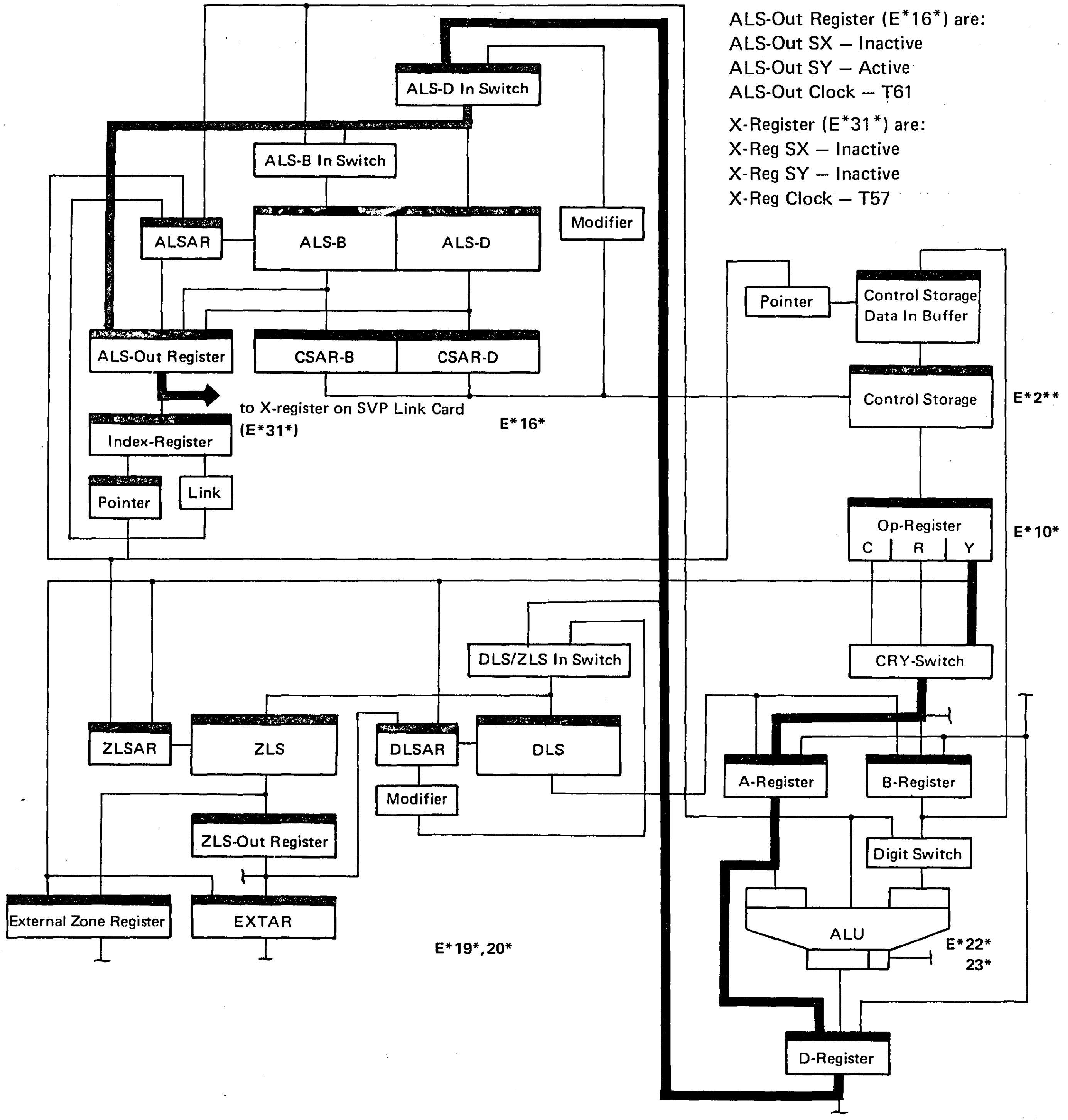
R5 'on' Reset 'SVP Requ' FL

R6 Unused

R7 'on' Reset 'prevent I/O' FL. See also page 4-055

- Timing





- Control Signals for:
  CRY Switch
  ALU A-Register
  ALU D-Register

See page 3-325

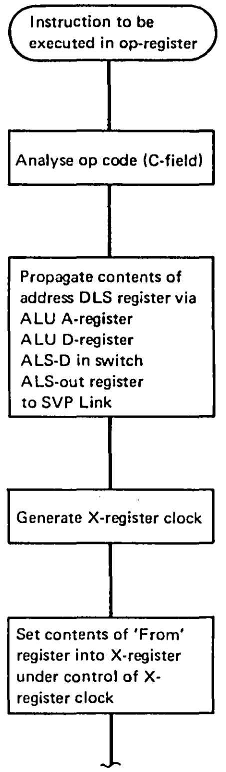
ALS-Out Register (E*16*) are:
ALS-Out SX – Inactive
ALS-Out SY – Active
ALS-Out Clock – T61

X-Register (E*31*) are:
X-Reg SX – Inactive
X-Reg SY – Inactive
X-Reg Clock – T57

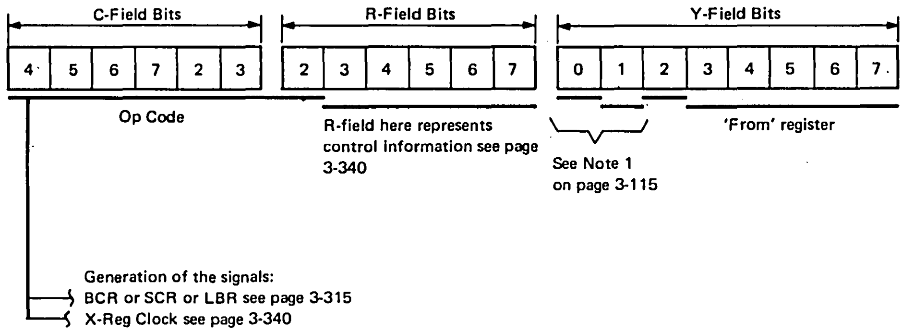
SLKR Process Cycle

(DLS register to X-register)

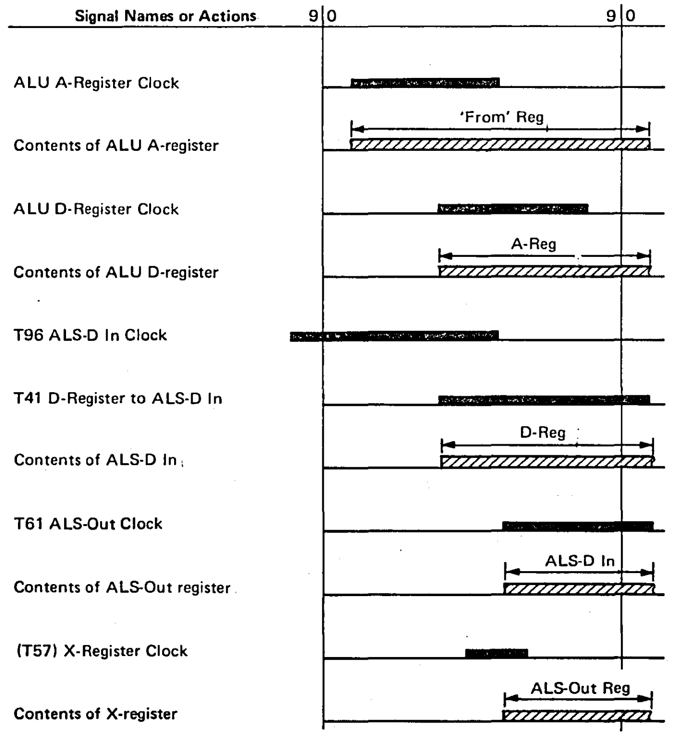
- Group 3: Process cycle of SLKR (Store into X-Register of SVP Link) instruction



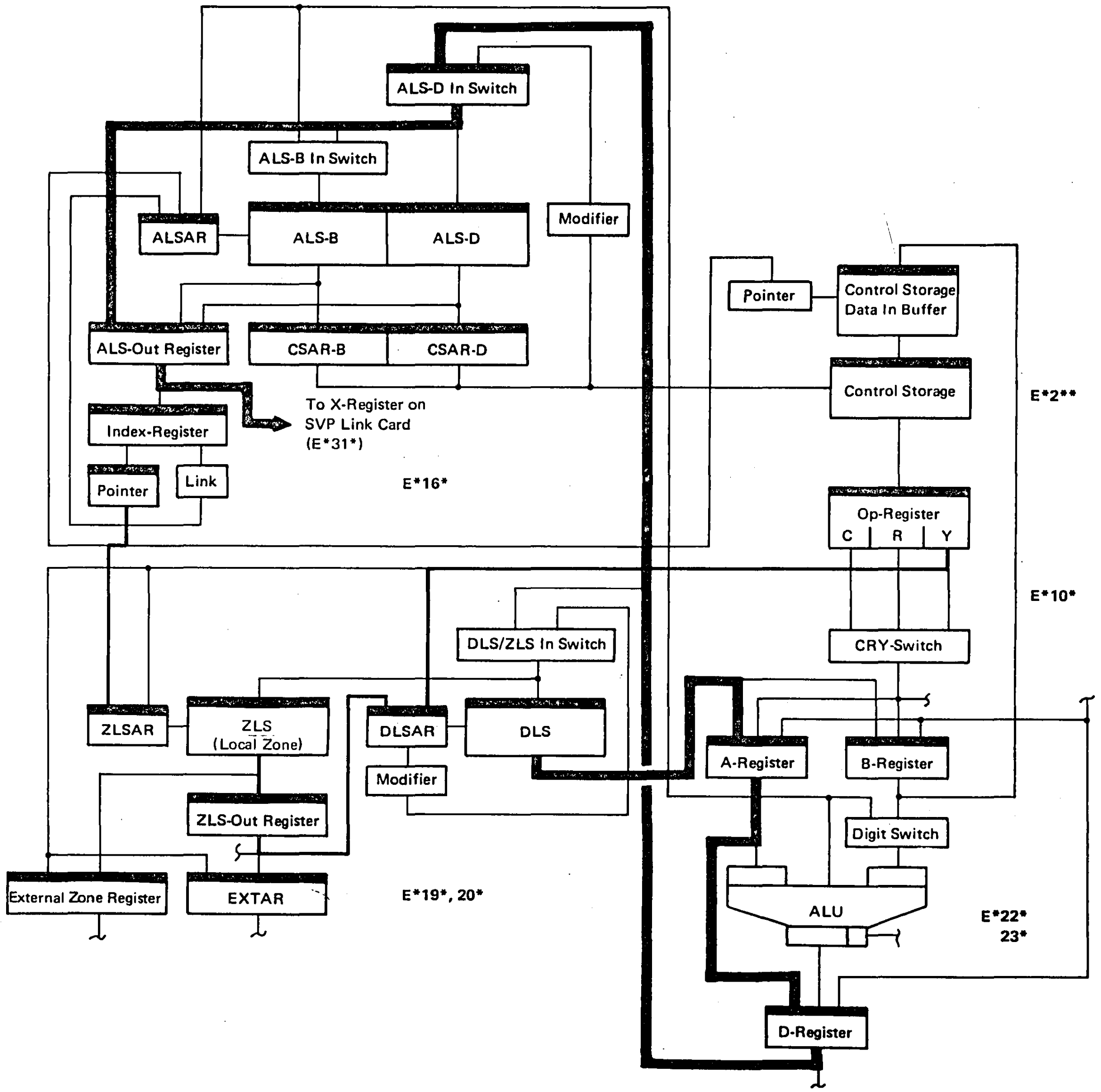
- Op-register Layout



- Timing



- Control signals for:
  ALU A-Register (E*13*) are:
  A-Reg SX – Inactive
  A-Reg SY – Inactive
  ALU D-Register
  ALS-Out Register
  X-Register
  See page 3-340

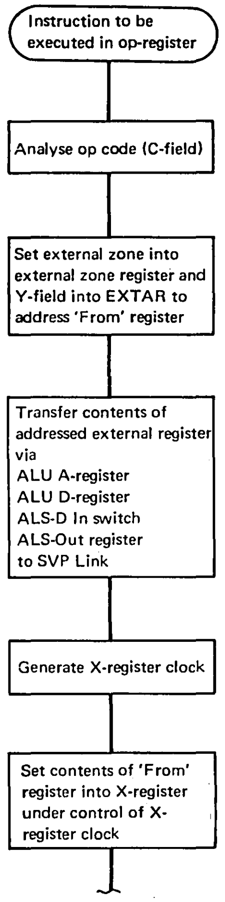


Microinstructions (continued)

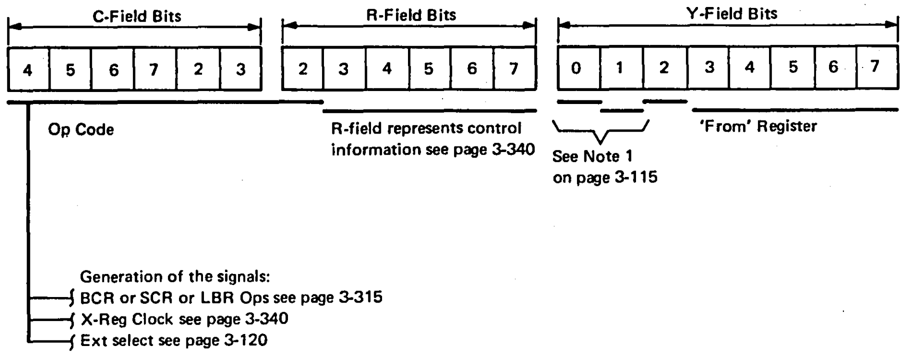
SLKR Process Cycle

(External register to X-register)

- Group 3: Process cycle of SLKR (Store into X-Register of SVP Link) instruction

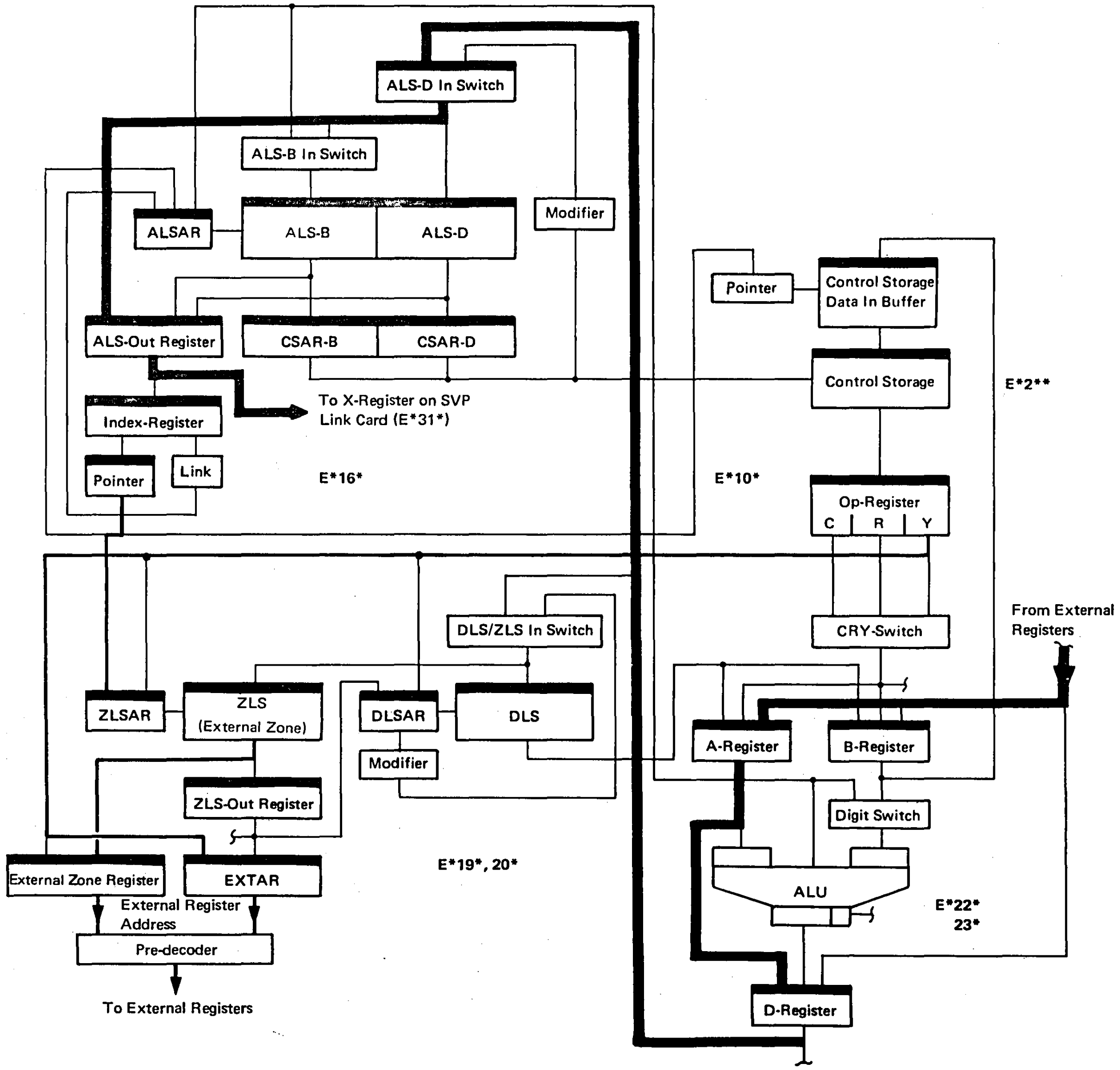


- Op-Register Layout



- Timing (see page 3-345)
  Same as for DLS to X-register except that external register is loaded into ALU A-register

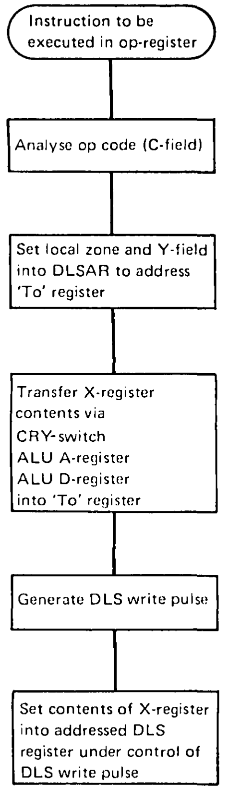
- Control Signals for:
  ALU A-Reg (E*13*) are:
  A-Reg SX – Inactive
  A-Reg SY – Active
  ALU D-Register see page 3-335
  ALS-Out Register see page 3-340
  X-Register see page 3-340



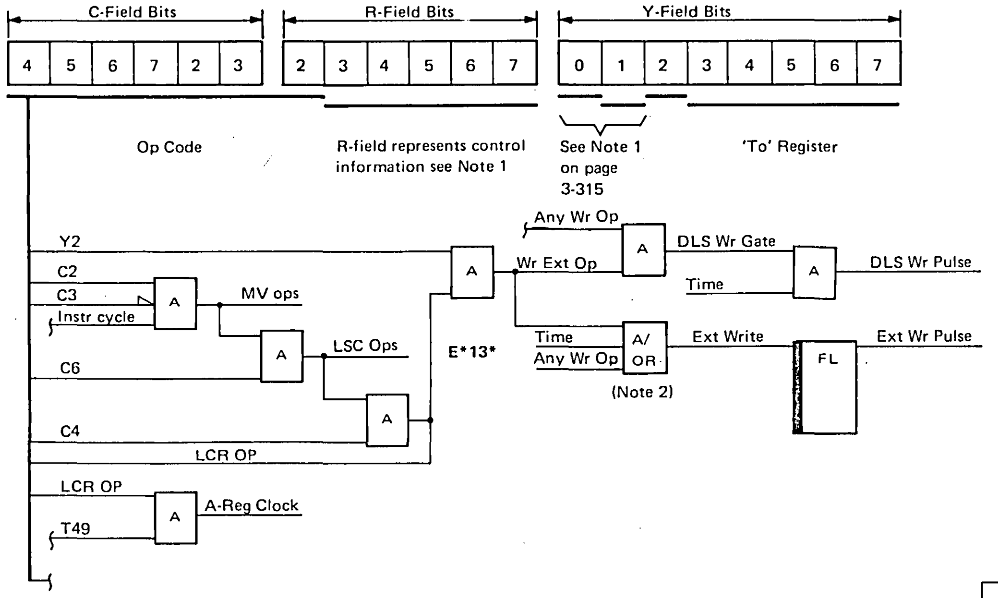
LLKR Process Cycle

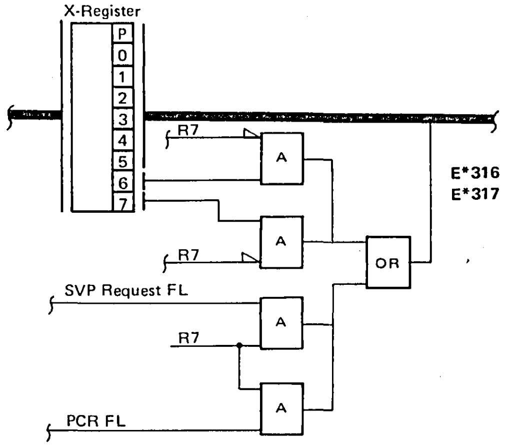
(X-Register to DLS)

- Group 3: Process cycle of LLKR (load X-Register contents into 'To' register) instruction.

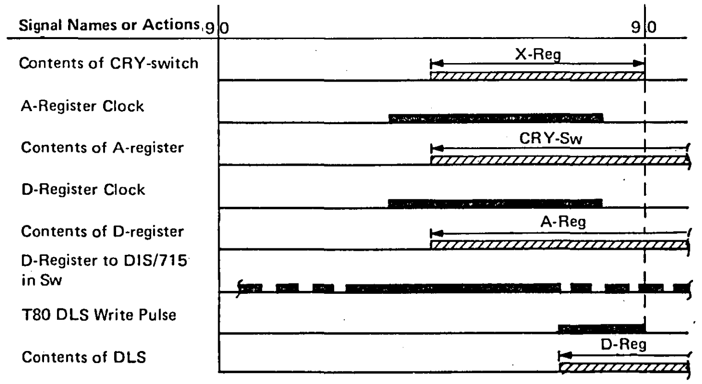


- Op-Register Layout





- Timing



- Control Signals for:

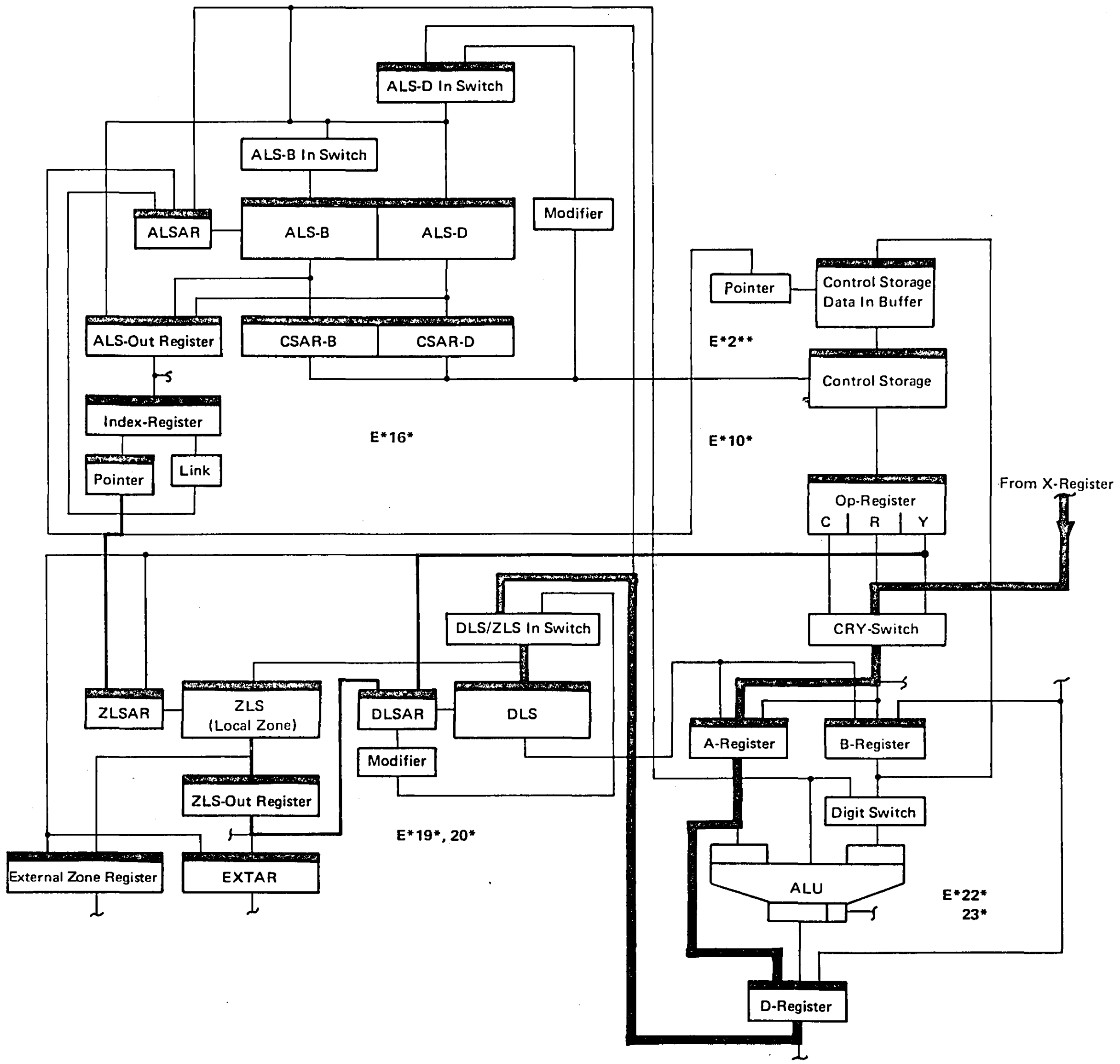
CRY -Switch (E*10*) are:
CRY-Switch SX 0 to 5 – Active
CRY-Switch SX 6,7 – Active
CRY-Switch SY – Active

ALU A-Register (E*13*) are:
A-Register SX – Active
A-Register SY – Inactive

ALU D-Register (E*13*) are:
D-Register SX – Inactive
D-Register SY – Active

Notes:

1. R2 – unused
   R3 to R6 – have to be zero
   R7 off – X-register is set into register specified by Y3 to Y7
   R7 on – X-register bits 0 to 5 are set into bit positions 0 to 5 of register specified by Y3 to Y7.
   Bit position 6 = SVP Requ FL
   Bit position 7 = PCR FL
2. This box represents a number of AND/OR blocks

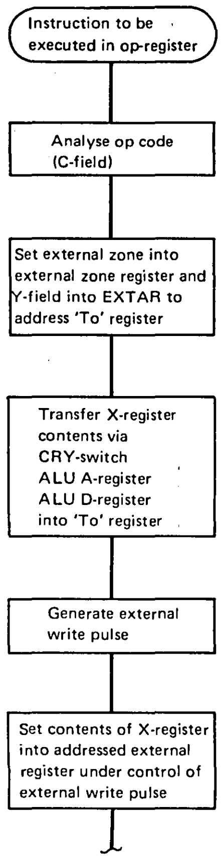


Microinstructions (continued)

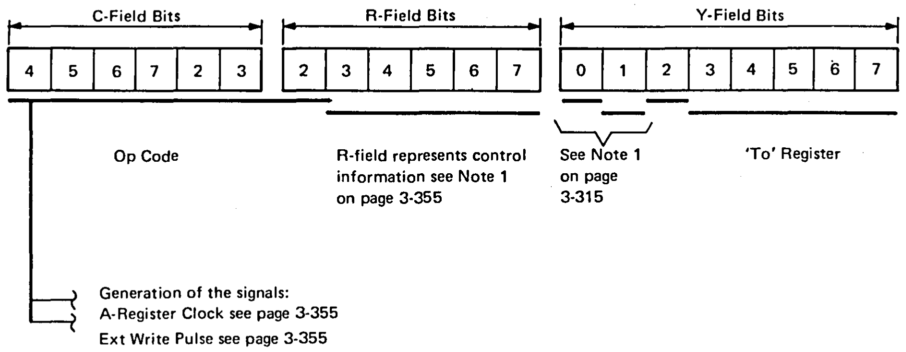
LLKR Process Cycle

(X-Register to External Register)

- Group 3: Process Cycle of LLKR (load X-Register contents into 'To' Register) instruction.

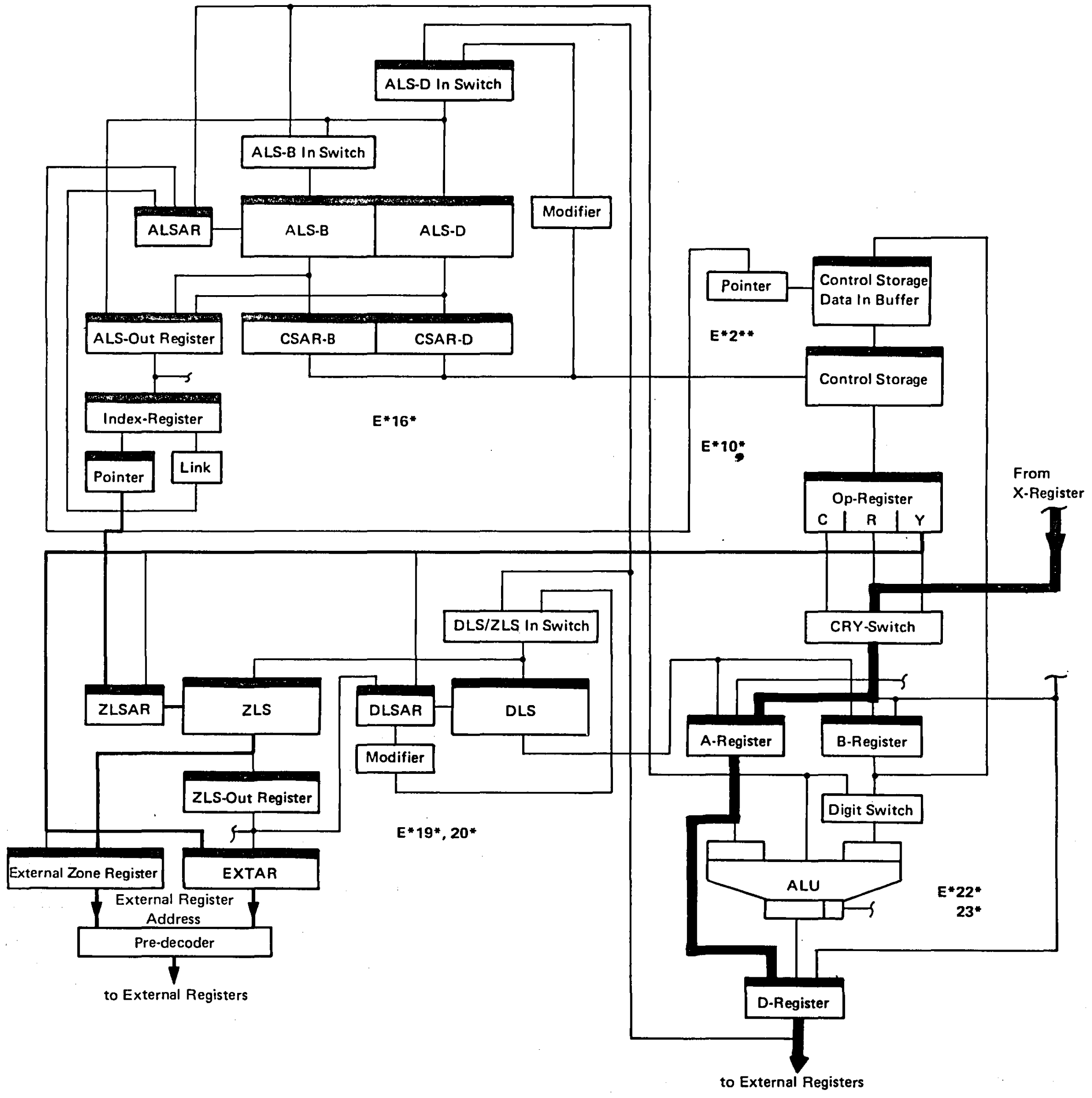


- Op-Register Layout



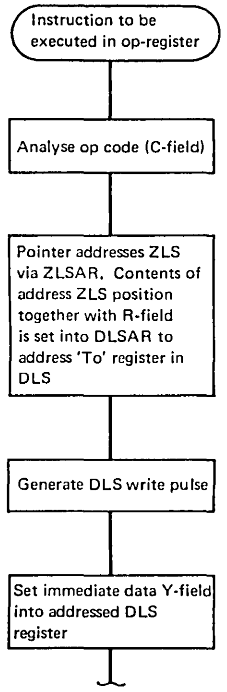
Control Signals for:
CRY-Switch
ALU A-Register
ALU D-Register
see page 3-355

- Timing (see page 3-355)
  Same as for X-register to DLS, except that D-register contents is loaded into external register

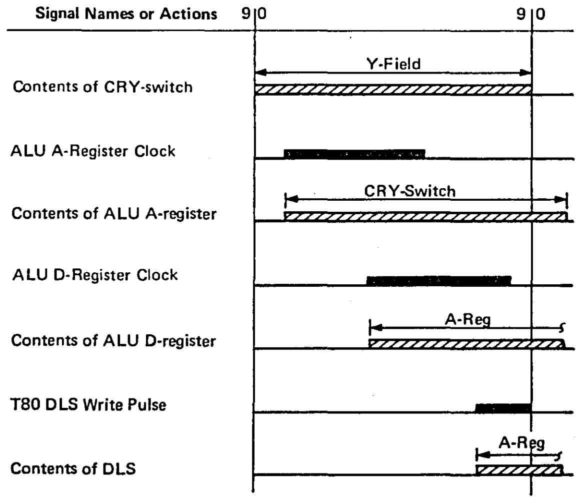


LBI to DLS Register Process Cycle

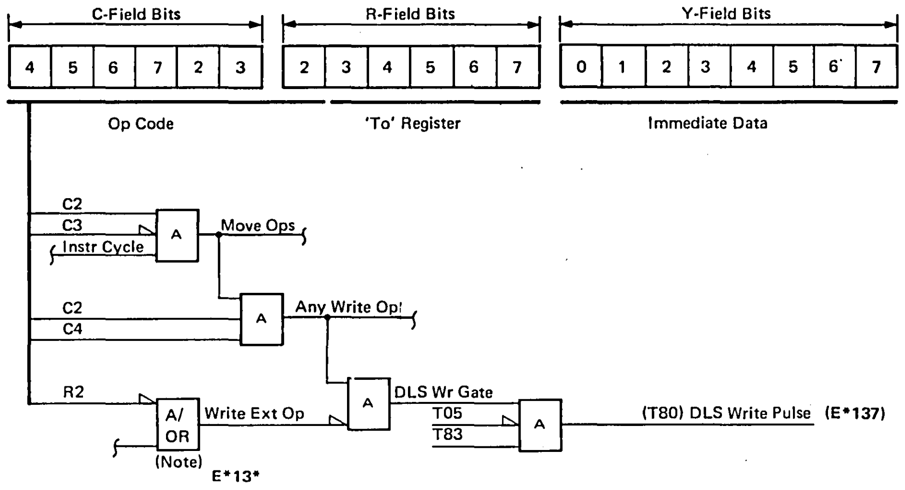
- Group 3: Process Cycle of LBI (load byte into DLS register specified by R-field and contents of addressed ZLS position)



- Timing

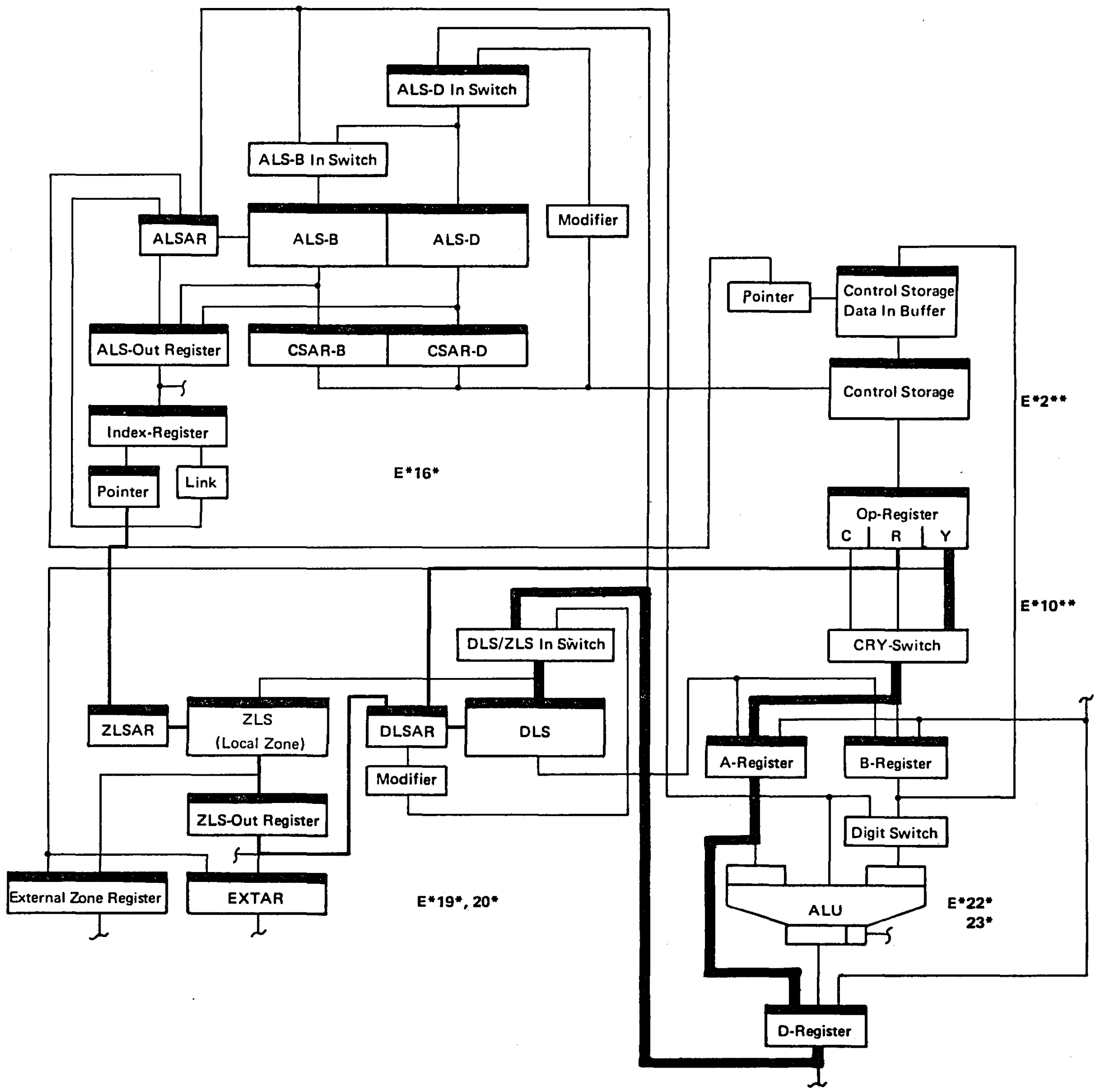


- Op-Register Layout and DLS Write Pulse Generation



- Control Signals for:
  CRY-switch
  ALU A-Register
  ALU D-Register
  see page 3-355

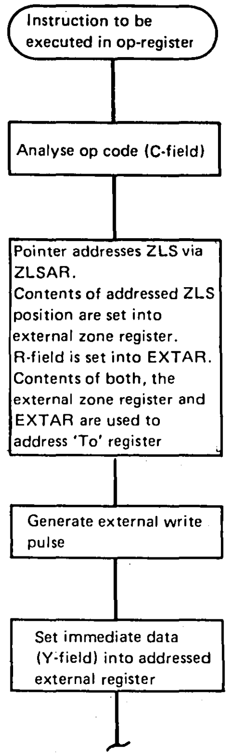
Note: This box represents a number of AND/OR blocks



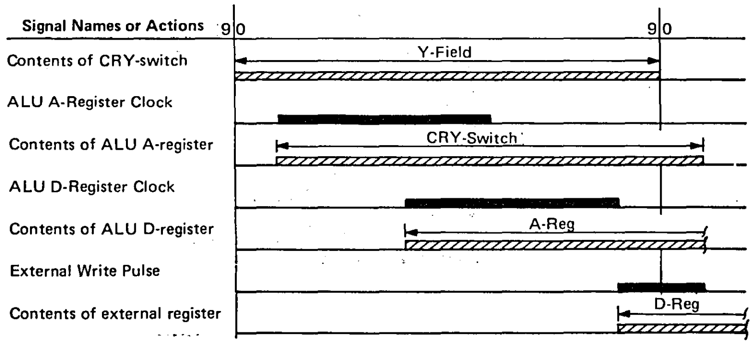
Microinstructions (continued)

LBI to External Register Process Cycle

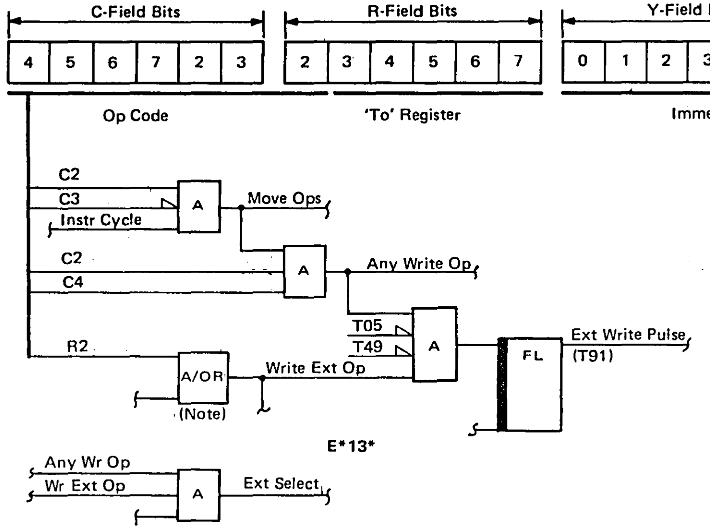
- Group 3: Process Cycle of LBI (load byte into the external register specified by external address)



- Timing



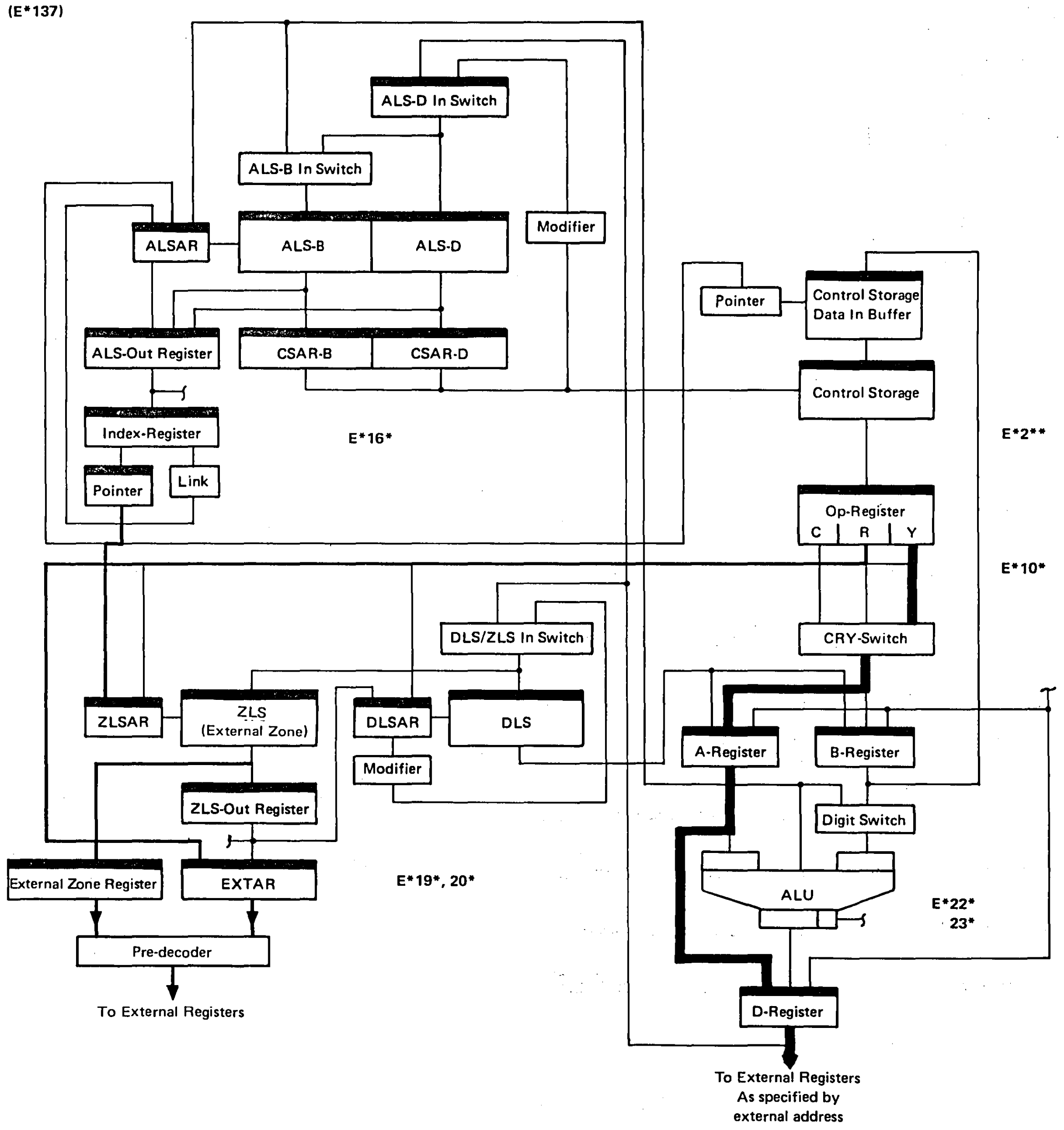
- Op-Register Layout and External Write Pulse Generation



- Control signals for:

CRY-Switch
ALU A-Register
ALU D-Register
see page 3-355

Note: This box represents a number of AND/OR blocks



This page is intentionally left blank

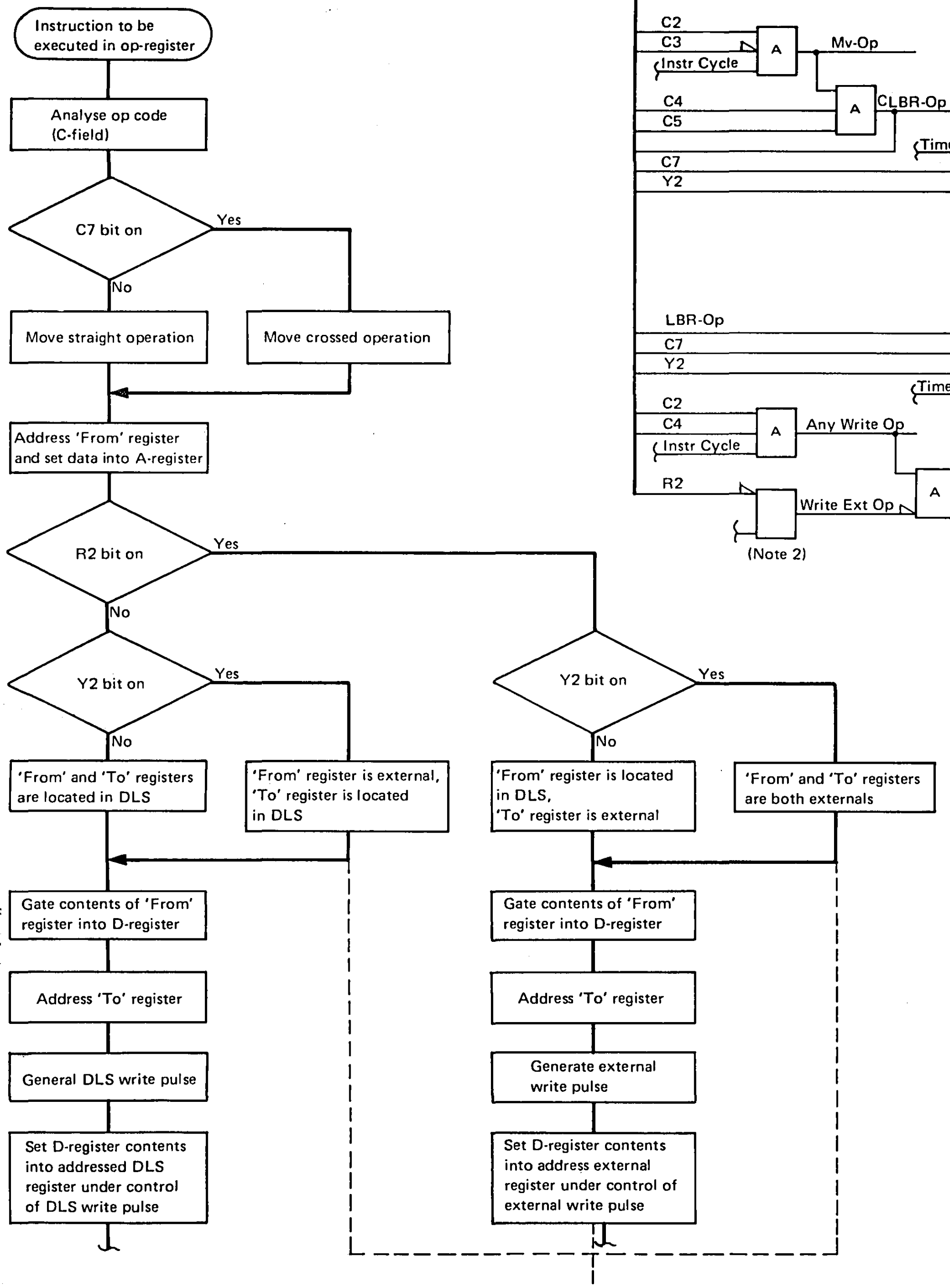
Microinstructions (continued)

MV, MVX Process Cycle

(Both Registers in DLS)

- Group 3: Process cycle of LBR operations.

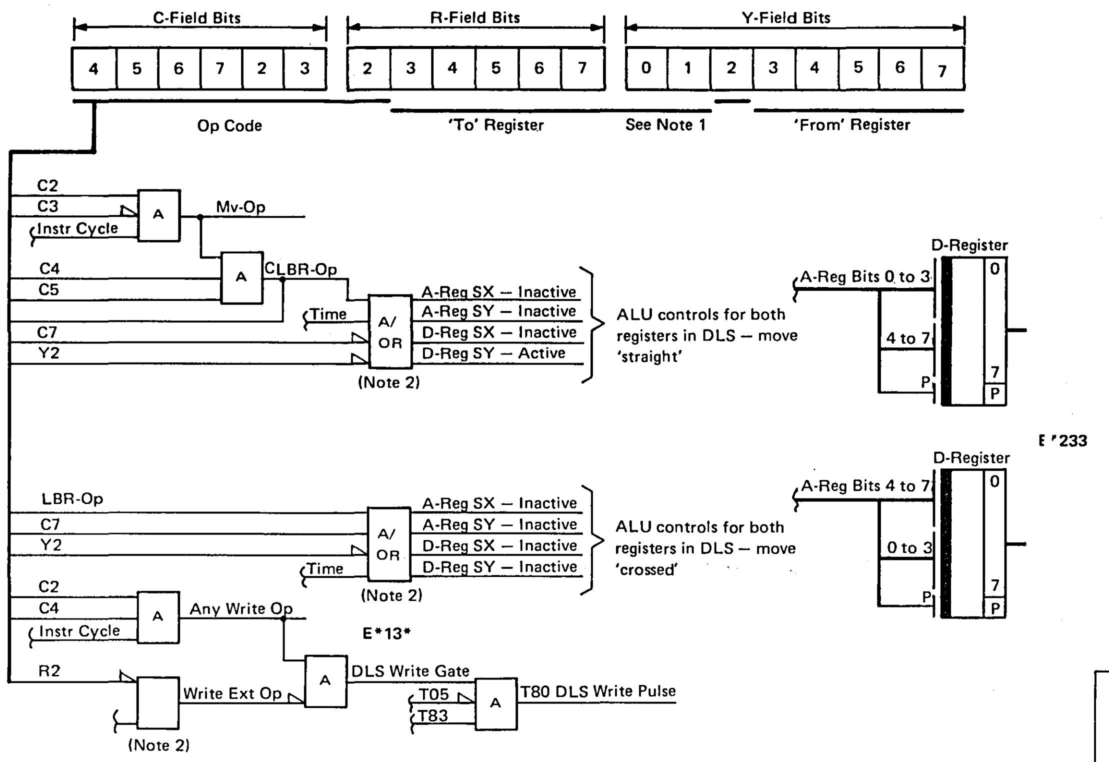
These operations allow the movement of the content of one register to another register by either 'straight' operation or 'crossed' operation.



For these operations, the following are shown on page 3-080:

- Op-Register Layout and Control Signal Generation
- Timing
- Data Flow

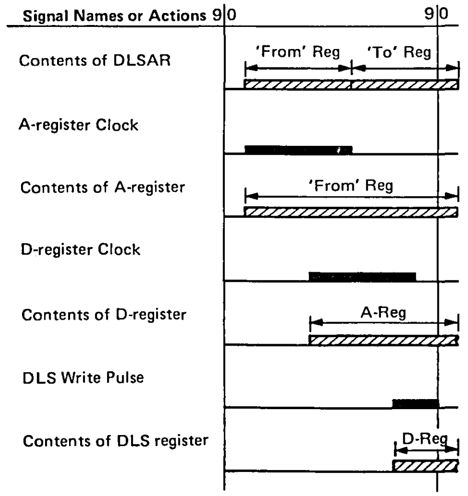
- Op-Register Layout and Control Signal Generation

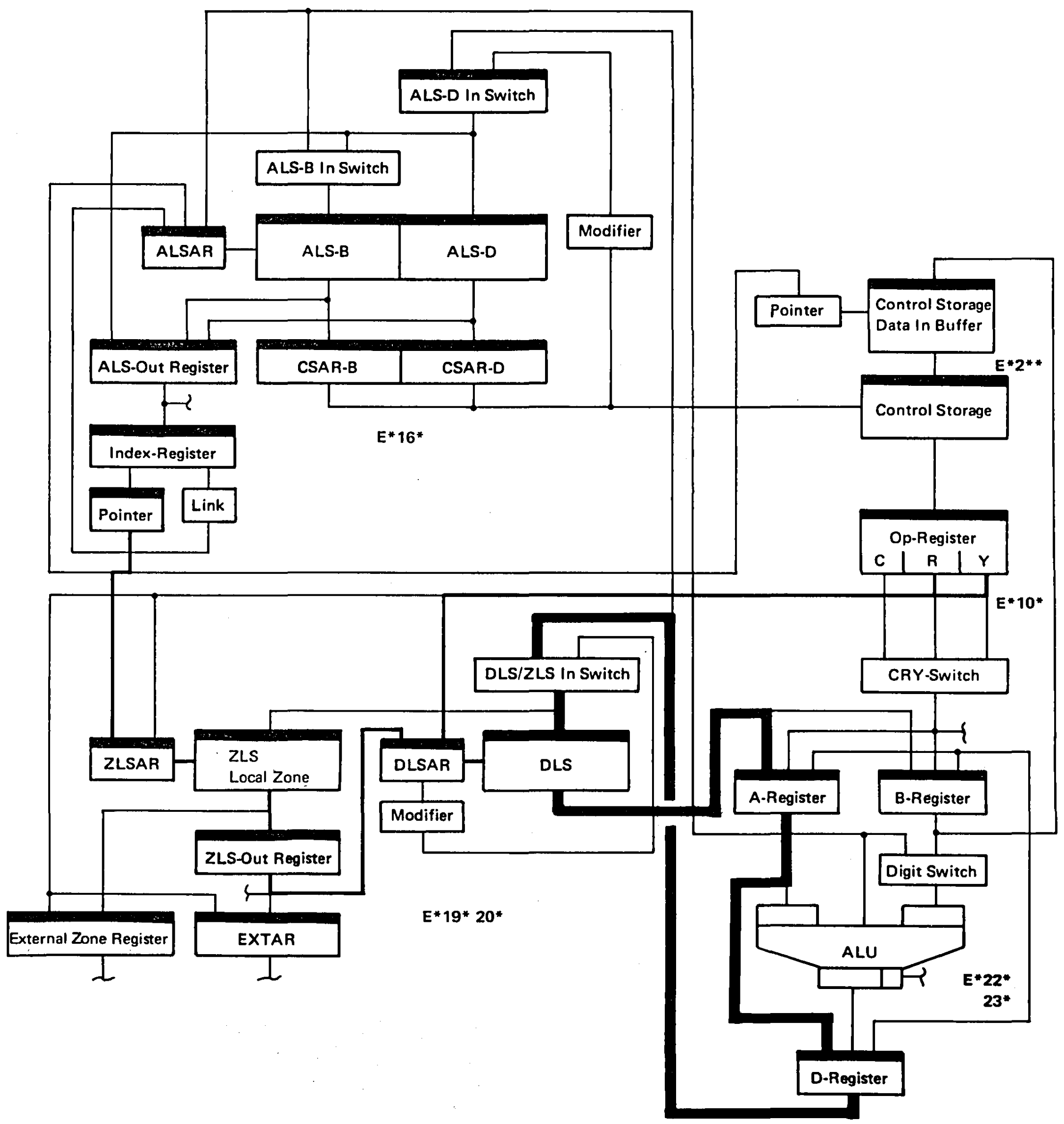


Notes:

1. Y0 – Suffix U bit, allows the changes of IARs after execution of the instructions
   Y1 – Has to be zero
2. This box represents a number of AND/OR blocks

- Timing

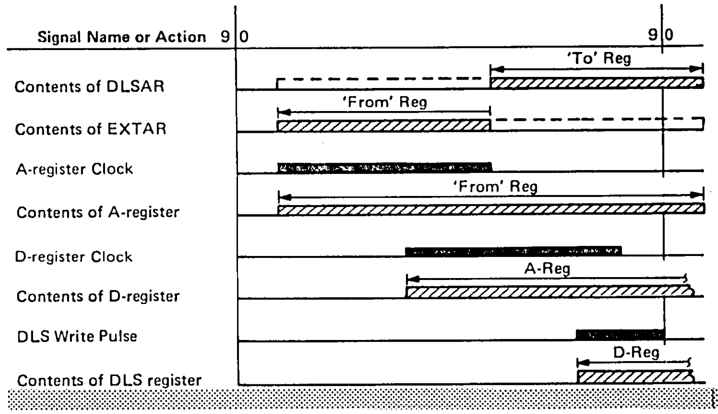




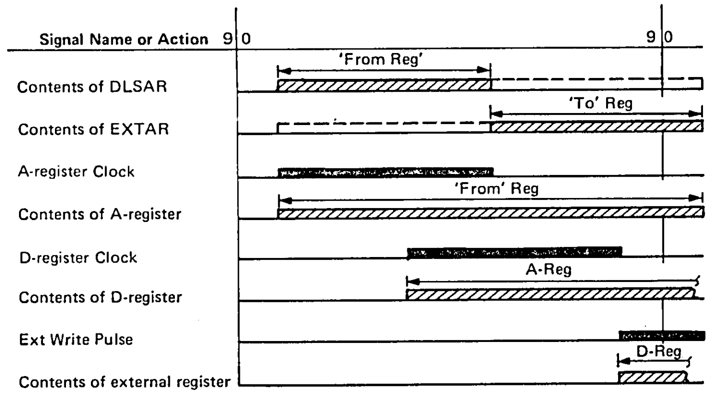
MV, MVX Process Cycle (continued)

(External Register to DLS Register, DLS Register to External Register, External Register to External Register)

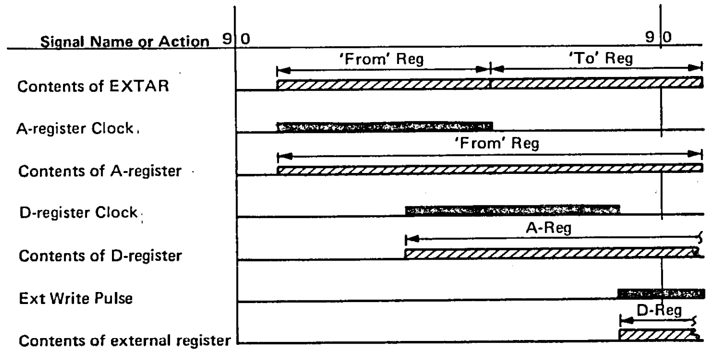
- Group 3: Process cycle of LBR operations (continued from page 3-375)
- Timing for Move External to Local Register



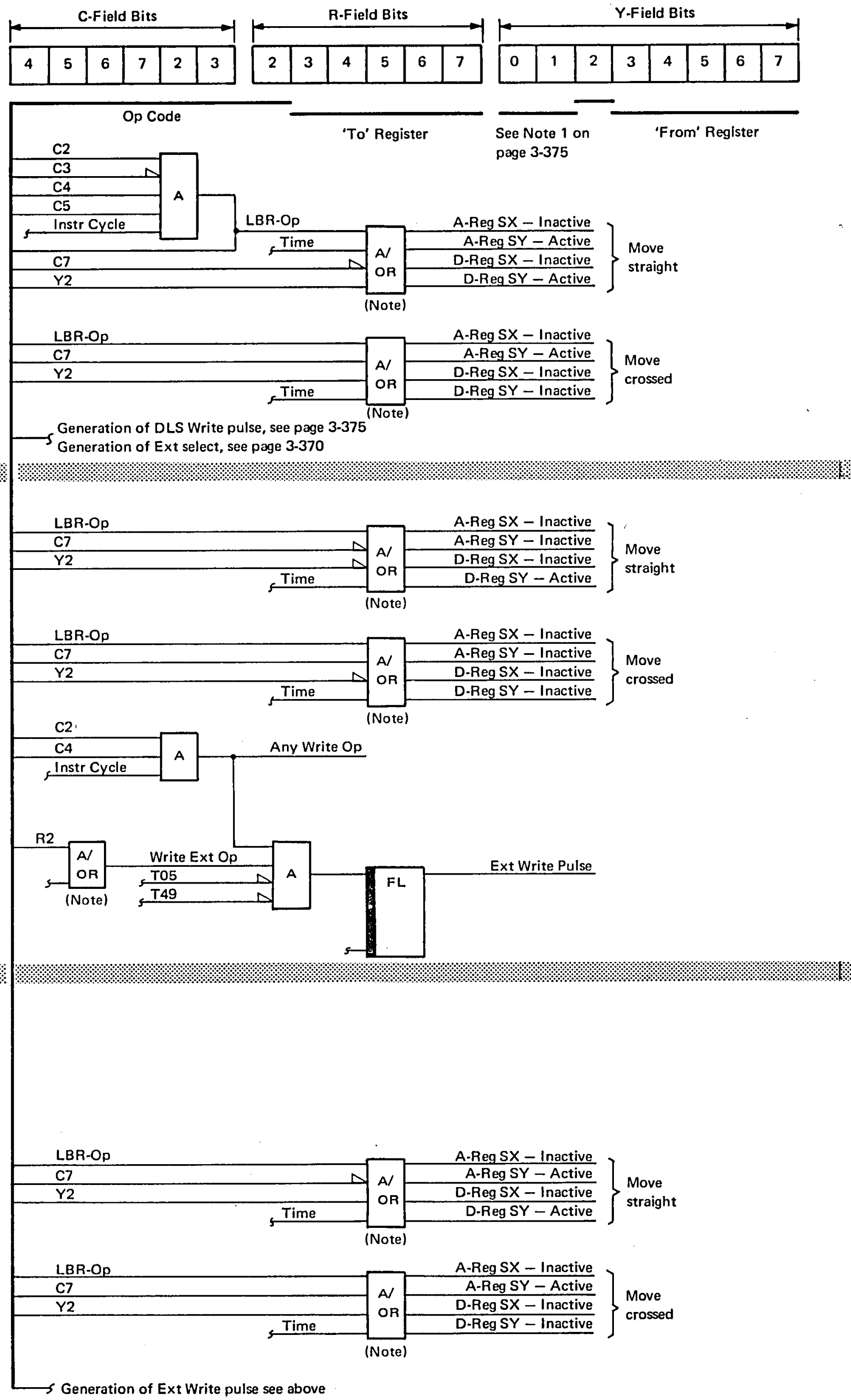
- Timing for Move Local to External Register



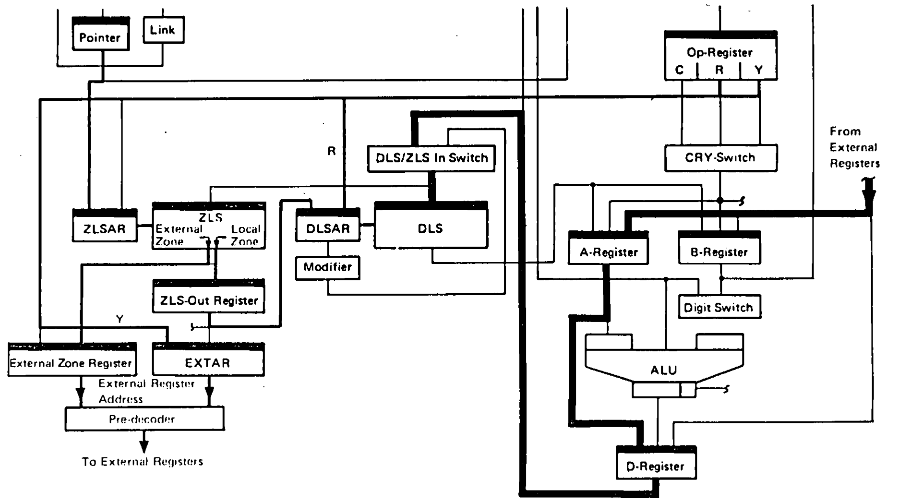
- Timing for Move External to External Register

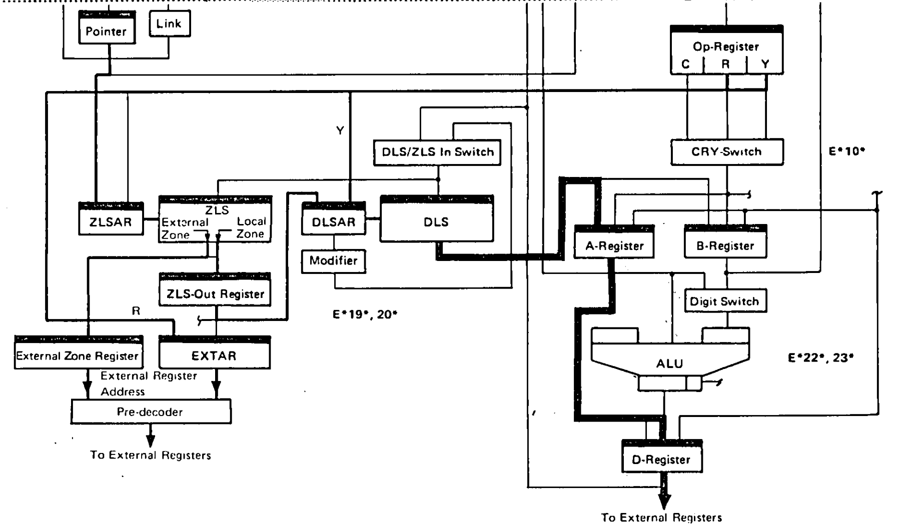


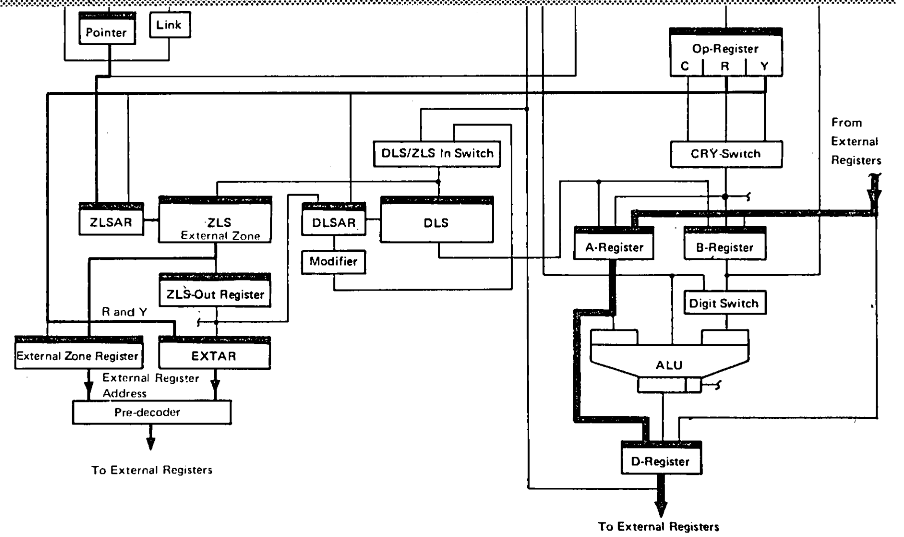
- Op-Register Layout and Control Signal Generation



Note: This box represents a number of AND/OR blocks

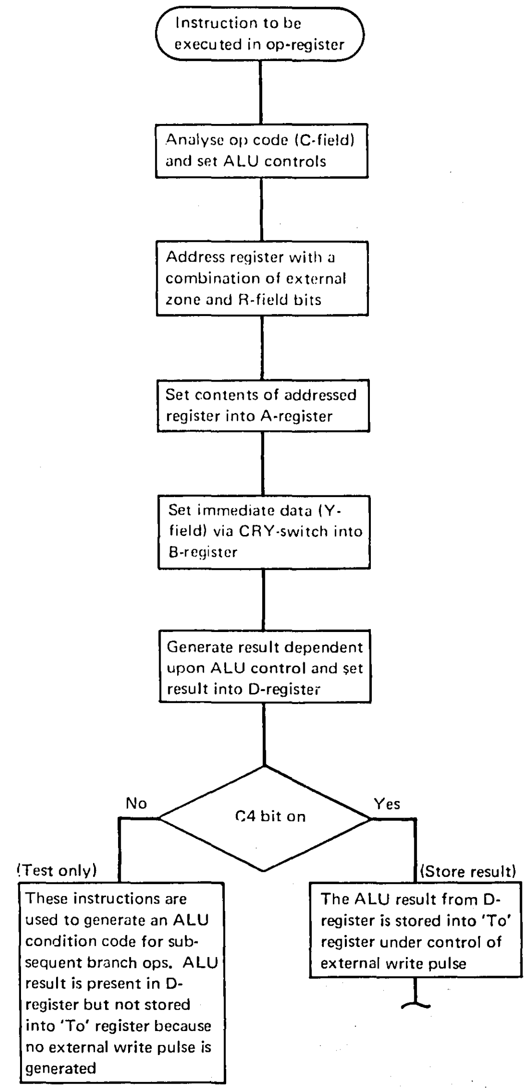




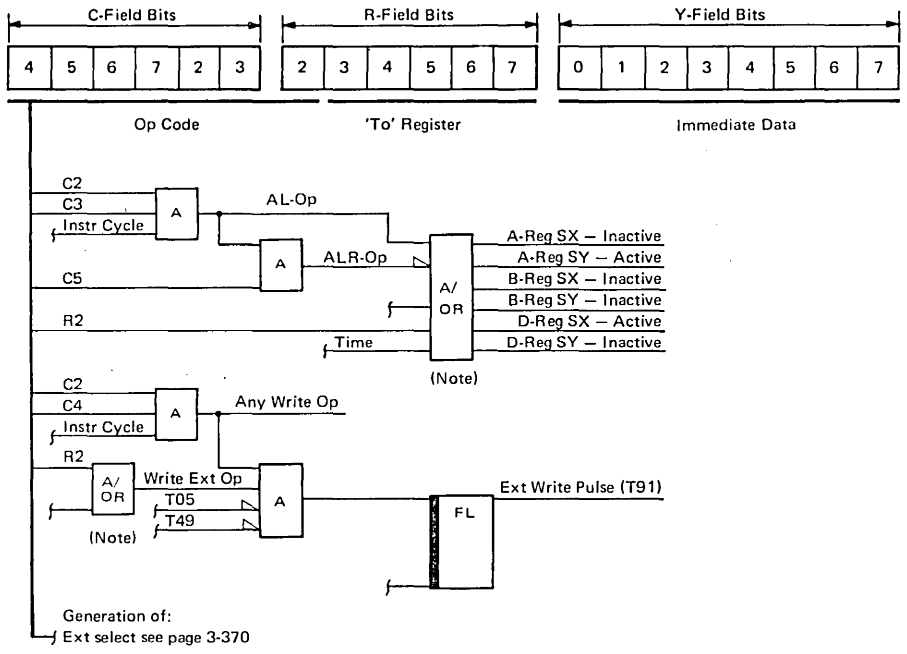


- Group 4: Process cycle of:

ADDI ANDI EORI ORI	Immediate data to external register and store result
TADDI TANDI TEORI TORI	Immediate data to external register without store result



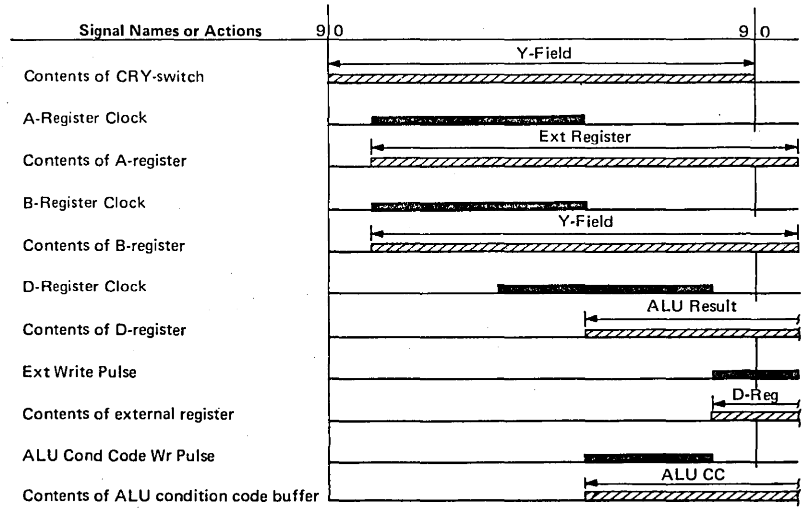
- Op Register Layout and Control Signal Generation

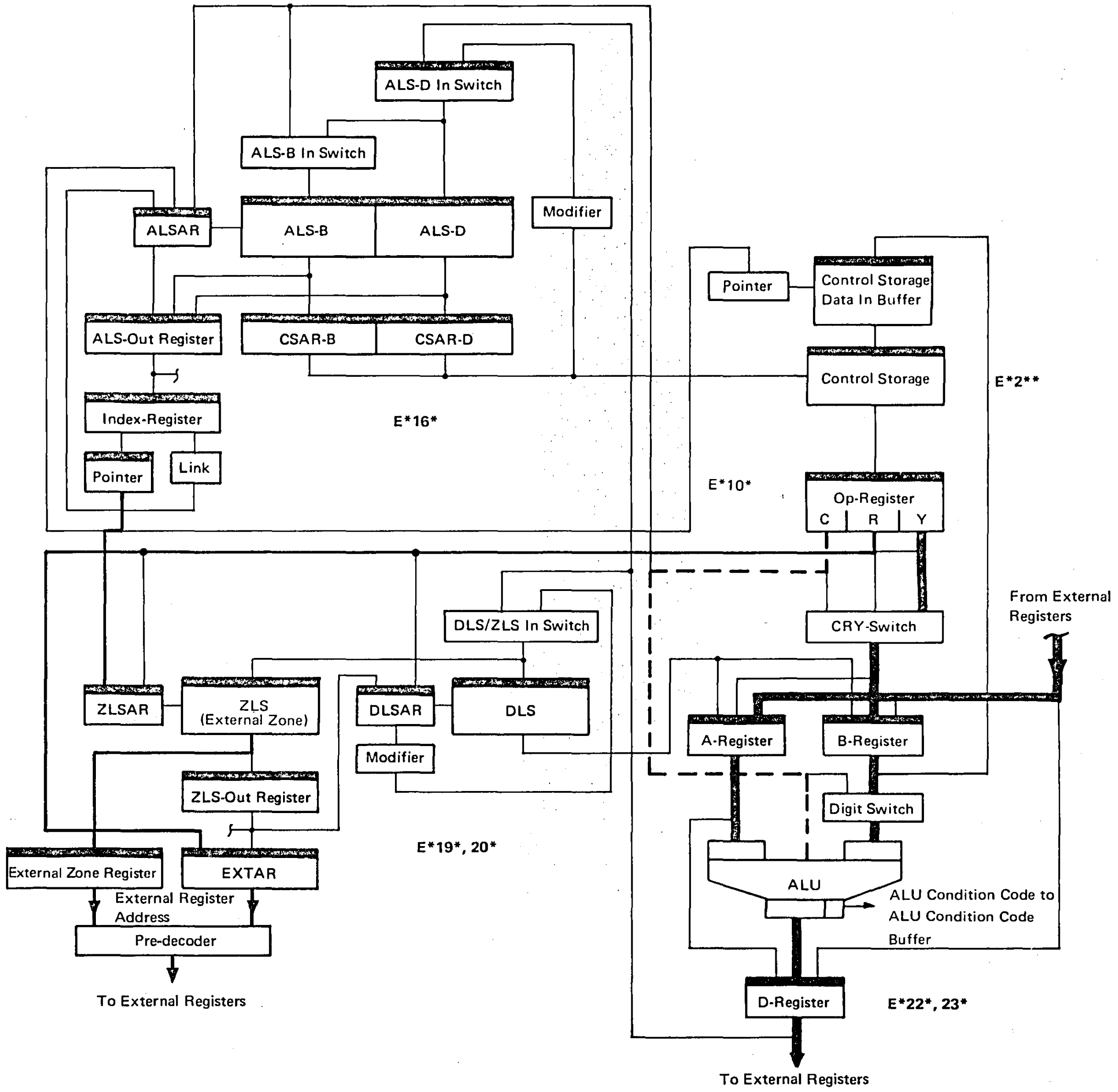


- Control signals for CRY-switch see page 3-410
- ALU functions see page 3-420

Note: This box represents a number of AND/OR blocks

- Timing





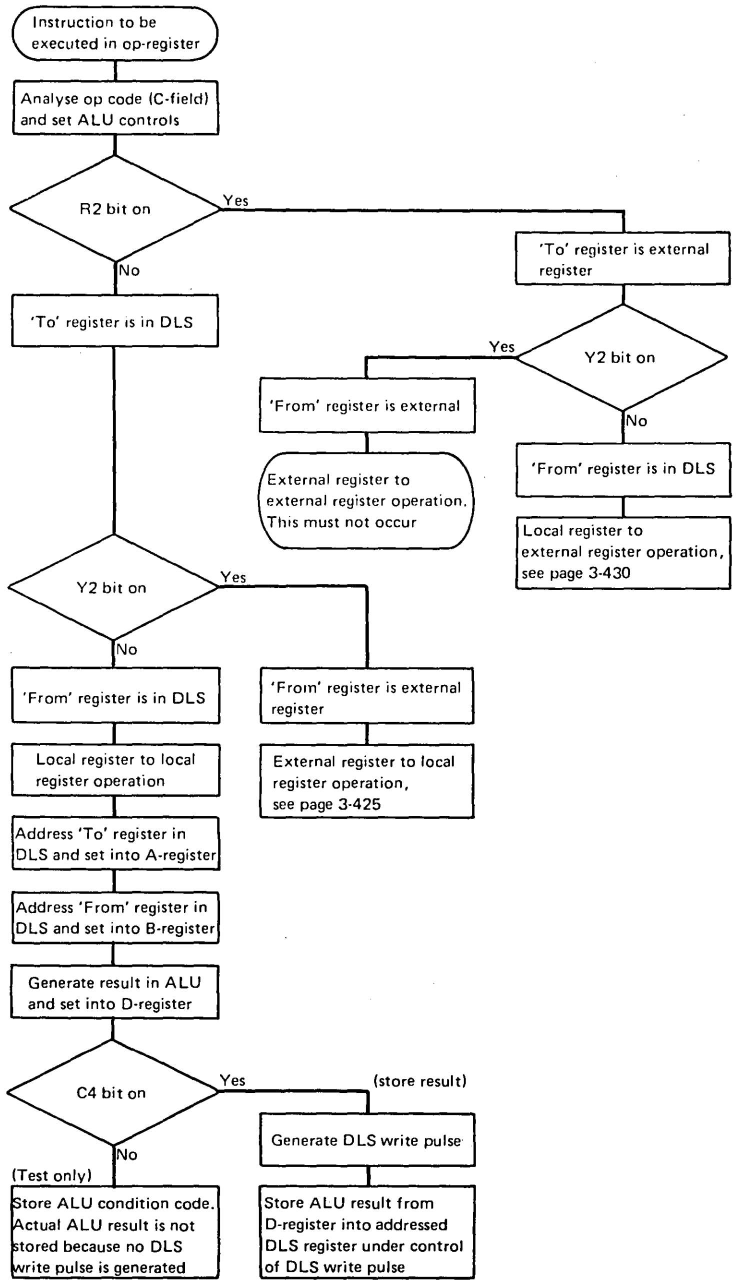
Microinstructions (continued)

ADD, AND, EOR, OR, TADD, TAND, TEOR, TOR Process Cycles

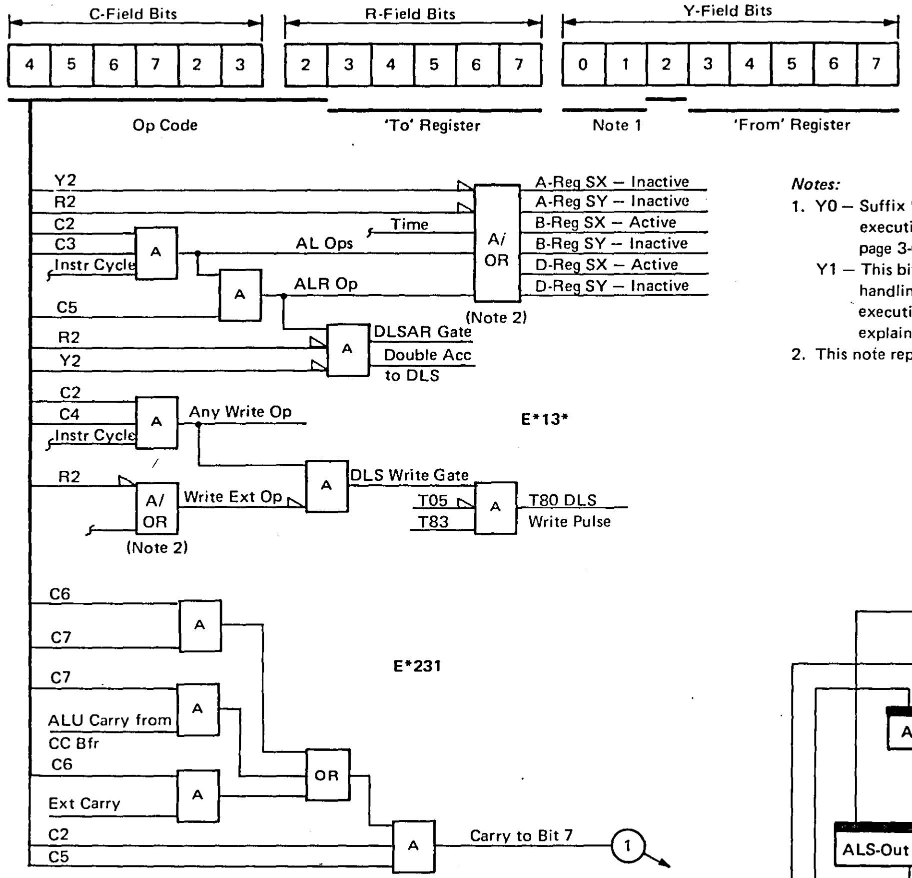
(Both registers in DLS)

- Group 4: Process Cycle of:

ADD AND EOR OR	Register to register with store result
TADD TAND TEOR TOR	Register to register without store result



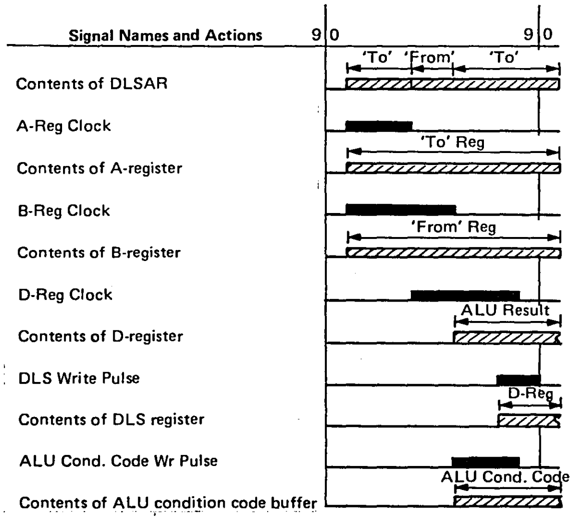
- Op-Register Layout and Control Signal Generation



Notes:

1. Y0 – Suffix U bit, allows the change of IARs after execution of the instruction. (See also page 3-040)
   Y1 – This bit defines special conditions for carry handling, that are considered with the execution of the ADD or TADD instructions, explained in Note 3 on page 3-030
2. This note represents a number of AND/OR blocks

- Timing

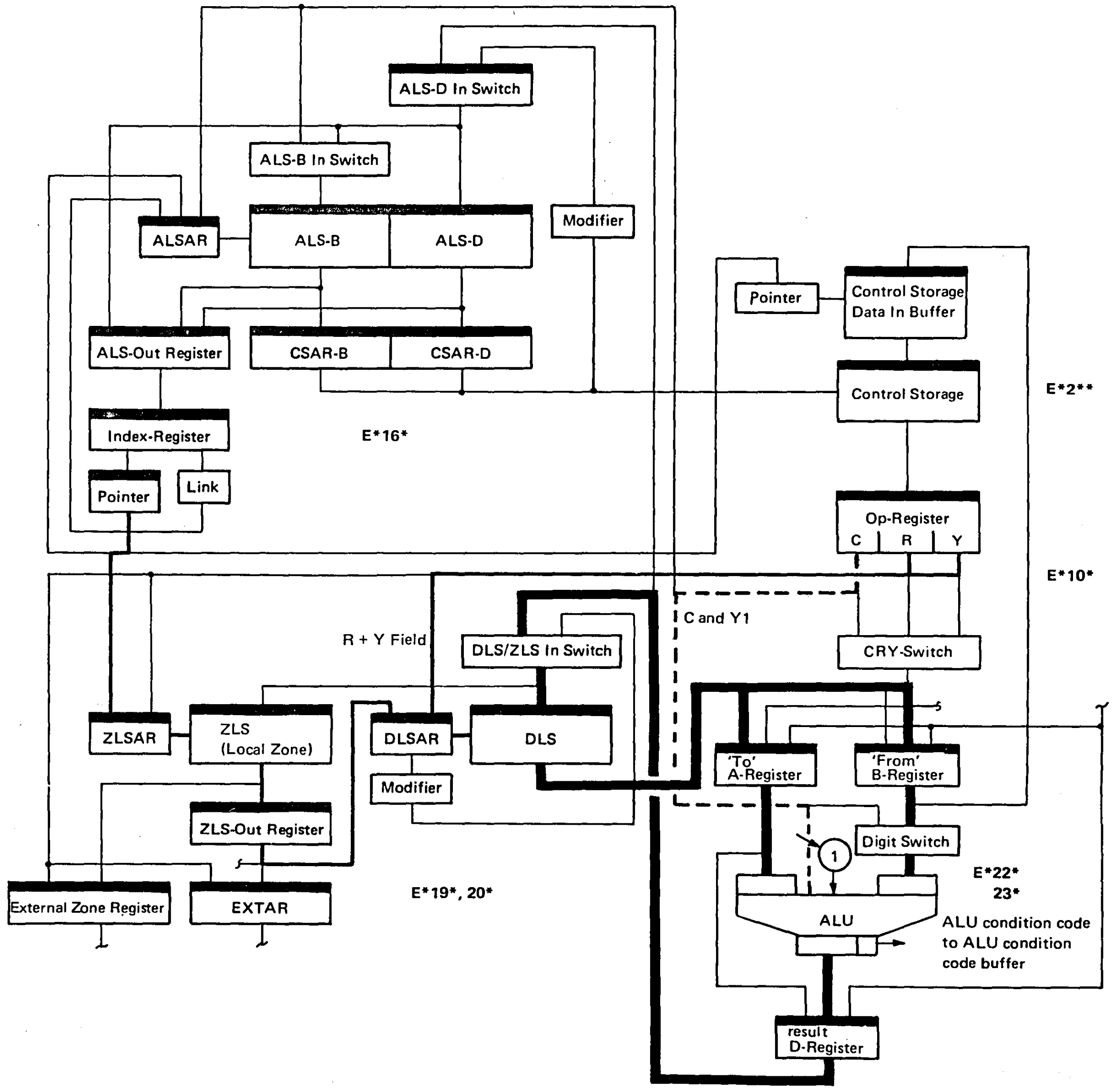


- ALU Functions:

ADD	A + B	A $\hat{=}$ 9	1 0 0 1
		B $\hat{=}$ 3	0 0 1 1
		C $\hat{=}$ 12	1 1 0 0
AND		A	1 0 0 1
		B	0 0 1 1
		Result	0 0 0 1
EOR		A	1 0 0 1
		B	0 0 1 1
		Result	1 0 1 0
OR		A	1 0 0 1
		B	0 0 1 1
		Result	1 0 1 1

- ALU conditions are:
  ALU result zero
  ALU carry

A number of branch conditions may be generated in connection with C-field bits 6 and 7. (See page 3-115).

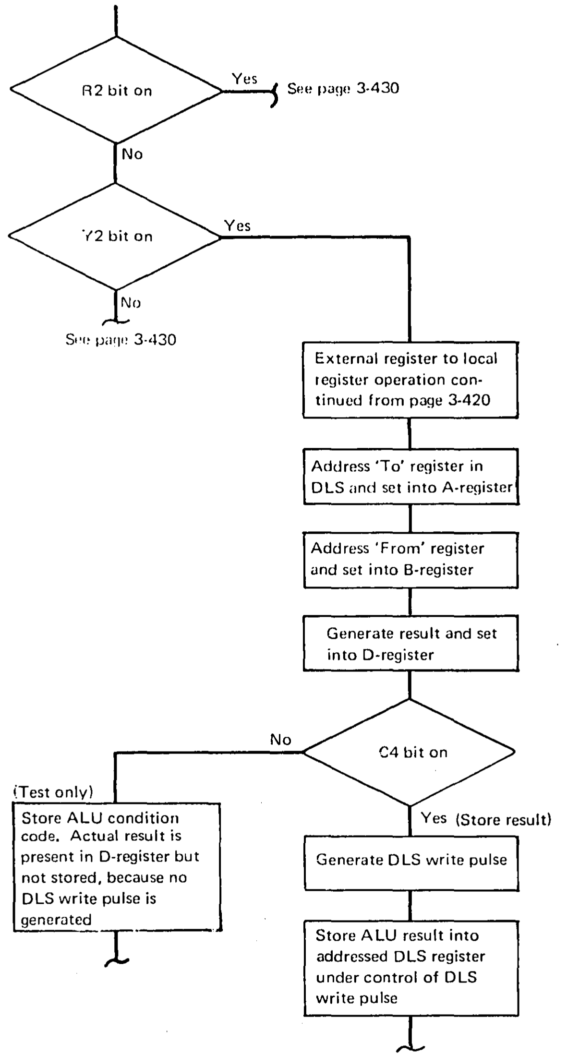


ADD, AND, EOR, OR, TADD, TAND, TEOR, TOR Process Cycles

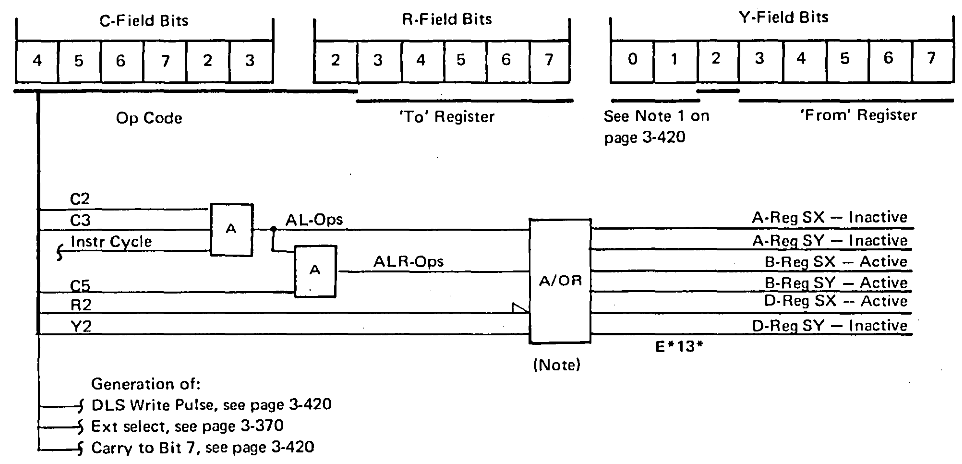
('To' register in DLS, 'From' register external)

- Group 4: Process cycles of

ADD AND EOR OR	Register to register with store result
TADD TAND TEOR TOR	Register to register without store result

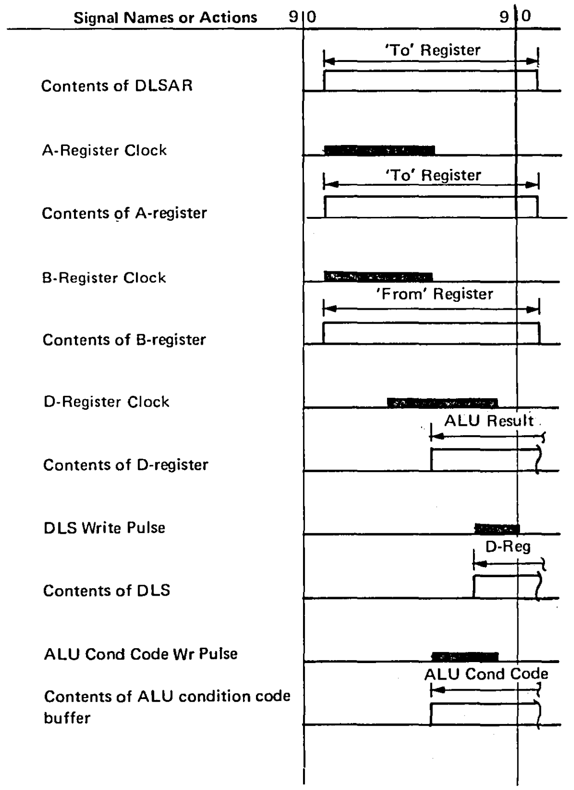


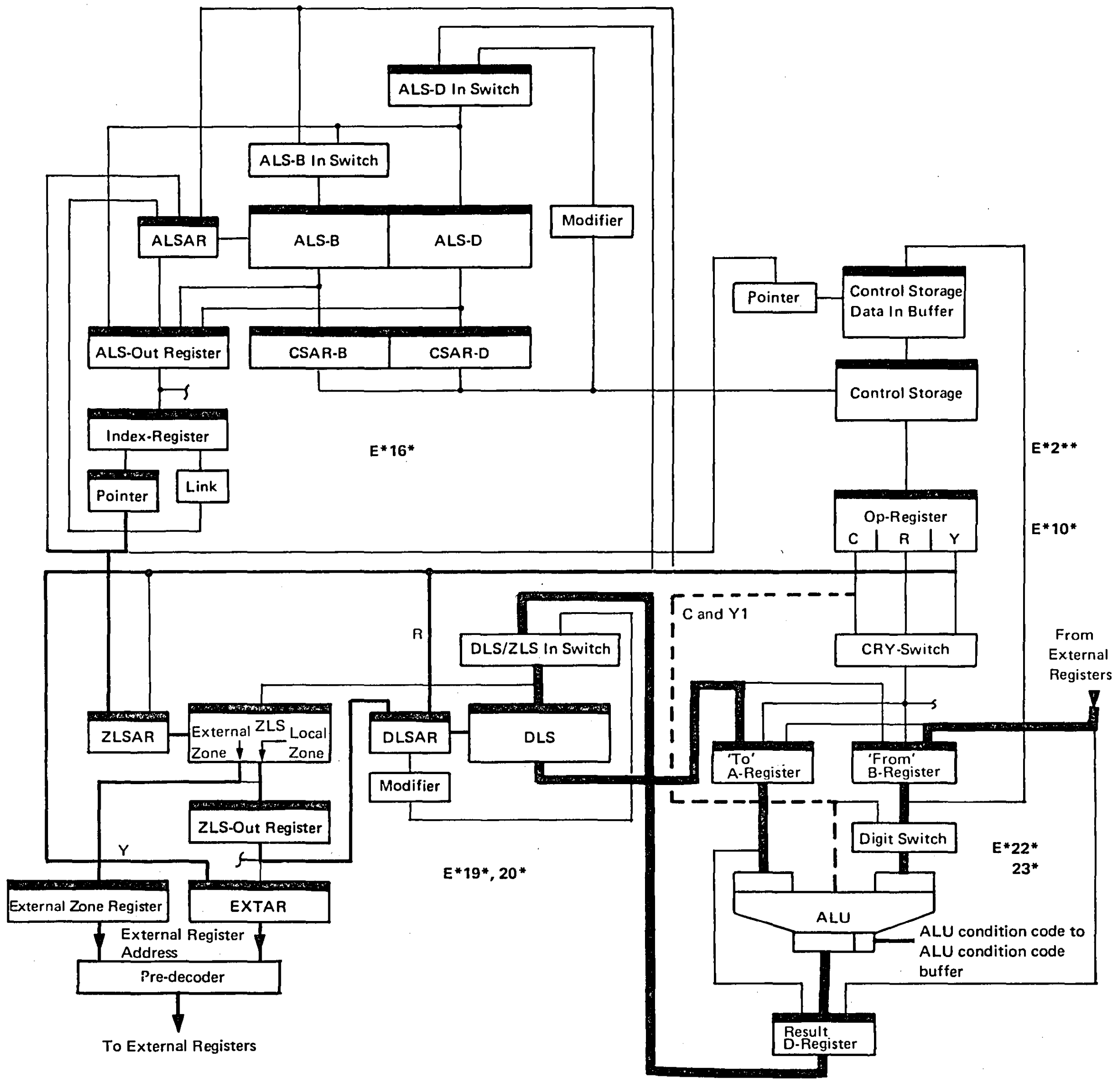
- Op-Register Layout and Control Signal Generation



Note: This box represents a number of AND/OR blocks

- Timing





Microinstructions (continued)

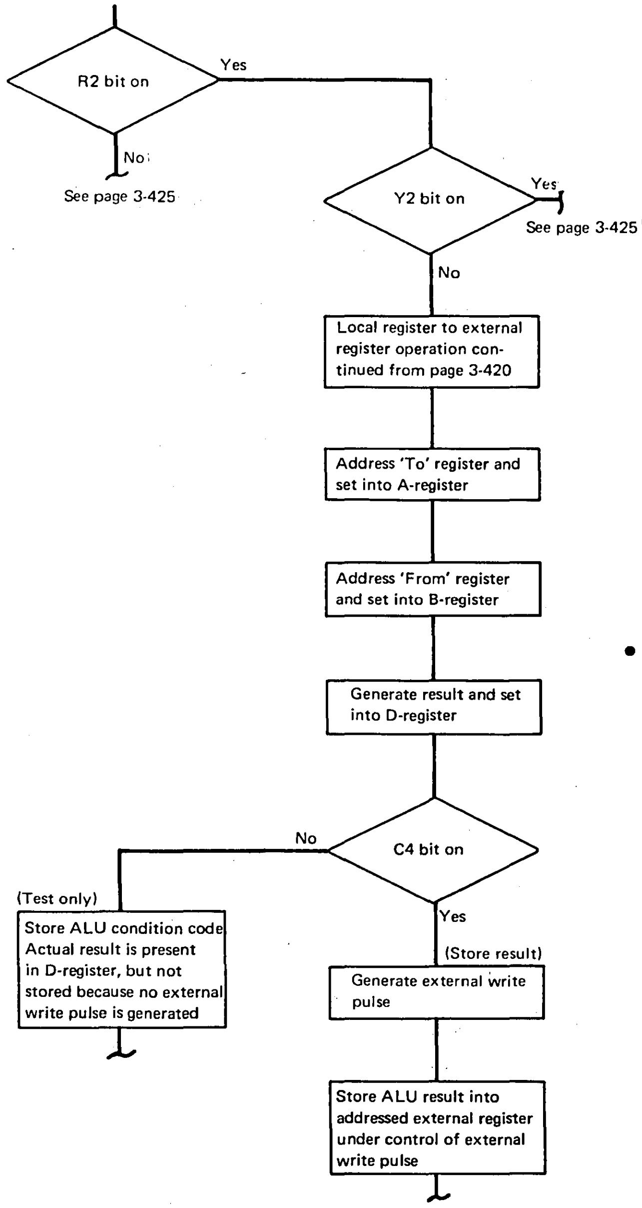
ADD, AND, EOR, OR, TADD, TAND, TEOR, TOR Process Cycles

('To' register is external, 'From' register in DLS)

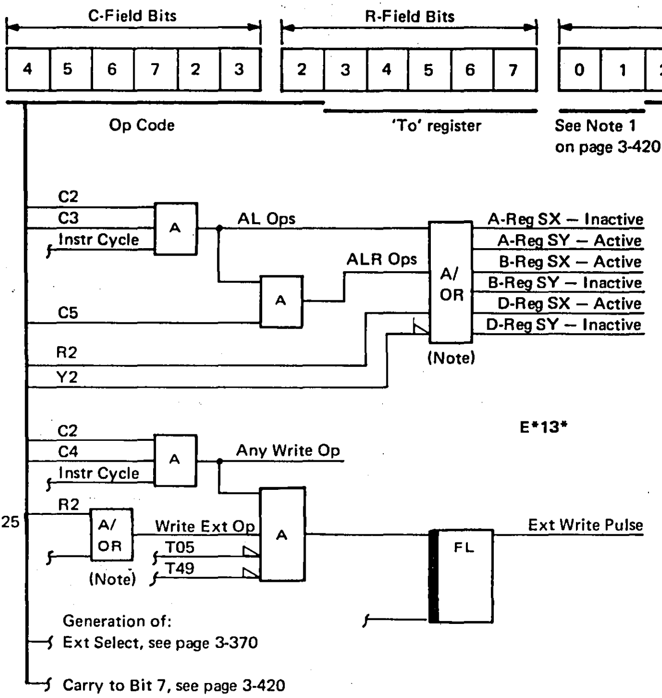
- Group 4: Process cycle of

  ADD
  AND
  EOR
  OR
  } Register to register with store result

  TADD
  TAND
  TEOR
  TOR
  } Register to register without store result

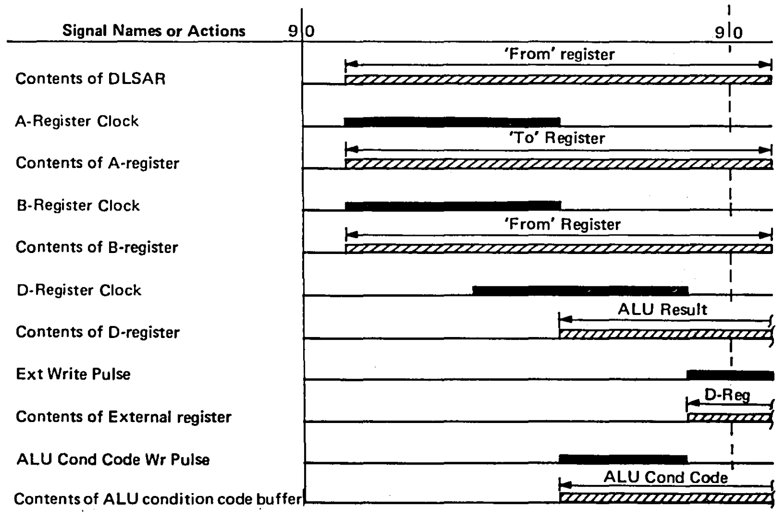


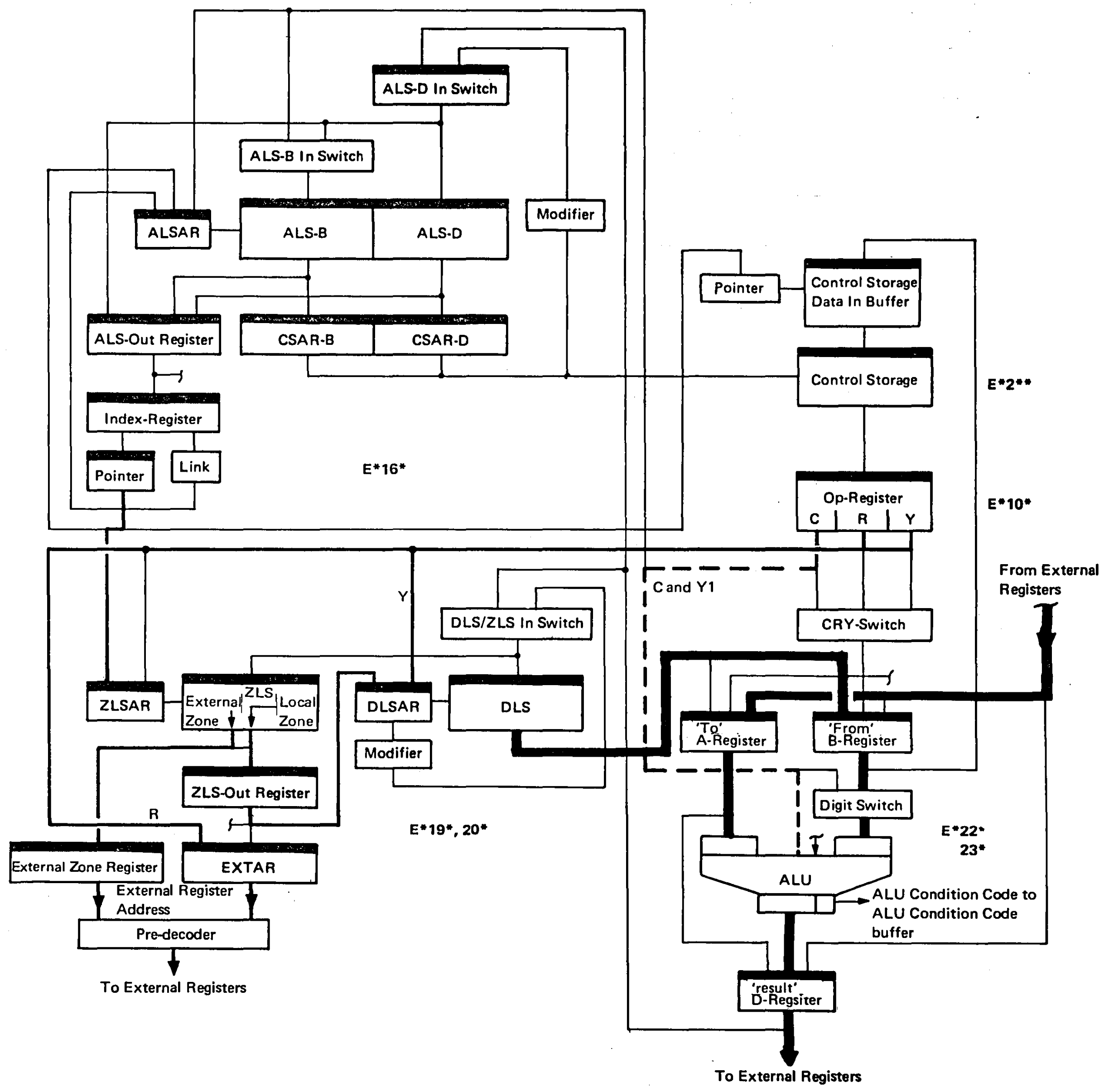
- Op-Register Layout and Control Signal Generation



Note: This box represents a number of AND/OR blocks

- Timing





This page is intentionally left blank

Microprogram Control

Cycle Timing and Arrangement

The diagram shows how an instruction is fetched and how addressing is carried out for access and process cycles.

The operation of IOPs is microprogram controlled. Execution of the individual microprograms follows the same scheme, because all IOPs are of the same structure.

All microinstructions, except those that specify data load/store operations, are executed in one cycle.

Each process cycle is preceded by an access cycle as shown in the following table:

Access cycle n	Access cycle n+1
process cycle n-1	process cycle n

Access cycle: to read out the instruction from control storage into op registers.

Process cycle: to execute the microinstruction.

Overlapping of access and process cycles (except for the very first and very last instruction) is the reason for process cycles being followed by process cycles.

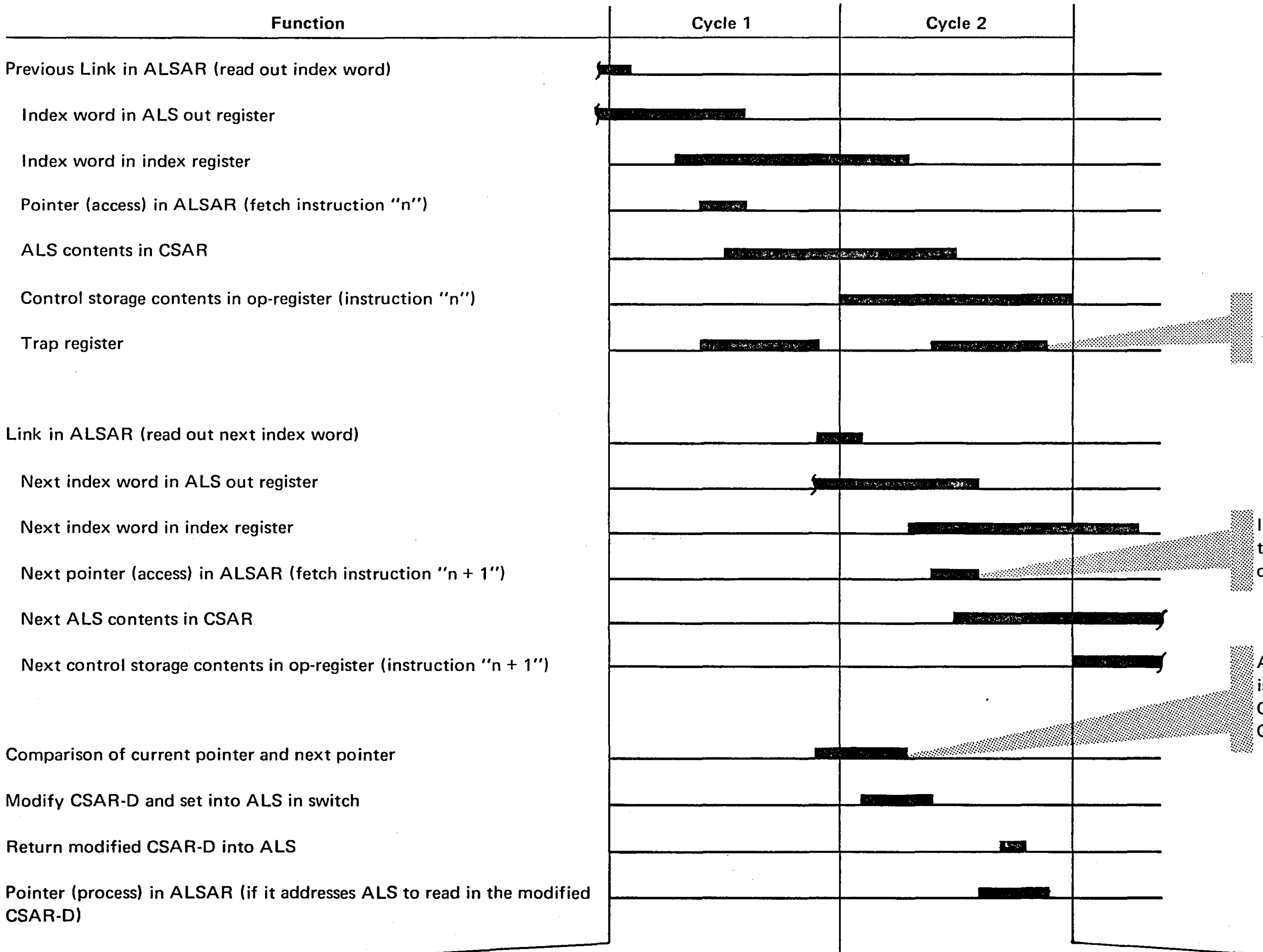
Cycles "n" belong together and represent access and process cycles for instruction "n"

Cycle "n-1" represents process cycle of instruction "n-1", preceeding instruction "n"

Cycle "n+1" represents access cycle of instruction "n+1" following instruction "n".

If time sliceing mode is "on" access and process cycles "n" may belong to one routine, while access cycle "n+1" and process cycle "n-1" may belong to other routines.

Cycle Timing



During this time, the link may be modified. This is early enough to have the new link address ready for the next access cycle

If instruction "n" is a data-fetch instruction or a store instruction, the process pointer is used instead of the access pointer to read out the data address

At the beginning of each cycle, the current pointer (index register) is compared with the new pointer (ALS-out register).
Compare equal specifies *Normal Mode*
Compare unequal specifies *Time Slice Mode*

Access Cycle for Instruction "n"

During this cycle, the link portion of the preceding index word addresses the ALS. The index word is read out and set into the index register. The pointer portion is now used as access pointer to address ALS and fetch the address of the next instruction that is to be executed. The contents of ALS are set into CSAR, which addresses control storage to set instruction "n" into the op-register. At the end of this cycle, the link portion is set into the ALSAR to fetch the next index word

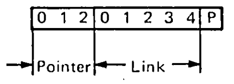
Access Cycle for Instruction "n + 1" and Process Cycle for Instruction "n"

During this cycle, the next index word is fetched as described under "cycle 1". Simultaneously, the CSAR content is modified, returned to ALS, and instruction "n" is executed. The pointer is now used as process pointer and is set into ALSAR to fetch the control storage address when data has to be fetched or stored. The process pointer simultaneously addresses the control storage data in buffer, the ALU condition code buffer, and the ZLS. This enables addressing of local and external registers and allows the storing of results and conditions of the current process cycle.

Execution of IOP Microprograms

The execution of IOP microprograms is controlled by index words.

These index words consist of a 3-bit wide pointer and a 5-bit wide link portion. The format of index words is as follows:



The pointer always indicates an IAR. This IAR contains the control storage address of the instruction that is to be executed next.

The link leads to the next index word.

Index words are stored in ALS.

The results from comparing pointers (current pointer in index register with new pointer in ALS-Out register) define the mode.

Compare equal = normal mode, compare unequal = time slice mode means that in normal mode, all pointers point to the same IAR. In time slice mode, pointers point to different IARs.

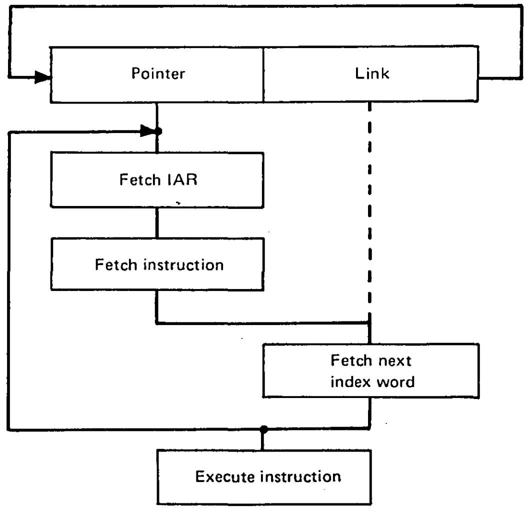
Additionally, both modes, may be trapped, see Page 3-550.

For arrangement of access and process cycles see Page 3-500, timing of access and process cycles see Page 3-500, actions performed during access and process cycles see Pages 3-050, 3-055, 3-500.

- Normal mode

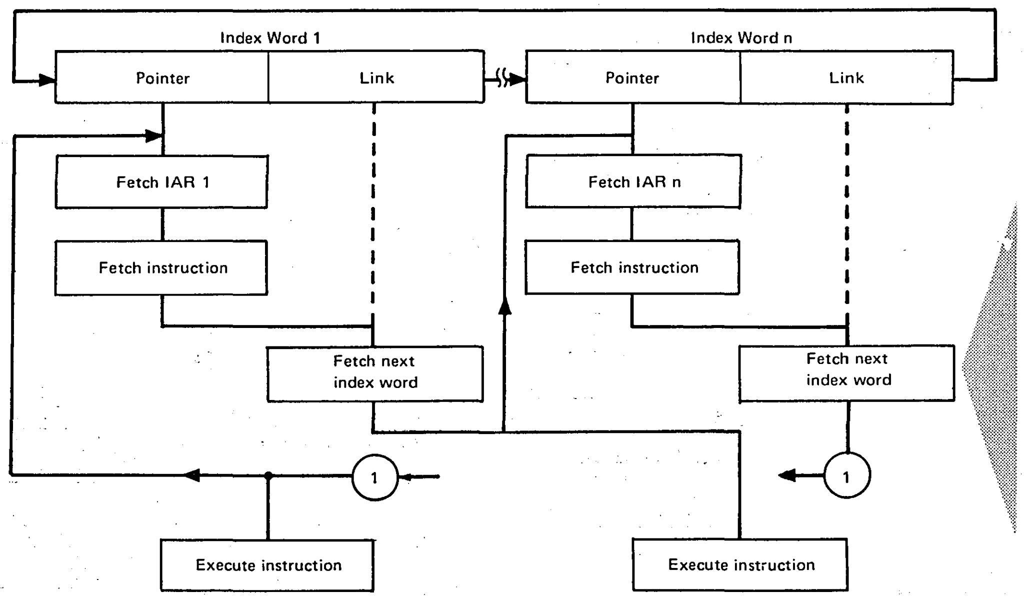
In normal mode one program or one routine is executed under control of at least one index word. A chain of index words is not absolutely necessary.

If only one index word is used, execution of the microprogram would be as follows:



The Link leads to the address of its own index word. This means microprogram control loops on one index word.

If a chain of n index words (maximum is = 16) is used, execution of microprogram would be as follows:



Each link leads to the address of another index word, while all pointers point to the same IAR.

Working with a pre-determined chain of index words, allows the setting up of timings, because index word number one is always used after a fixed number of IOP cycles, and each cycle is of a defined length. (e.g. time out conditions).

- Time slice mode

Time slicing allows multiprogramming. Time slicing is used to run several microprograms (or routines) stored in the control storage of one IOP simultaneously. This means the execution of one or more microinstruction of the program is followed by the execution of one or more microinstruction of another microprogram.

All programs then run at a reduced and variable speed, according to the pre-determined distribution of the IOP cycles.

The distribution of the IOP cycles is carried out by the microprogrammer. The program sequence is defined by the index words.

Here the pointers are different and may be considered as program numbers or program identifiers.

The link too, leads to the next index word.

With other words = The chain of index words is used in the same way as in normal mode, but in particular the pointers point to different IARs. This means that during one cycle an instruction of one program is executed, and during the next cycle an instruction of another program is executed.

Circuitry necessary to implement time slicing is shown on Page 3-520 (time slice mechanism).

The independency of pointer and link allows any sequence in the execution of program steps.

The number of index words required, depends upon the number of microinstructions to be executed in a repetitive sequence.

The resulting chain of index words is also called "time slice period".

For regular programming a maximum of 16 index words is provided.

Versions in which time slicing may be used are shown on Page 3-530.

Cycle timing for access and process cycles is shown on Page 3-500.

Microprogram Control (continued)

Time Slicing Mechanism

According to the internal arrangement of the following registers up to 8 different microprograms may be executed.

ALS	(4 x 8 halfwords)
ZLS	(4 x 8 bytes)
ALU Cond. Code BFR	(1 x 8 positions)
Ctrl Stg Data In BFR	(2 x 8 positions) see Note
Mode BFR	(1 x 8 positions)

All consist of 8 or a multiple of 8 storage positions.

The diagram shows that one of the storage positions 1 to 8 out of each group is assigned to one program.

These assigned storage positions are addressed by one pointer, representing the program number or program identifier.

1 After an index word is read into ALS-out register, the pointer first addresses mode buffer position

2 The contents of mode buffer position and pointer then addresses ALS position (access pointer)

3 Addresses ALS position according to "mode" either:
Main IAR or
Sub IAR or
Data address register is selected.

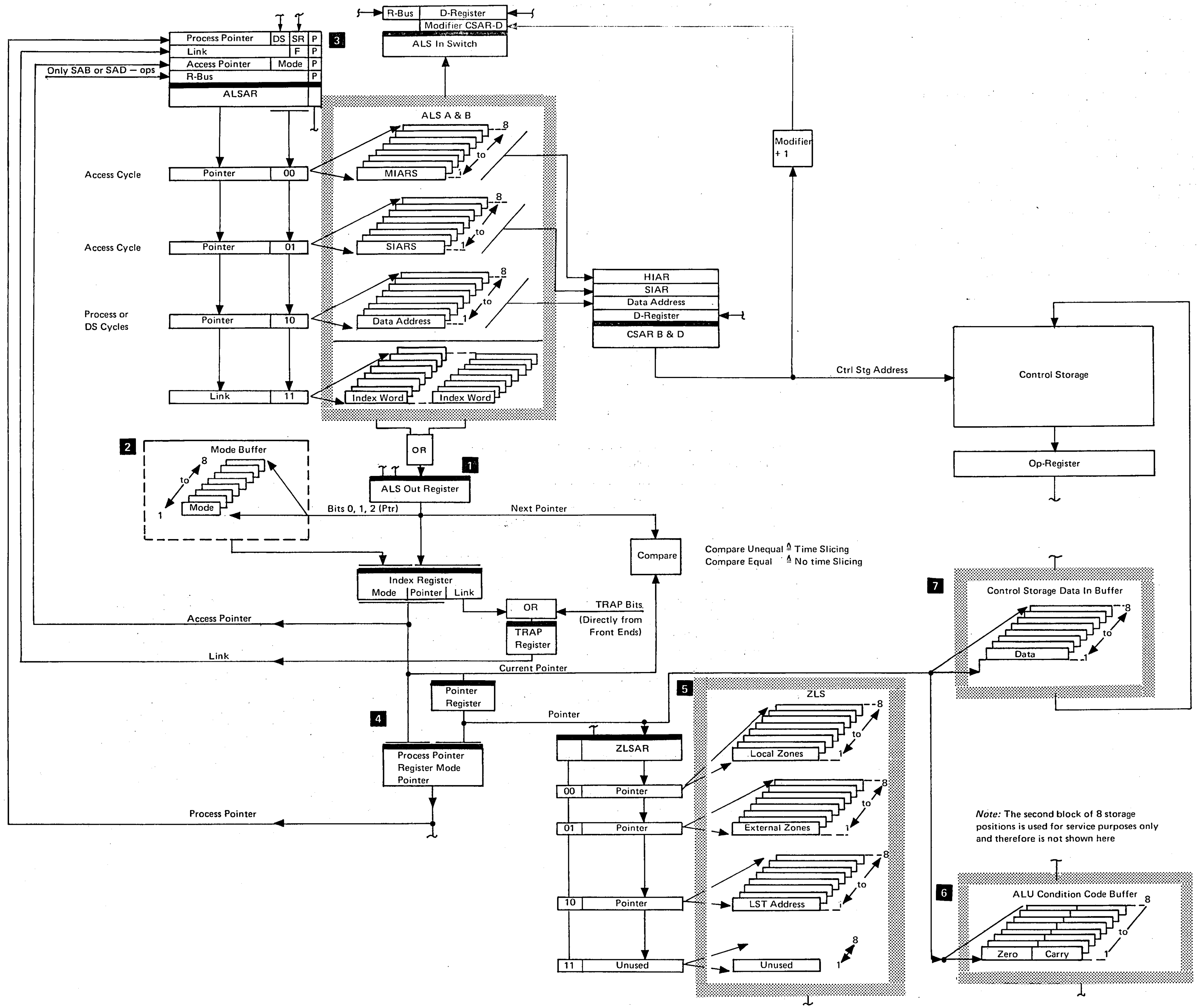
4 Subsequently, the pointer is set into pointer register and process pointer register. The pointer is held for the process cycle to address ZLS position.

5 Addresses ZLS position, according to ZLSAR bits 0 and 1 either local zones or external zones or LST address register is selected. Both, local as well as external zones together with op-register contents, are used to address DLS or external work registers. During DS cycles work register addressing is carried out by the contents of LST address register.

6 Addresses ALU condition code buffer position to hold ALU zero and carry conditions (besides the actual result) for later branch operations.

7 Addresses control storage data in buffer that works as an interim storage for data, to be stored during DS cycles.

ALS and mode buffer once more are addressed by the same pointer, to restore modified addresses and updated mode (process pointer).



Note: The second block of 8 storage positions is used for service purposes only and therefore is not shown here

Versions of Time Slicing

Execution of more than one program (or routine) can be carried out in two different ways:

- Fixed time slice period:

  A fixed number of index words is used to control execution of the different microprograms. The chain of index words remains unchanged.
  Depending upon time requirements cycle distribution is achieved by charging pointer portion of index words by the use of SAB or SAD instructions

- Variable time slice period:

  A variable number of index words control the execution of the different microprograms. The chain of index words is altered.
  Depending upon time requirements cycle distribution is achieved by changing the link portion of index words by the use of SAB or SAD instructions.

Fixed Time slice period

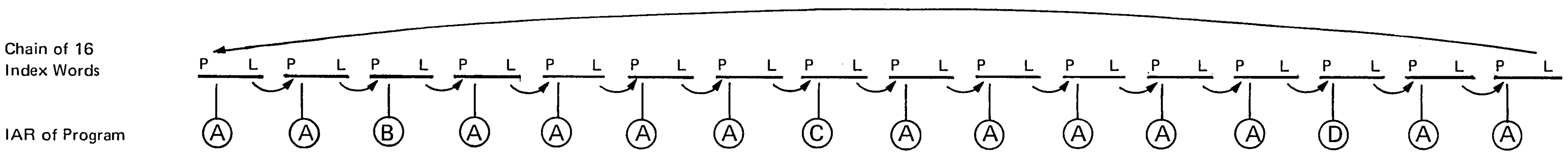
A common program (A) and three programs (B, C, D) which serve I/O devices A, B, and C run in time slice mode. The diagram shows that no device requires service at the moment. The 16 index words are used for program control. Each index word represents one sixteenth of the fixed time slice period.

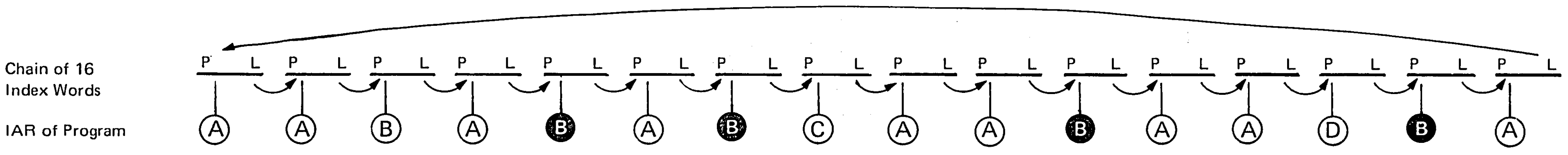
As soon as an I/O device requires service, a trap bit or request bit is activated. With this bit active a branch to the device program is performed. This device program starts with a series of SAB and SAD instructions to change the cycle distribution according to the time requirements of the device. The chain of index words remains unchanged, but in this example 4 pointers have been altered. This increases performance of program B, while the performance of program A is reduced.

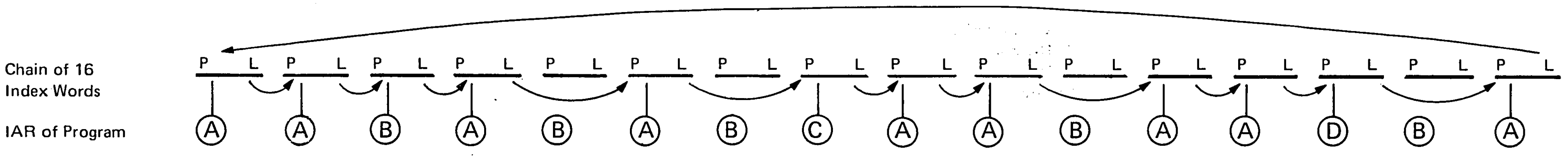
Variable Time slice period

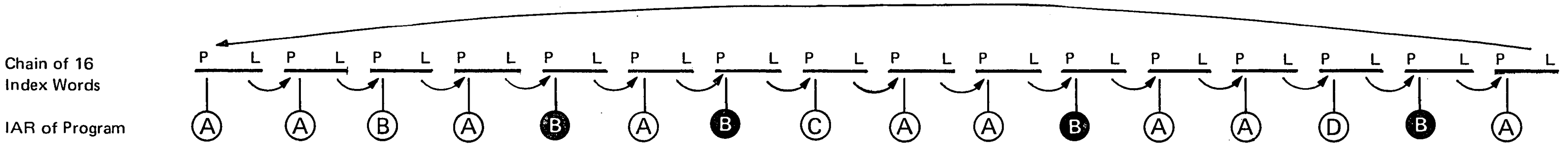
Same conditions as for fixed time slice period but with the exception that with a variable time slice period the index words do not represent fixed fractions.

Same conditions as above, but with the exception that the time slice period is changed as a result of changing 4 link portions by SAB or SAD instructions.









Microprogram Control (continued)

Trapping

The microprogram that is operating can be trapped both in normal mode and in time slice mode.

Trapping is used by the devices as soon as immediate service is required.

The three trap bits A, B, C are ORed with the link portion of the current index word.

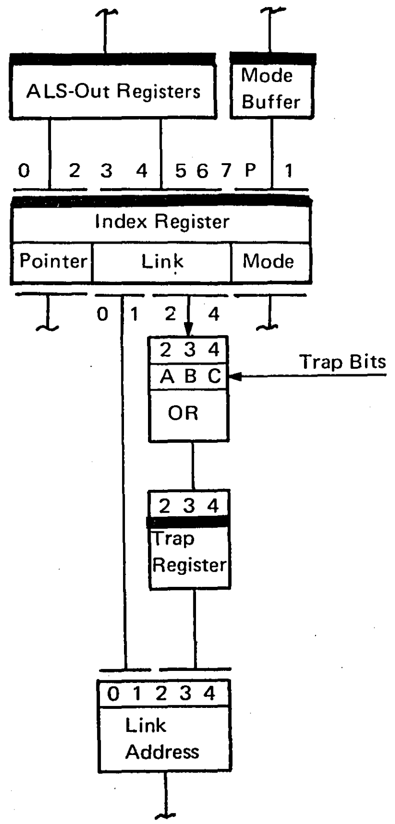
According to the bit pattern of the link, active trap bits either influence microprogram control, or do not.

This permits "to leave", or "to alter", or "to keep" the chain of Index words which permits jumping to another routine, changing the sequence in executing program steps, and continuing with the unchanged chain of index words.

With other words, program levels or priorities may be classified by these traps.

The modified link then leads to an index word which points to a routine that serves the needs of the requesting device.

Direct trapping permits three trap levels, but *coded trapping* permits up to seven trap levels.



Trap Bit Priority
A: Highest
B: Medium
C: Lowest

This page is intentionally left blank

IOP, SVP Communication

The following data flow shows in principle how the SVP data bus and SVP address bus interconnects the SVP and IOPs.

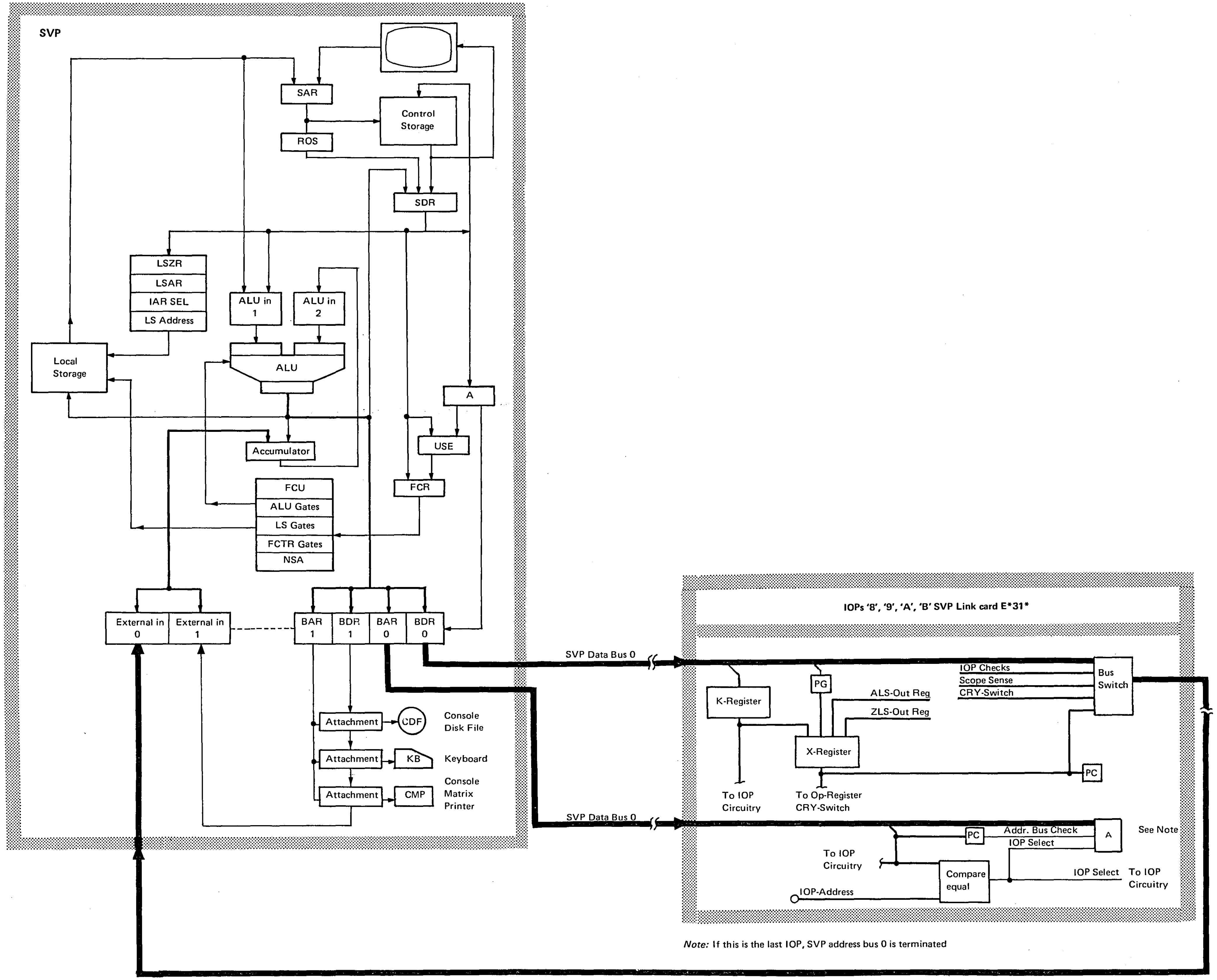
Both buses are one byte wide and are connected from SVP via MSC and IPU through the magnetic tape adapter and all IOPs.

The SVP data bus returns from the last subprocessor or adapter to the SVP, while SVP address bus is terminated in the last subprocessor or adapter.

The SVP data bus has no parity bit. The parity bit line is used to propagate the control strobe signal through the system as mentioned above.

The SVP buses are used for:
- IMPL
- System survey (idle sense)
- SENS operations
- CTRL operations

For more information, refer to pages 3-920, 3-930, 3-940, 4-050, 4-055.



Note: If this is the last IOP, SVP address bus 0 is terminated

SVP SENS Operation

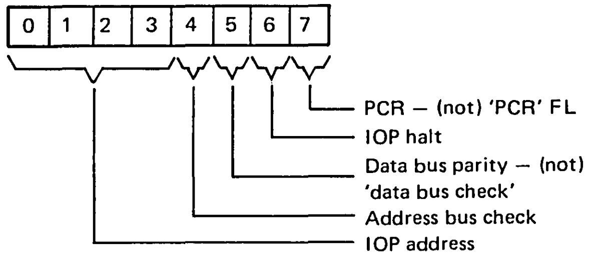
- Control strobe is inactive.

Two types of sense operations have to be distinguished:

SVP idle sense
SVP error sense.

The idle sense is used for system survey and is represented by an SVP microprogram routine called SVP main sense loop. Within this main sense loop each subprocessor and adapter is addressed periodically. The addressed subprocessor or adapter then gates its idle sense information on the SVP data bus. (In this way, the operating conditions of each subprocessor and adapter are checked periodically.)

The idle sense information is one-byte wide and contains the following:



If the idle sense does not contain any unusual condition no action is taken by the SVP and the SVP continues with its main sense loop.

If an address bus check occurs, propagation of the SVP address bus is prevented, and the idle sense bit pattern is set onto the SVP data bus. This points to the subprocessors and adapters between which the address bus error occurred.

SVP CTRL Operation

- Control strobe is active.

By use of SVP control operations the SVP is able to load certain bit patterns, via the SVP link, into the IOP circuitry. According to these bit patterns the IOP is operated under control of the SVP.

To perform a useful log, the K-register on the SVP link card may be loaded (by control 0) with bit patterns via the SVP data bus. These bit patterns appear on the K-bus, which activates and inactivates numerous control signals within the IOP circuitry. This permits different sense operations (according to the K-bus setting) which check out the IOP circuitry.

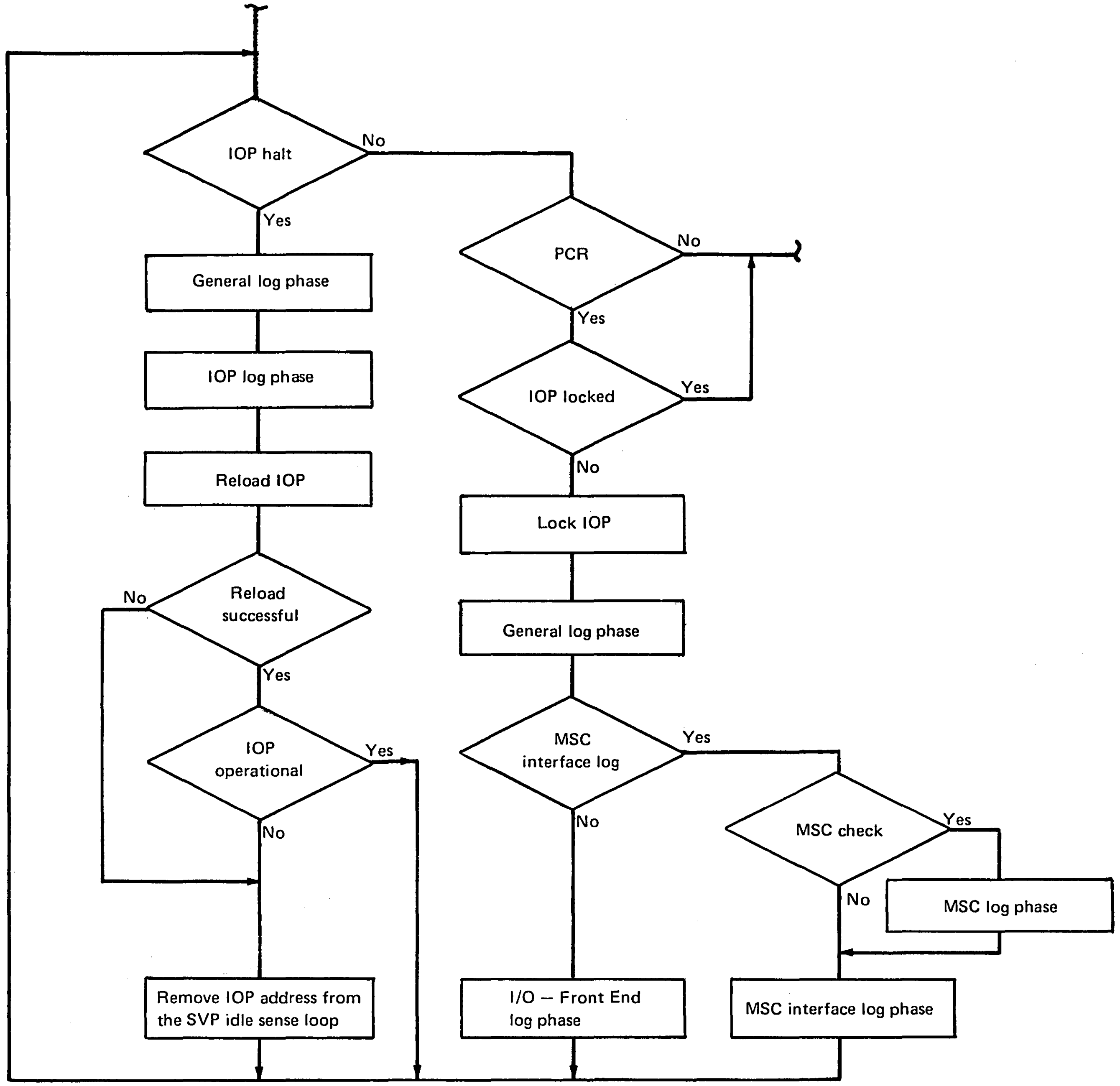
The K-bus setting also defines the mode, under which an IOP is operated. (See page 4-050.)

For further information, see the following pages:

4-010 and 4-020	Data Flow
3-930	SVP SENS Table
3-940	SVP CTRL Table.

SVP Main Sense Loop

- With IOP halt or with PCR bit in idle sense, the SVP takes the following action:



SVP SENS Table

- IOPs and IOP registers are addressed via SVP address bus 0.
- Information is transferred, via SVP data bus 0, from IOPs to the SVP.
- Control strobe is inactive.
- Further information is given under "Link to SVP" and "K-Bus" on Page 4-050

IOP Address	Register Address	Mode			SVP Data Bus 0								Remarks
SVP Addr Bits 0 to 3	Bus 0 Bits 0 to 7	K-Bus			Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	
		2	3	4									
	0		0		IOP Address 0	1	2	3	Addr Bus Check	(Not) Data Bus Check	IOP Halt	(Not) PCR FL	Idle sense
	1		0		(Not) Clock Check	(Not) Op-Reg Check	(Not) ZLS Check	(Not) Ext Addr Check	(Not) B-Reg Check	(Not) D-Reg Check	(Not) CSAR Check	(Not) X-Reg Check	Check sense
	2		0		SVP Requ FL	(Not) Br Cond	Scope Sense 1 2	3	4	5	6	Prevent I/O	Scope sense bit positions are connected to pins for test purposes
	3		0		Scope Sense 2 0	1	2	3	4	5	6	7	
	4	1	1		R-Bus Bits 3	4	5	6	7	Pty R2 to R7	Time Slice FL	Addr Equal Comparison	From ALSAR via ALS-out register to X-register
	5	1	1		Process Pointer Bits 0	1	2	Set DS Mode	Set SR Mode	Set MP	Time Slice FL	Addr Equal Comparison	From ALSAR via ALS-out register to X-register
	6	1	1		Link Bits 1	2	3	4	Set to 1	Link Bit 0	Time Slice FL	Addr Equal Comparison	From ALSAR via ALS-out register to X-register
	7	1	1		Access Pointer Bits 0	1	2	Access Mode M0 (DS)	M1 (SR)	MP	Time Slice FL	Addr Equal Comparison	From ALSAR via ALS-out register to X-register
	8	1		1	CR-Register Bits C2	C3	R2	R3	R4	R5	R6	R7	From CRY-switch
		1	1		ALS-D Bits 0	1	2	3	4	5	6	7	Addressed by R-bus bits 3 to 7 via ALS-out register to X-register
	9	1		1	Y-Register Bits Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7	From CRY-switch
		1	1		ALS-D Bits 0	1	2	3	4	5	6	7	Addressed by process pointer and mode via ALS-out register to X-register
	A	1		1	DS Cycle FL	Load Cycle FL	Invert	Control Storage Word Parity	C-Register Bits C4	C5	C6	C7	From CRY-switch
		1	1		Pointer Bits 0	1	2	Link Bits L0	L1	L2	L3	L4	Index word D via ALS-out register to X-register
	B	1		1	X-Register Bits 0	1	2	3	4	5	6	7	From CRY Switch
		1	1		ALS-D Bits 0	1	2	3	4	5	6	7	Addressed by access pointer and mode via ALS-out register to X-register
	C	1	1		ALS-B Bits 0	1	2	3	4	5	6	7	Addressed by R-bus bits 3 to 7 via ALS-out register to X-register
	D	1	1		ALS-B Bits 0	1	2	3	4	5	6	7	Addressed by process pointer and mode via ALS-out register to X-register
	E	1	1		Pointer Bits 0	1	2	Link Bits L0	L1	L2	L3	L4	Index word B via ALS-out register to X-register
	F	1	1		ALS-B Bits 0	1	2	3	4	5	6	7	Addressed by access pointer and mode via ALS-out register to X-register
			0		X-Register Bits 0	1	2	3	4	5	6	7	

SVP CTRL Table

- IOPs and IOP-registers are addressed via SVP address bus 0.
- Information is transferred, via SVP data bus 0, from the SVP to IOPs.
- Control strobe is active.
- Further information is given under "Link to SVP" and "K-Bus" on Page 4-050

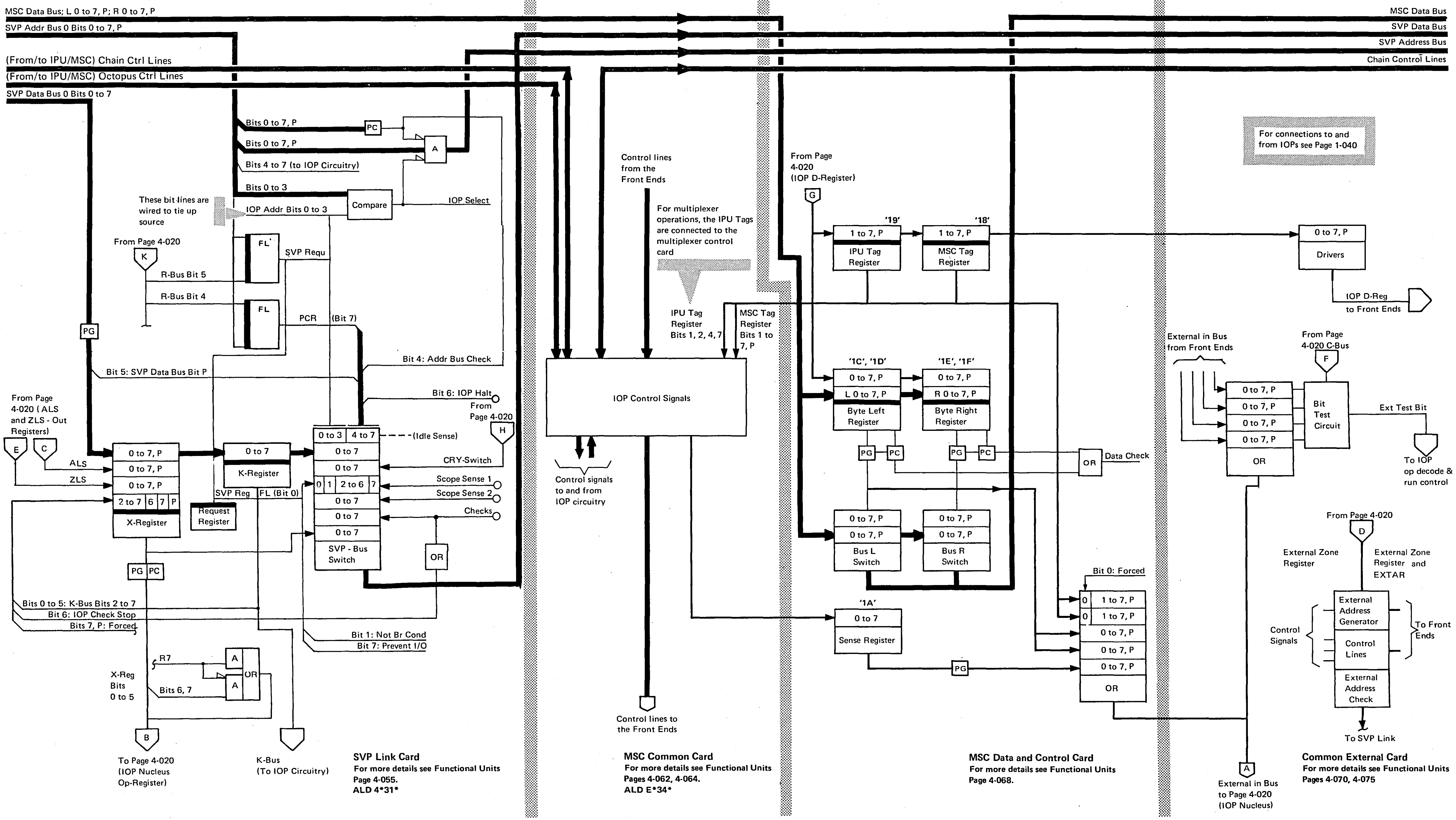
IOP Address	Register Address	Mode			SVP Data Bus 0								Remarks
SVP Addr Bits 0 to 3	Bus 0 Bits 4 to 7	K-Bus Bits			Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	
		2	3	4									
The IOP address is common for all registers and may be either '8', '9', 'A', or 'B'	0				Check Stop Override		Service Mode	Clock Reset	Stop at Cycle End	Address Equal Comparison	Clock Step		Set SVP data bus 0 into K-register
	1												Set SVP request FL
	2												Reset CE FL
	3	0			0	1	2	3	4	5	6	7	Reset PCR FL and set SVP data bus 0 into X-register
	4 & 5	1			0	1	2	3	4	5	6	7	Set ZLS out register contents into X-register
	6	1			K2	K3	K4	K5	K6	K7	IOP Check Stop	Set to 1	Set K-bus into X-register
	7	1			0	1	2	3	4	5	6	7	Set D-register (ALU) into X-register
	8	1		1	0 to R0 (later C2)	1 to R1 (later C3)	2 to R2	3 to R3	4 to R4	5 to R5	6 to R6	7 to R7	Set X-register contents into CR-register (op-register)
	9	1		1	0 to Y0	1 to Y1	2 to Y2	3 to Y3	4 to Y4	5 to Y5	6 to Y6	7 to Y7	Set X-register contents into Y-register (op-register)
	A	1		1	Not used		Invert	Control Storage Word Parity	4 to C4	5 to C5	6 to C6	7 to C7	Set X-register contents into C-register (op-register)
	B		0		0	1	2	3	4	5	6	7	Set SVP data bus into X-register (via ALS-D in switch ALS out register)
	C	0											IOP start (process access)
		1			Prevent control storage to Op-register	Control Storage Write Right	Control Storage Write Left	Control Storage Data In Buffer Process Area Select	Prevent modification to ALS-D	Retain Index Register	Prevent CSAR clock	D-register to Index Register	IOP start (service access) (See Note)
	D (ALU service control)	0											IOP start process
		1					2						C2, C3, R2 to R7 into CRY-switch
									0	0			CRY-switch to A-register; CRY-switch to B-register; A-register to D-register
									0	1	Not used		'X' FF to B-register; A-register crossover to D-register
									1	0			DLS to A-register; DLS to B-register, external in bus to D-register
							Not used		1	1			External in bus to A-register; external in bus to B-register; ALU to D-register
											0	0	A-register clock
											0	1	B-register clock
											1	0	A-register and B-register clock
											1	1	D-register clock
	E	Not used									ZLS to ZLS-Out Register	R-bus Bits 4 to 7 to External Zone Register	
	F	Not used							R-bus to CSAR-B D-register to CSAR-D	R-bus and D-register to Address comparison	Force 'Ext Wr' Pulse	Force DLS Wr Pulse	R-bus = R-bus Bits 2 to 7

Note:

Prevent control storage to op-register:	'Op-reg clk' is suppressed.
Control storage data in buffer process area select:	'Control storage data in bfr sel' is suppressed.
Prevent modification to ALS-D:	'ALS-D wr pulse' is suppressed.
Retain index register:	'T35 Index register clk' is suppressed.
D-register to index register:	1. Via ALS-D in switch and ALS-out register. 2. 'Control storage data in bfr wr pulse' is suppressed.

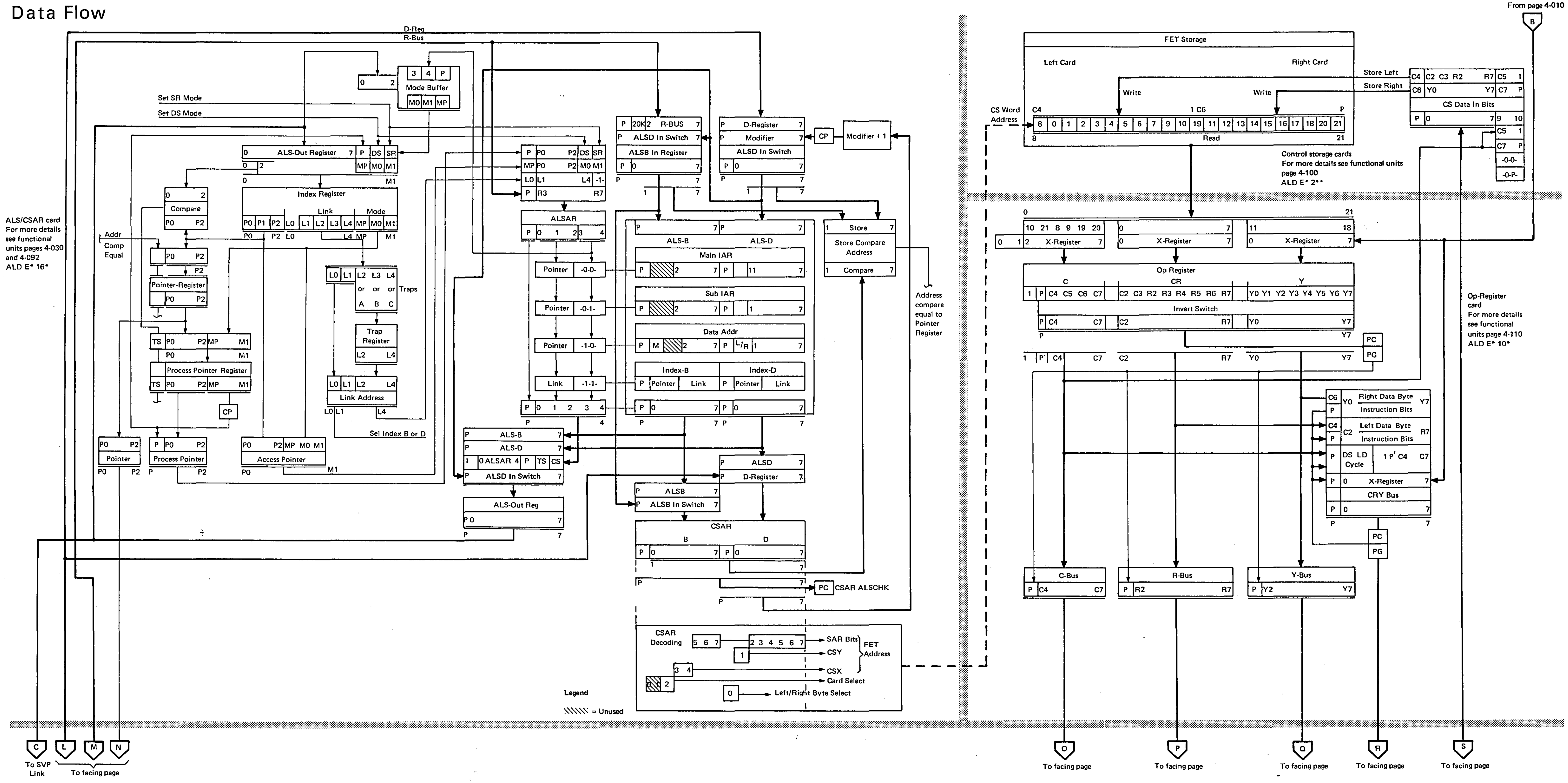
Chapter 4. Functional Units

IOP Data Flow (System Linkage)



This page is intentionally left blank

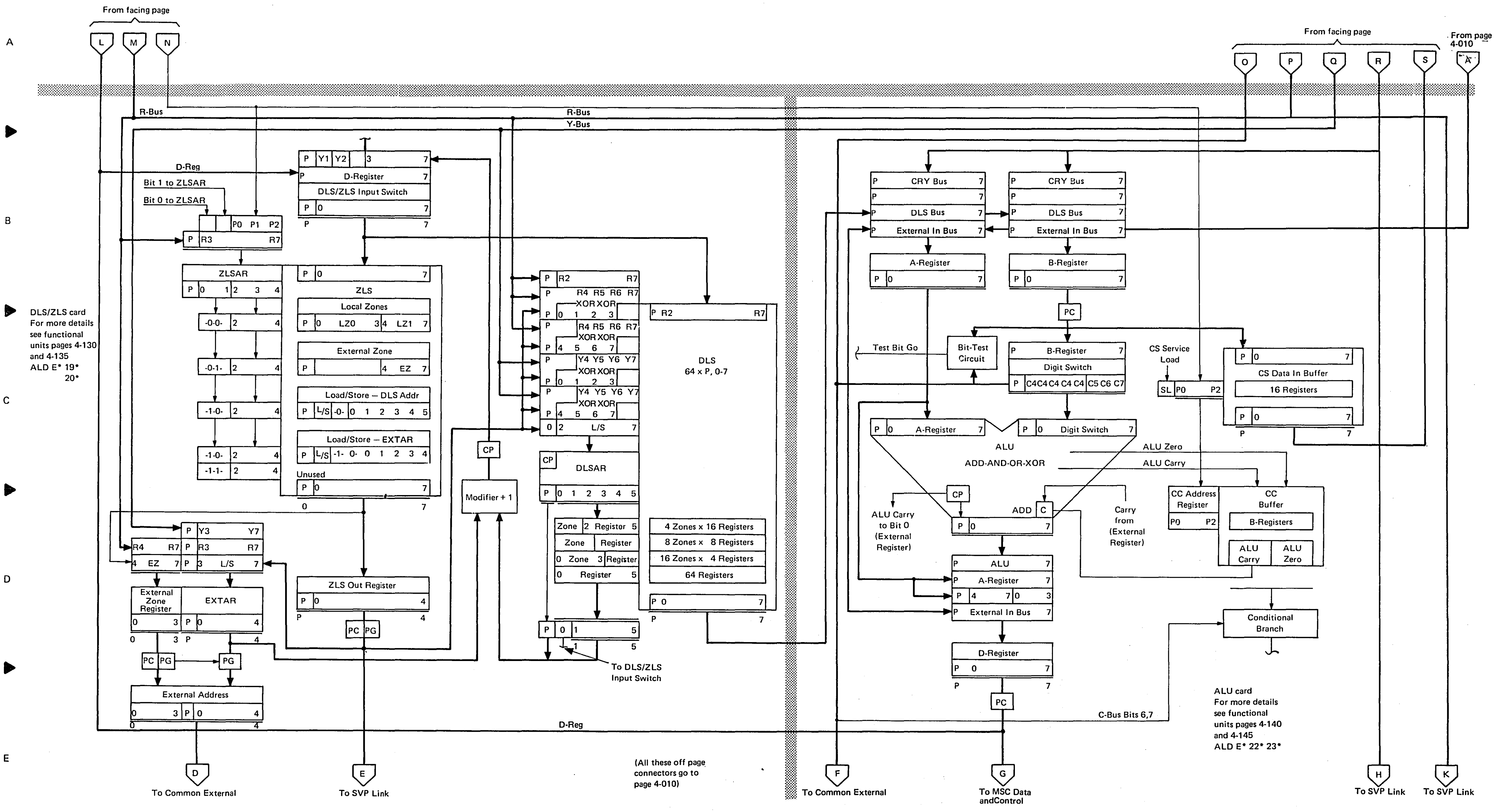
Data Flow



ALS/CSAR card
For more details see functional units pages 4-030 and 4-092
ALD E* 16*

Control storage cards
For more details see functional units page 4-100
ALD E* 2**

Op-Register card
For more details see functional units page 4-110
ALD E* 10*



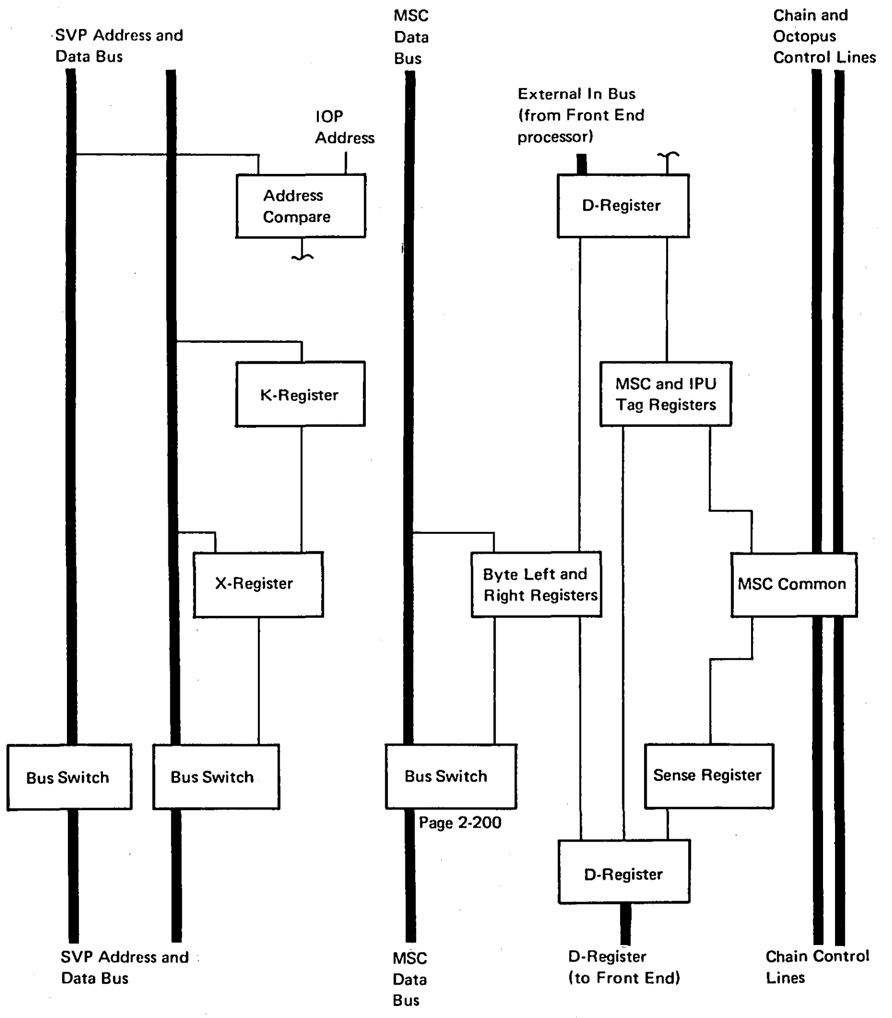
DLS/ZLS card
For more details
see functional
units pages 4-130
and 4-135
ALD E* 19*
20*

ALU card
For more details
see functional
units pages 4-140
and 4-145
ALD E* 22* 23*

Link to System

General

- IOPs are connected via "buses" to system components (subprocessors and Front Ends)
- These buses are connected to "external registers", which are addressed by encoding the external zone register bits and the EXTAR bits.
- A number of chain and octopus control lines determine the conditions and direction of any data transfer between IOPs and the MSC.
- For a definition of chain and octopus control lines refer to pages 2-120 and 2-130.



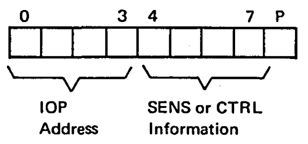
IMPL (Initial Microprogram Load)

- IMPL is performed under control of the SVP. Information is stored as bytes of data into addressed control storage locations.
- Each stored byte is fetched by the SVP and the received bit pattern is compared with the transmitted bit pattern.
  If the comparison is equal, the SVP continues with the IMPL.
- If the comparison is unequal, the SVP attempts to store the inversion of the failing byte. If a further comparison is equal, the SVP continues with the IMPL. If the comparison is again unequal, IMPL terminates and a message is given to the operator.
- Simultaneous with the store operations, the SVP generates a "hash total # 1".
- After IMPL is completed, the whole microprogram block is refetched and a "hash total # 2" is generated.
- The two hash totals are compared. If the comparison is equal, the IMPL is successful and the IOP is started. If the comparison is unequal, the SVP continues with the IMPL for the next subprocessor and a message is given to the operator.
- The started IOP continues with the initialization phase. The IOP is cleared and the chain of index words is generated. After completion of initialization, the address of the first index word is forced.
- The first index word is then set, via the index register, into the pointer and link registers, and the first instruction is read from control storage into the op-register. This instruction is an instruction of the basic loop, where the IOP idles and waits for the first selection or service request.
- For further details of SVP operations, refer to *IBM 3125 Processing Unit, Service Processor Subsystem,* Maintenance Library Manual, Order No. SY33-1065.

Link to SVP

- An IOP cannot start communication with the SVP by itself.
  Two possibilities exist to request SVP service:
  1. By microprogram. The 'PCR' latch is set, its output being gated to the SVP data bus switch.
  2. In the event of circuit errors, the IOP is stopped automatically, thereby activating the 'IOP halt' line, which is gated to the SVP data bus switch.
- The 'PCR' line and the 'IOP halt' line that are associated with the IOP address are sensed periodically by the main sense loop (idle sense) of the SVP. If one of these lines is active, the SVP branches to its log routine and fetches pertinent information from the requesting IOP.
- Link to SVP is established via an SVP data bus and an SVP address bus, both one-byte wide.
- SVP SENS operations do not influence the execution of the IOP microprogram.
- A number of special microprogram instructions also allow transfer of information between IOP and the SVP link.
- Detailed information is given on the pages listed below:
  Data flow, Page 4-010
  SENS and CTRL tables, Pages 3-930 and 3-940
  Microinstructions, Pages 3-030, 3-340, 3-345, 3-350, 3-355, 3-360
  Sense and control operations, Page 3-920

Address Bus



- With this address and no control strobe, the information specified by the SENS is set onto the SVP data bus.
- With this address and control strobe, either SVP data bus 0 or information specified by the CTRL is set into the X-register. The IOP then operates under control of the SVP, according to the bit pattern that has been set.

K-Bus

- The K-register bit combinations represent the mode in which each IOP operates under control of the SVP microprogram (see table below).
- Functional Units (Page 4-055) shows K-register and its circuitry.

K-Bus Bits (see Note 1)						Mode	
2	3	4	5	6	7		
0	0					Process mode	Mode Select (see Note 2)
1	0					Service control mode	
1	1					Service display mode	
1	0	0	0	0		Control service access cycle	Started by CTRL/C Instruction
1	0	0	1	0		Control service access cycle with stop on address comparison equal	
1	0	1	0	0		Single access cyc	
1	0	1	0	1		Single access cyc with clock step	
0	0	0	0	0		Cont process cyc	Started by CTRL/D Instruction
0	0	1	0	0		Single process cyc	
0	0	1	0	1		Single process cyc with clock step	

Notes

1. Bit 2 must not be activated before the IOP has been stopped through bit 4 being activated.
2. The K-bus can be regarded as the "mode switch" of former systems.

SVP Link Card

The SVP address bus is a byte-wide bus that is used to address IOPs and registers within the IOPs. The bus ends at the last subprocessor. Bus layout is shown on Page 4-050.

The SVP data bus is a byte-wide bus without parity. Data is transferred, via this bus, from the SVP to the subprocessors, and from the subprocessors to the SVP.

Control strobe is a control line (from SVP) used to distinguish between SENS and CTRL operations (see also Page 4-050).

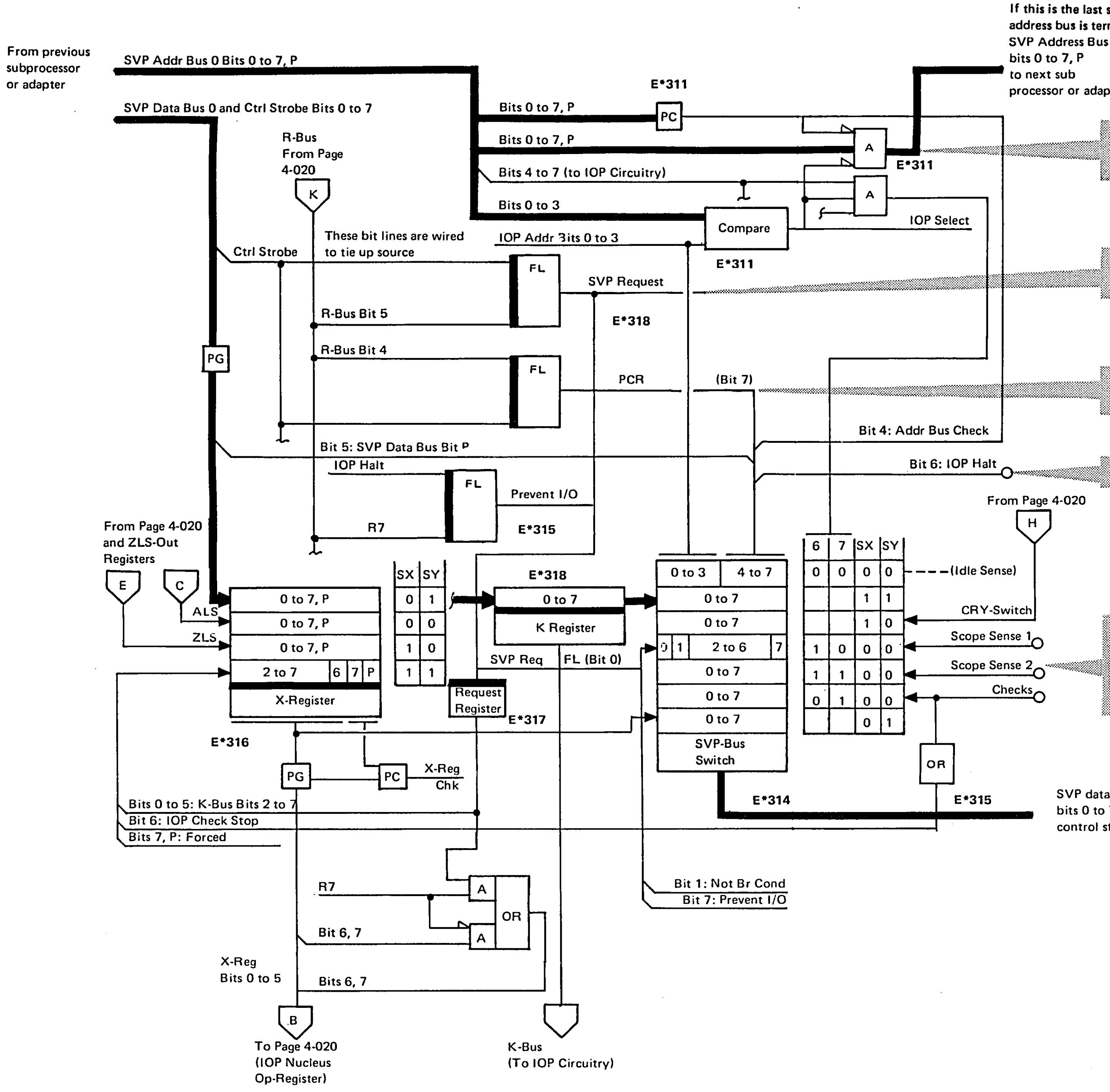
Control strobe in connection with SVP address bus is used.

To set SVP Request FL

To reset PCR FL

As clock signals for

K-register and X-register



If this is the last sub processor SVP address bus is terminated here

SVP Address Bus 0 bits 0 to 7, P to next sub processor or adapter

The SVP address bus and the SVP data bus are propagated from one IOP to the next IOP under two conditions:

(Not) 'IOP select' (see example A).

(Not) 'addr bus check' (see example B).

The signal 'SVP Requ FL' indicates that the SVP requires IOP service. The 'SVP Request' FL is set as a result of the bit pattern on the SVP address bus and is reset when the desired information is loaded into the X-register.

The signal 'Progr Cntrl Req FL' indicates that the IOP requires SVP service. This signal is activated by the microprogram after the desired information is loaded into the X-register.

The 'IOP halt' line indicates that the IOP requires SVP service. The line is activated whenever a circuit error occurs in an IOP

The bus lines 'Scope Sense 1' and 'Scope Sense 2' represent a number of available pins to which additional signals may be connected for fault finding purposes.

'Scope sense 1' = SENS X'2'/2-6.

'Scope sense 2' = SENS X'3'/0-7.

Example A: IOP Select

The 'IOP select' signal, when active, prevents the propagation of the SVP address bus to the next IOP. The 'IOP select' signal also prevents the 'Sel Bus SX' and 'Sel Bus SY' signals from becoming active. These two signals, inactive in conjunction with 'SVP Addr Bus Bits 4 to 7' being zero, select *idle sense*.

Example B: Address Bus Check

When an address bus error is detected, the 'Addr Bus Check' signal is generated which prevents the propagation of the SVP address bus to the next IOP. The 'Addr Bus Check' signal also prevents the gating of 'SVP Addr Bus Bits 5, 6, 7' and prevents the generation of the 'Sel Bus SX' and 'Sel Bus SY' signals.

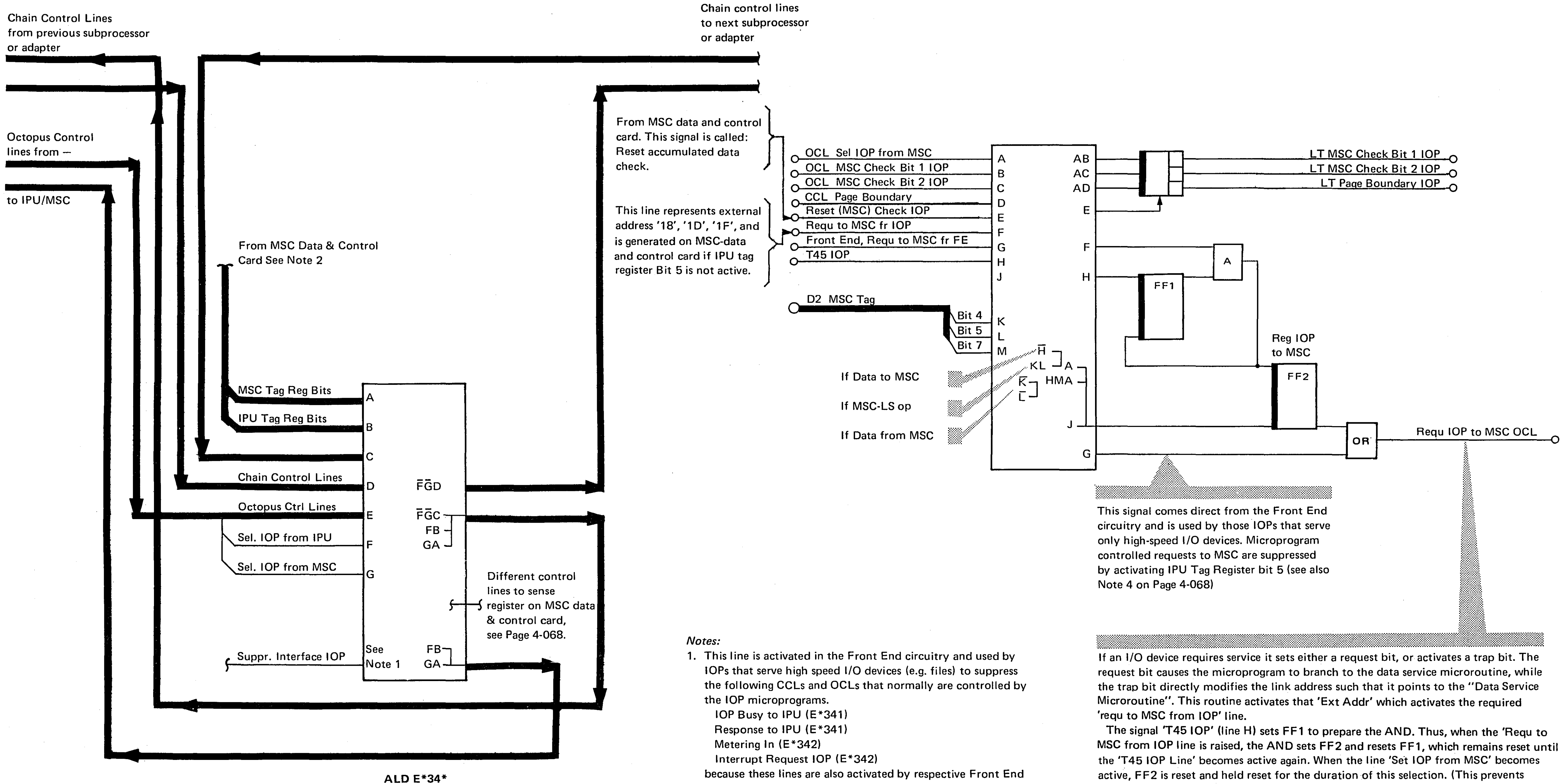
(Not) 'SVP Addr Bus Bits 6, 7' (Gated) and (Not) 'Sel Bus SX, SY' cause the IOP address, together with some checks, to be set on the SVP data bus.

Link to IPU/MSC

- Data transfer from and to the IOPs is controlled by a number of chain and octopus control lines.
- For outbound operations, information is transferred from the MSC to the IOP; for inbound operations, information is transferred from the IOP to the MSC.
- The control lines specify the transfer conditions. Furthermore, the IOP uses a number of control lines to indicate its status.
- Request lines are used by the IOP when the IOP needs service, that is, with its 'request' line active, the IOP is ready to start communication.
- The request lines are connected to priority networks; those for the MSC being connected within the MSC, and those for the IPU being connected within the IPU. These priority networks are scanned periodically and, if more than one request is active at the same time, requests are handled according to their priority.
- A check circuitry ensures that not more than one request can be accepted unnoticed.
- If a request is accepted, the MSC or the IPU responds by activating the 'select' line to the requesting IOP.
- However, the IPU can start communication with an IOP by activating the appropriate 'select' line. The selected IOP responds by activating its 'response' line.
- Any data transfer from or to the MSC is initiated by activating the 'request' line. If accepted, the transfer is executed, via a two-byte wide MSC data bus. (Halfword, two-byte, or single-byte transfers may take place.)
- Each byte on the MSC data bus has its own parity bit. Bus parity is checked and considered to be good for as long as an odd number of "up" levels exists.
- At least two registers of the MSC local storage (MSC-LS) are assigned to each IOP. Two other registers of the MSC-LS are used as common registers by all IOPs and by the IPU.
- The assigned registers are used for storing the main storage addresses that are required for data transfers.
- The common registers are used for interim storage of information such as IOP status, interruption code, condition code, etc.
- Before data transfer starts the IOP loads the start address into the MSC-LS. This start address is used by the MSC to address the main storage location into which data is to be stored or from which data is to be read.
- During the MSC access operation, the start address is modified either by ±1 or ±2 (as specified by the control lines 'increment', 'decrement', or 'halfword') and returned to the MSC-LS position from which the address was taken.
- The addressable MSC-LS registers are three-bytes wide. When MSC-LS is specified, the 'byte left' line has a different purpose: it specifies the left or right halfword of a local storage register. (The 'byte left' line normally specifies a transferred byte on the left- or right-hand side of the MSC data bus.)
- The 'page boundary' signal indicates that the next halfword or byte is transferred from, or to, a new page in main storage. If the next page is the adjacent one, this signal is of no importance. If the next page is not the adjacent one, the IOP microprogram must have the new page address available in the MSC-LS in sufficient time.
- IOPs with high data transmission rates have a second pair of registers (called *Area 2*) assigned in the MSC-LS. In these registers, a second data address can be stored in advance to save time. For further information see "Indirect Data Addressing" on Page 2-195.
- A 'request' line is inactivated while 'select' is active, but data transfer is not complete before 'select' is inactivated. For further information, see "Interaction Between IOP and MSC" on Pages 2-060 and 2-065.
- The decoding of MSC check bits 1 and 2 is shown on Page 2-130.
- Details of the MSC data and control card are given on the following pages:
  - Data flow, Page 4-010
  - Functional units, Page 4-068
  - Register arrangement, Page 2-070
- Details of the MSC common card are given on the following Pages:
  - Data flow, Page 4-010
  - Functional units, Pages 4-062 and 4-064
- Further information on IOP and MSC communication is given on the following pages. IOP-MSC-LS operations, Page 2-100 IOP-main storage operations, Page 2-110 IOP write-and IOP read-operations, Pages 2-030 to 2-055
- The MSC-LS layout is shown on Page 2-150
- For details of chain control lines, see Page 2-120
- For details of octopus control lines, see Page 2-130

This page is intentionally left blank

MSC Common Card



ALD E*34*

This signal comes direct from the Front End circuitry and is used by those IOPs that serve only high-speed I/O devices. Microprogram controlled requests to MSC are suppressed by activating IPU Tag Register bit 5 (see also Note 4 on Page 4-068)

Data Flow, see Page 4-010
Explanation of CCLs, see Page 2-120
Explanation of OCLs, see Page 2-130

Notes:

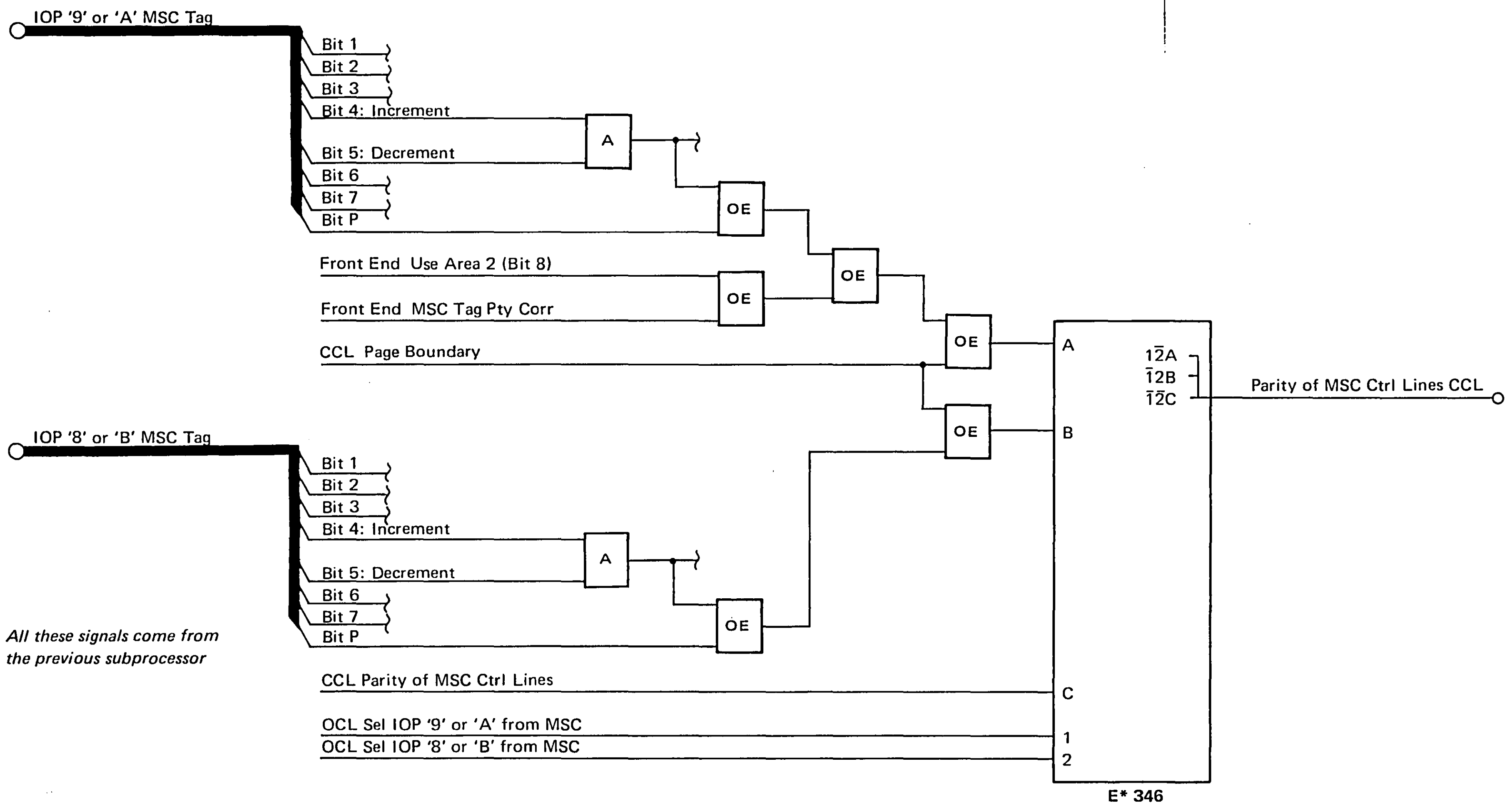
1. This line is activated in the Front End circuitry and used by IOPs that serve high speed I/O devices (e.g. files) to suppress the following CCLs and OCLs that normally are controlled by the IOP microprograms.
   IOP Busy to IPU (E*341)
   Response to IPU (E*341)
   Metering In (E*342)
   Interrupt Request IOP (E*342)
   because these lines are also activated by respective Front End circuitry.
2. An exception exists for the MPX channel. IPU tag register bits are connected to MPX control card and from there returned to MSC common card.

If an I/O device requires service it sets either a request bit, or activates a trap bit. The request bit causes the microprogram to branch to the data service microroutine, while the trap bit directly modifies the link address such that it points to the "Data Service Microroutine". This routine activates that 'Ext Addr' which activates the required 'requ to MSC from IOP' line.

The signal 'T45 IOP' (line H) sets FF1 to prepare the AND. Thus, when the 'Requ to MSC from IOP line is raised, the AND sets FF2 and resets FF1, which remains reset until the 'T45 IOP Line' becomes active again. When the line 'Set IOP from MSC' becomes active, FF2 is reset and held reset for the duration of this selection. (This prevents setting the same request twice.)

An exception occurs for data transfer operations from MSC to IOP: the reset of FF2 is also timed. This timed-reset condition causes a delay for the resetting of sense register bit 5 which is the bit that represents the branch condition for the beginning and end of the select signal. The delay is needed at the beginning of the selection to ensure correct data transfer.

Generation of 'Parity of MSC Ctrl Lines'



Note:
The MSC tag bits 1 to 7 are delivered with correct parity, but because bits 4 and 5 have a double function and are also used to generate a third signal ('Main Store Sel': OCL), new parity must be generated, as shown on this page.

For IOPs that serve high-speed I/O devices (such as disk files) two additional lines are used in the parity generation:
'Use area 2'
'MSC Tag Pty Corr'

The output from this parity generation circuit is 'Parity of MSC Ctrl Lines', which represents the parity of *all* MSC control lines. This parity is connected to the MSC, where it is used for checking purposes.

MSC Tag Bits

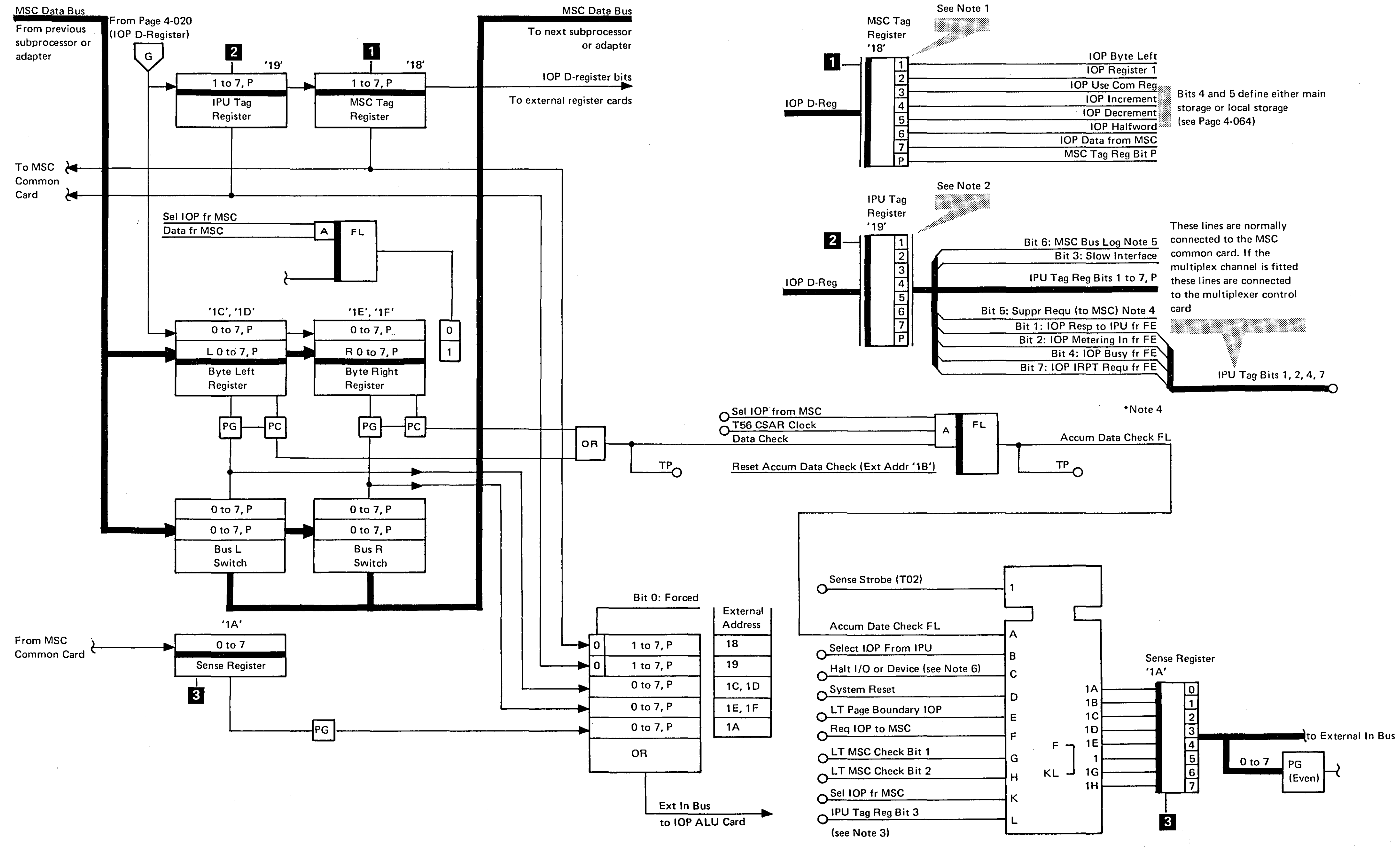
Bit 1 – Byte left
Bit 2 – Register 1
Bit 3 – Use common register
Bit 4 – Increment
Bit 5 – Decrement
Bit 6 – Halfword
Bit 7 – Data from MSC
Bit P – Parity

MSC tag register bits 4 and 5 are used to generate the line 'Main Store Sel'. The decode of these bits is shown below.

Bit 4	Bit 5	Meaning
Inactive	Active	Decrement, and select main storage
Active	Inactive	Increment, and select main storage
Inactive	Inactive	No modification, and select main storage
Active	Active	No modification, and select MSC-LS

MSC Data and Control

The MSC data bus is two bytes wide and is designated 'MSC data bus left' and 'MSC data bus right'. This bus is connected to each subprocessor and adapter of the system for data transfer to and from the MSC.



Notes:

1. Bit 0 of the MSC tag register is not used and is, therefore, always set to 1 by tie down.
2. Bit 0 of the IPU tag register is not used and is, therefore, always set to 0 by tie up.
3. *IPU tag register bit 3* can be set by microprogram, when necessary, irrespective of whether the start or the end of the 'select from MSC' signal is recognised.
   Bit 3 off: The start of 'select from MSC' controls bit 5 in the sense register (upper branch of OR condition).
   Bit 3 on: The end of 'select from MSC' controls bit 5 in the sense register (lower branch of OR condition).
4. *The IPU tag register bit 5* when on allows suppression of microprogram-controlled requests, but only on the IOPs whose Front Ends can generate (within their circuits) requests to the MSC (also see comments on Page 4-062).
5. *The IPU tag register bit 6* is used only by IOP #8 as a remember bit, when a MSC bus log is required after MSC check has occurred.
6. The 'Halt I/O or Device' line is required to ensure correct termination of a previously started I/O instruction. If an IOP is busy (that is, it is executing an I/O instruction), the IOP might not have time (because of a timeout in the IPU) to answer a 'Select IOP fr IPU' of a subsequent operation. This is undesirable for IOPs serving devices that have high data transmission rates, or when the multiplex channel is used.

   The 'select from IPU' is detected in the *basic loop* only and the microprogram of the busy IOP has to be forced to its basic loop by the active 'Halt I/O or Device' line. The IOP is thus able to detect the new selection in sufficient time to perform the following steps:
   - Fetch the new instruction identifier and analyze it.
   - Answer the 'Select IOP fr IPU' by activating the response signal in sufficient time.
   - Terminate the previously started instruction.

   This is achieved in the following way:
   - The active 'HIO/HDV' line is ANDed with select from IPU. This results in:
   - Bit 2 to be set into sense register on MSC data and control card.
   - Detection of activated bit 2 in this register by microprogram causes a branch to the basic loop.

Front End	ALD
MPX	KA 25x, 26x
2560	MG 13x, 14x
3525	MP 13x, 14x
3504	See note below
5425	MM 13x, 14x
1403	See note below

Note: ALDS for 3504 and 1403 have the prefix "MG", "MM", or "MP", according to the I/O configuration

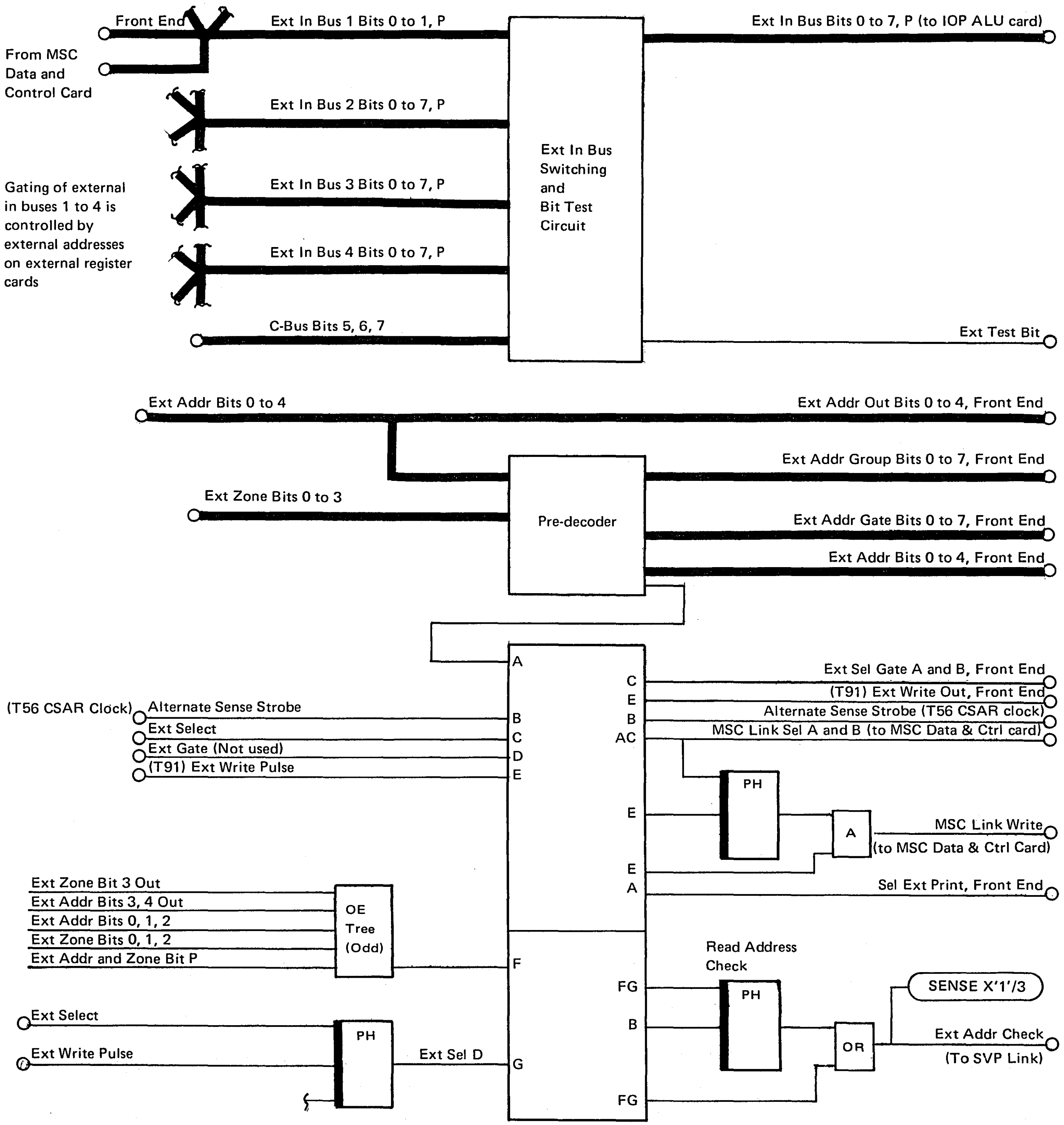
This page is intentionally left blank

Link to Front Ends (Common External Card)

- Four one-byte wide buses (designated ext in bus 1, 2, 3, and 4) are incoming lines from the four external register cards in the Front End.
- An external test bit is generated for use with "branch" instructions for branching on "bit on" or "bit off" conditions.
- Four other lines, which are connected direct to the IOP nucleus, are the 'trap bits A, B, C, and D'. These lines cause setting of bits into the trap register. The trap register content is ORed with the link portion of an index word to allow fast switching between different microprogram routines (trapping, see Page 3-550).
- Trapping is normally used when a device requires immediate service.
- Information to Front Ends is gated from the D-register via a one-byte wide bus.
- The external address group lines and the external address gate lines are generated from the contents of the external zone register and the EXTAR, and are also gated to the Front Ends.

  These lines are used in conjunction with the rest of the EXTAR bits to address external registers.
- The output of EXTAR and EXTAR zone register is checked; invalid or faulty addresses cause an 'external address check'.

The heavy dots represent Dot-OR functions, where the exits of the external register cards are dotted together. A maximum of four cards may be connected to one Dot-function. For 'Ext In Bus 1', one input is always the exit of the MSC Data and Control card.



Note: ALDs for 3504 and 1403 have the prefix "MG", "MM", or "MP", according to the I/O configuration

Front End	ALD
MPX	KA 34X, 35X
2560	MG 10X, 11X
3525	MP 10X, 11X
3504	See note
5425	MM 10X, 11X
1403	See note

The listing shows the lines between IOPs and Front Ends, an explanation, and description.

Connections Between IOPs and Front Ends

• *Connections from Front Ends*		
'Ext In Bus 1 to 4'	These lines represent four one-byte wide buses. Information is sent to the IOP, via these buses, from the external register cards and from the MSC data and control card.	
'MSC Tag Pty Corr'	This line is used for the generation of the line 'Parity of the MSC Ctrl Lines' (see Page 4-064)	These three lines are used with IOP 'A' and IOP '9' only (that is, the IOPs that serve high-speed I/O devices).
'IOP Use Area 2'	High-speed I/O devices may require a second area of registers in MSC-LS; this line defines this second area, see Page 2-130.	
'Suppr Interface'	This line prevents the propagation of control lines that are generated by the microprogram.	
'Ext Gate' 'Carry from Ext' 'Req to MSC fr FE'	These lines are not used.	
'Reset MSC Check IOP' (Reset accumulated data check)	This line represents external address '1B' and is used to reset latched MSC check bits 1 and 2 and latched page boundary signals on MSC common card (see Page 4-062) and to reset accummulated data check FL on MSC data and control card (see Page 4-068).	
• *Connections to Front Ends*		
'Ext Addr Out Bits 0 to 4' 'Ext Addr Group 0 to 7' 'Ext Addr Gate Bits 0 to 7' 'Ext Addr Bits 0 to 4 Out'	These four groups of lines are used to address the external registers in the different Front Ends. All of these lines represent the contents of the external zone register and the EXTAR. The two groups 'Ext Addr Group' and 'Ext Addr Gate' are generated by a decode of external zone registers and EXTAR bits.	
'IOP D-Reg Bits 0 to 7, P'	D-register contains ALU results and information that is to be transferred from the IOP to the Front End. The D-register output can be considered as the IOP bus out.	
'Sense Strobe'	This is the T02 pulse, which is used as the clock pulse for external registers.	
'Ext Write Out'	This is the T91 pulse, which is used as the clock pulse for external registers.	
'Ext Select Gate A, B'	This signal is used to generate the 'Card Select' signal in the Front End during "write external" or "load to external" operations.	
('T56 CSAR Clock Out') alternate sense strobe	This signal is used as a timing pulse in the Front Ends and is the set condition for the 'Pty Check' flip-latch.	
'Prevent I/O IOP'	In the event of an error this line is activated to prevent the gating of information in the Front End.	
'System Reset' 'Power on Reset' 'Lamp Test'	These lines are chain control lines that are activated and inactivated by the SVP. Detailed explanation is given on Page 2-120.	
'Metering Out' 'Clock Out'	These lines are chain control lines that are activated and inactivated by the IPU. Detailed explanation is given on Page 2-130.	
'ALU Carry to Bit Zero'	This line is not used.	
Sel Ext Print	This line represents an external address and is used by the printer front end circuitry to gate the external addresses.	

External Registers

Typical External Register Addressing and Arrangement for Model 125

- The table shows the way in which the external registers are arranged in connection with each individual IOP and the bits that are used for their addressing. The circuitry, used to generate these external address group and external address gate bits (pre-decoder) is indicated on page 4-070. The external addresses (hexadecimal) 18, 19, 14, 1B, 1C, 1D, 1E, and 1F are used to address:
  - MSC bus registers, (1C to 1F)
  - MSC tag register, (18)
  - IPU tag register, (19)
  - Sense register (1A)

These are all located on MSC data and control card. Address 1B is used as a reset signal.
(See also page 4-062 and 4-068.) The external register arrangement and addressing scheme is shown on page 4-096.

IOP Address (HEX)	Front End or Type of I/O Device	External Address Group Bits 0 to 7	External Address Gate Bits 0 to 7	External Address Out Bits 0 to 4	External Address Bits 0 to 4	Card in Front End on which Registers are Located	External Register Addresses (Hexadecimal)	Remarks
	IBM 2560	1	0		0, 1, 2	External register card	00 04 08 0C 10 14	
		1	1		0, 1, 2	External register card	01 05 09 0D 11 15	
		1	2		0, 1, 2	External register card	02 06 0A 0E 12 16	
		1	3		0, 1, 2	External register card	03 07 0B 0F 13 17	
8	IBM 3525	1	0		0, 1, 2	External register card	00 04 08 0C 10 14	
		1	1		0, 1, 2	External register card	01 05 09 0D 11 15	
		1	2		0, 1; 2	Auxiliary register card {	06 0E 16	
		1	3		0, 1, 2		03 07 0B 0F 13 17	
	IBM 5425	1	0		0, 1, 2		00 04 08 0C 10 14	
		1	1		0, 1, 2		01 05 09 0D 11 15	
		1	2		0, 1, 2		02 06 0A 0E 12 16	
		1	3		0, 1, 2	Print control card	03 07	These two registers are normally used as sens registers. If diagnostic mode, they are used as ctrl registers
	IBM 1403				1, 2, 3, 4,			IOP Addresses '30' – '3E' (sens) IOP Write Addresses '30' – '3C' (ctrl)
	IBM 3504	4	0		0, 1, 2	External register card	00 04 08 0C 10 14	
		4	1		0, 1, 2	External register card	01 05 09 0D 11 15	
9	MPX	0	0	0, 1, 2		External register card	00 04 08 0C 10 14	
		0	1		0, 1, 2	Multiplexer control card	01 0D	
		0	2, 3		0, 1, 2	UCW Buffer cards	(see remarks column)	Combinations of these bits allows the addressing of one out of the twelve bytes addressed by the contents of external register 0C
A	DDA		0		0, 1, 2, 3, 4		00 04 08 0C 10 14	
		0	1		0, 1, 2, 3, 4		01 05 09 0D 11 15	
		0	2		0, 1, 2, 3, 4		06 0A 0E 12 16	
		0	3		0, 1, 2, 3, 4		03 0B 0F 17	
		0						

This page is intentionally left blank

ALS/CSAR

ALS (AddressLocal Storage)

* Address local storage ia an 18-bit wide high-density array.
- ALS contains 32 halfwords which are addressed by pointer, link, or R-field.
- When addressed by *access pointer* the address of the next instruction to be executed is read out.
- When addressed by *process pointer* the address of the data to be processed is read out.
- When addressed by *link* the next index word is read out.
- When addressed by *R-field* either the Y-field of the instruction or the contents of the register that is specified by the Y-field is set into ALS (via micro-instructions SARI, SABR, SADI, SADR).
- The contents of ALS are set into either CSAR or into the ALS out register.
- If the contents are set into CSAR, the ALS contents are used to address control storage.
- If the contents are set into the ALS out register, the ALS contents are used to address mode buffer. ALS contents and mode buffer contents are used to generate the following:

Access pointer represented by index register bits 0, 1, 2 and access mode register bits 0, 1, P.

Process pointer represented by

1. Pointer register bits 0, 1, 2, which are used during the process cycle to address ZLS, ALU CC buffer and control storage data in buffer.
2. Process pointer register contents, which are used to address ALS during the process cycle and to set up the necessary controls (according to the mode buffer contents) for that process cycle.

- Index register bits 0, 1, 2 and ALS out register bits 0, 1, 2 are compared; if the two values are unequal, time slice mode is activated.

Note: The index register contains the current pointer; the ALS out register contains the next pointer. Refer to Pages 3-510, 3-520.

- Mode buffer contains status information for each routine.
- Mode buffer is addressed by ALS out register bits 0, 1, 2 (which represent the pointer) so that the status information is available early in the access cycle for each instruction.
- The status information controls the selection of the ALS contents appropriate for mainroutine, subroutine, data storage cycle, or the next index word. The information, pertinent to one program (or routine), such as MIAR, SIAR, and data address, is stored in adjacent positions of ALS. This allows the addressing of these storage positions by altering only the mode part of the ALS-address.
- Status information is updated depending upon the type of the executed cycle and is restored depending upon the condition of both time slicing and 'Mode Bfr Write Gate'.
- For details of the ALS circuits, see Page 4-092.

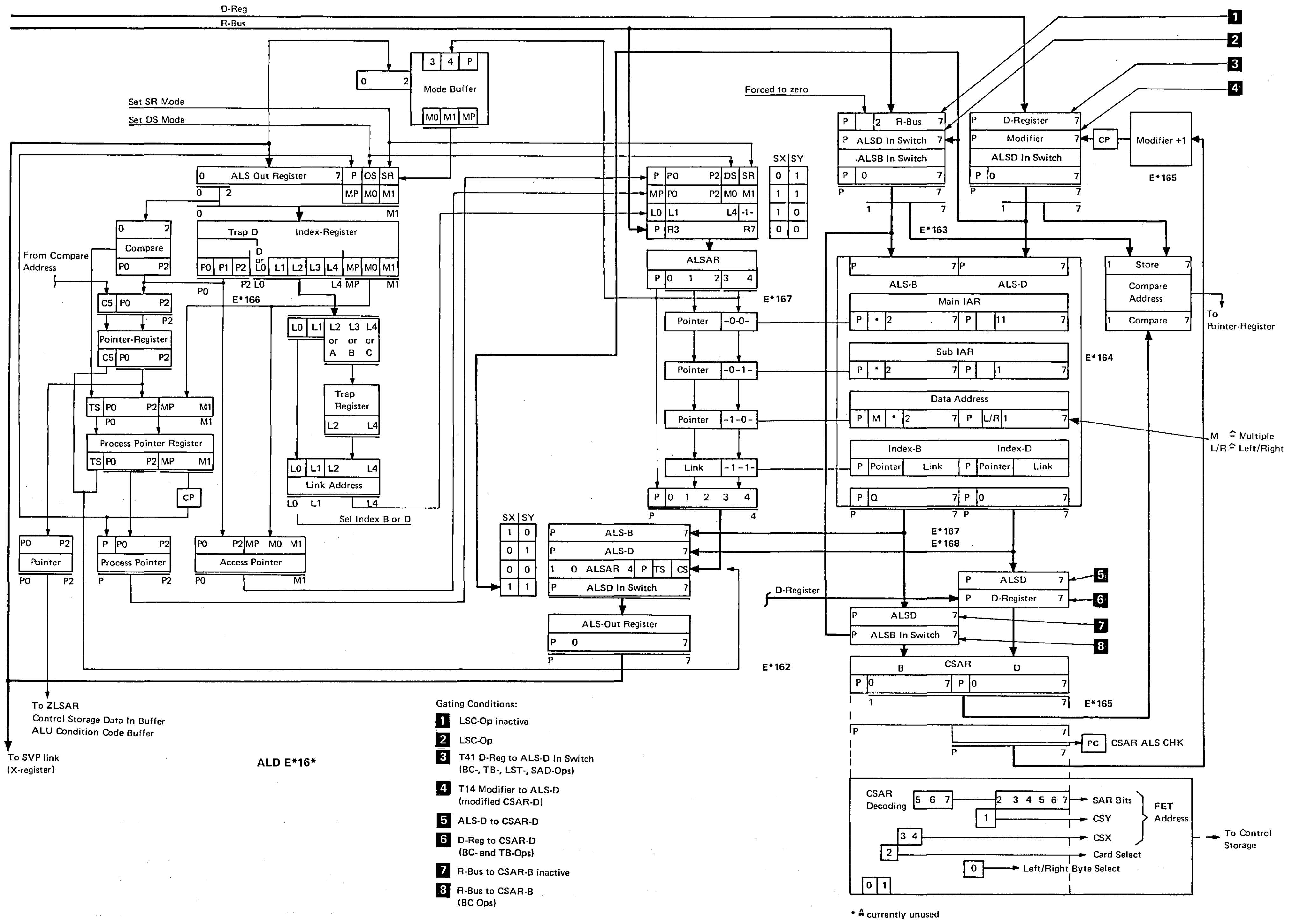
CSAR (Control Storage Address Register)

- Control storage address registers CSAR-B and CSAR-D, are both one-byte wide.
- CSAR-B represents the block portion and CSAR-D represents the displacement portion of a control storage address.
- Control storage can be addressed:
  Directly by using the ALS content.
  Indirectly by using the ALS-B portion together with the contents of a work register that is specified by the instruction.
- CSAR-D is gated through a modifier before being restored in ALS-D.
- CSAR-B and CSAR-D are compared with the ALS-B and ALS-D in switches and an 'Address Compare' signal is generated. This signal can be used either to stop an IOP at any desired storage address or for synchronization purposes.
- Normally ALS-B and ALS-D are set into CSAR-B and CSAR-D respectively. However, with branch operations the R-bus is set into CSAR-B; with branch or "Load/Store" ops D-register is set into CSAR-D (see Chapter 3).
- For details of the CSAR circuits, see Page 4-092.

CSAR Decoding

Function	CSAR-B								CSAR-D							
	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
SAR bits						X	X	X			X	X	X	X	X	X
CSY										X						
CSX				X	X											
Card select			X													
Left and right byte									X							
Not used for control storage addressing	X	X														

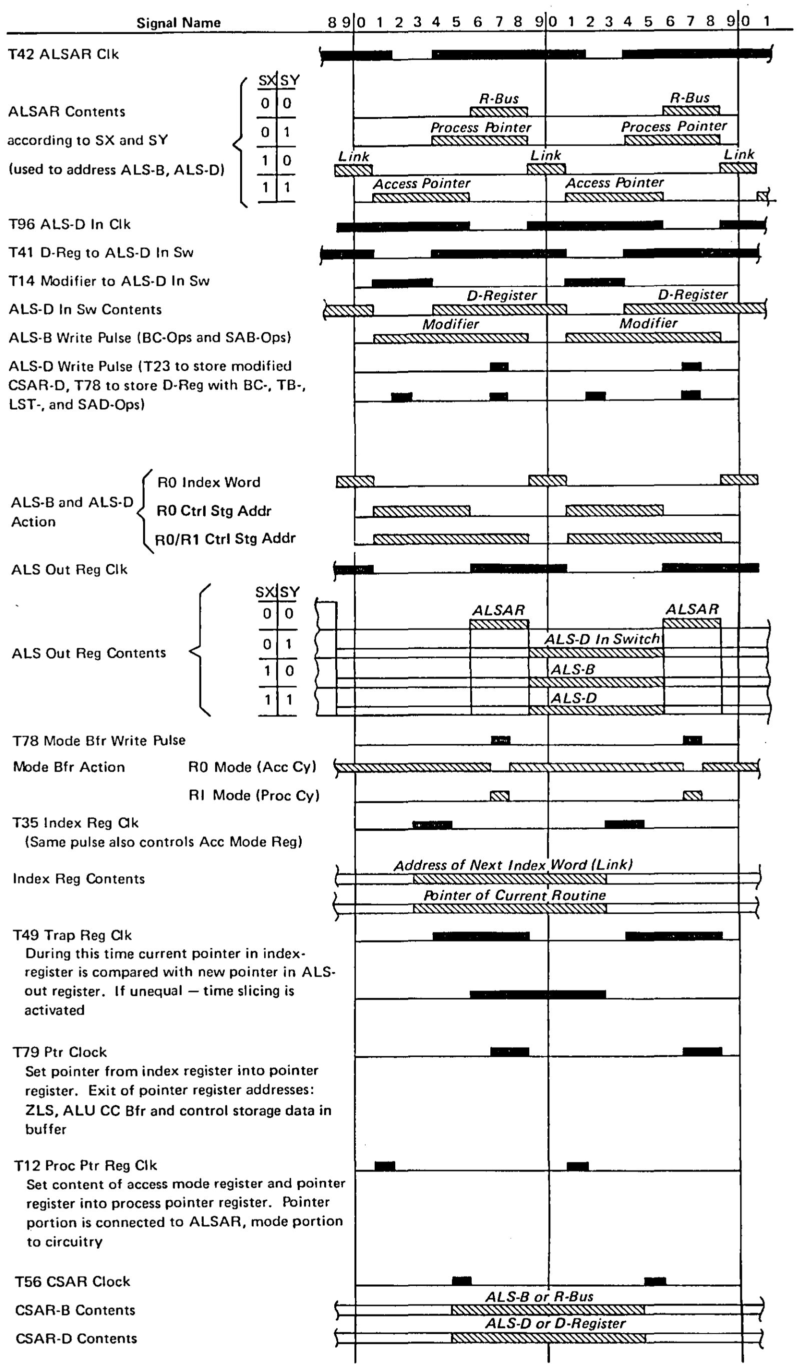
Data and Control Flow, and Timing



Gating Conditions:

1. LSC-Op inactive
2. LSC-Op
3. T41 D-Reg to ALS-D In Switch (BC-, TB-, LST-, SAD-Ops)
4. T14 Modifier to ALS-D (modified CSAR-D)
5. ALS-D to CSAR-D
6. D-Reg to CSAR-D (BC- and TB-Ops)
7. R-Bus to CSAR-B inactive
8. R-Bus to CSAR-B (BC Ops)

* ≙ currently unused

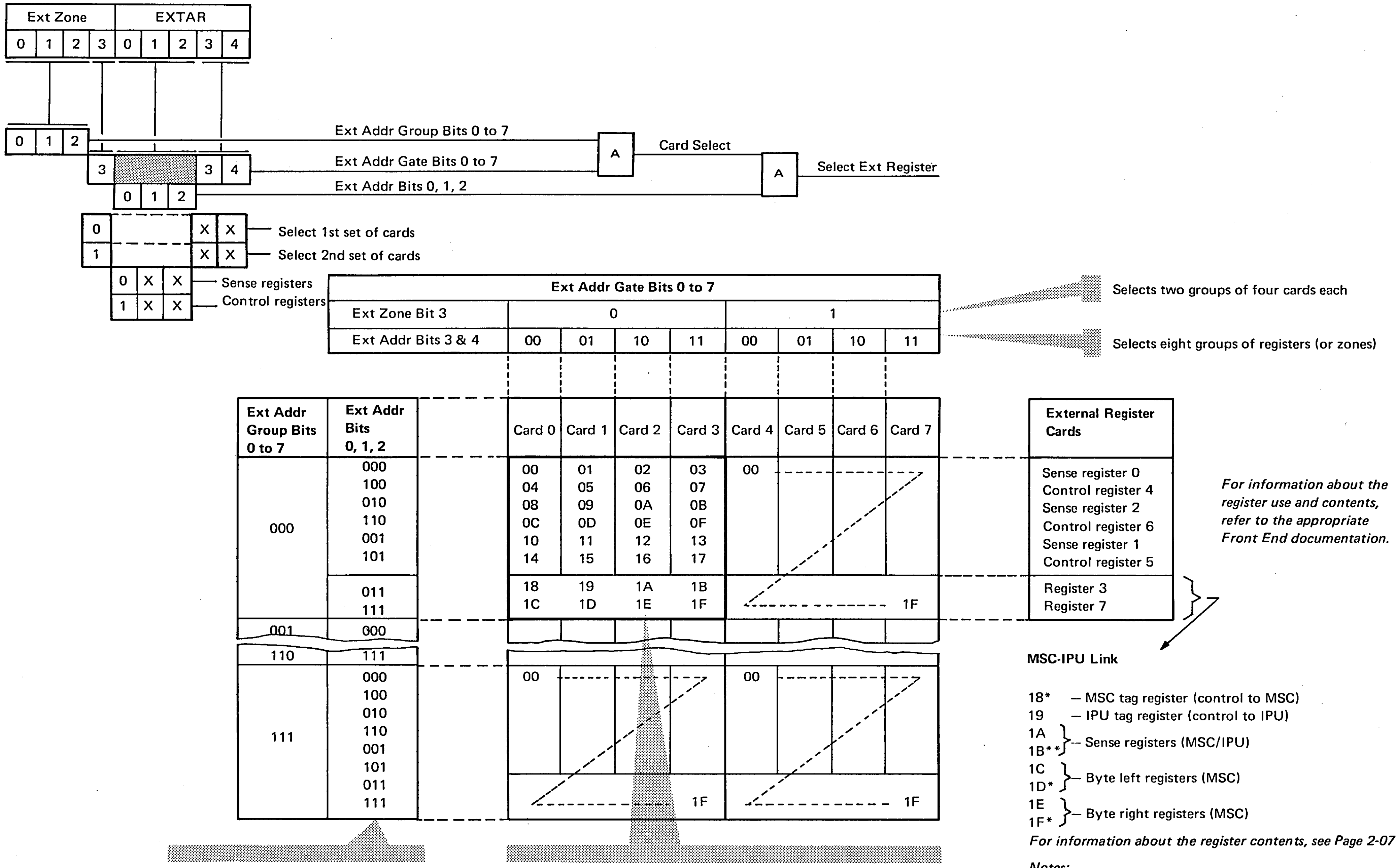


External Register Arrangement and Addressing Scheme

(See also pages 4-070, 4-080)

Coding of external zone bits 0, 1, 2 to external address group bits 0 to 7, as well as coding of external zone bit 3 and and EXTAR bits 3 and 4 to external address gate bits 0 to 7 is carried out on common external card.

Coding of EXTAR bits 0, 1, 2 to distinct external register addresses is carried out individually on those cards, which carry external registers.



Ext Addr Gate Bits 0 to 7								
Ext Zone Bit 3	0				1			
Ext Addr Bits 3 & 4	00	01	10	11	00	01	10	11

Ext Addr Group Bits 0 to 7	Ext Addr Bits 0, 1, 2	Card 0	Card 1	Card 2	Card 3	Card 4	Card 5	Card 6	Card 7	External Register Cards
000	000	00	01	02	03	00				Sense register 0
	100	04	05	06	07					Control register 4
	010	08	09	0A	0B					Sense register 2
	110	0C	0D	0E	0F					Control register 6
	001	10	11	12	13					Sense register 1
	101	14	15	16	17					Control register 5
	011	18	19	1A	1B					Register 3
	111	1C	1D	1E	1F				1F	Register 7
001	000									
110	111									
111	000	00				00				
	100									
	010									
	110									
	001									
	101									
	011									
	111				1F				1F	

For information about the register use and contents, refer to the appropriate Front End documentation.

The 'Ext Addr Bits 0 to 7' select a register within the group (or zone).

This area enclosed within the heavy-lined area represents one out of 16 register groups.

Registers X'18' to X'1F' are located on one card (the MSC data and control card).

The number of registers used can vary from one Front End to another according to the requirements of the attached device(s).

Registers are not necessarily arranged as shown; deviations are possible.

MSC-IPU Link

18* — MSC tag register (control to MSC)
19 — IPU tag register (control to IPU)
1A, 1B** — Sense registers (MSC/IPU)
1C, 1D* — Byte left registers (MSC)
1E, 1F* — Byte right registers (MSC)

For information about the register contents, see Page 2-070.

Notes:

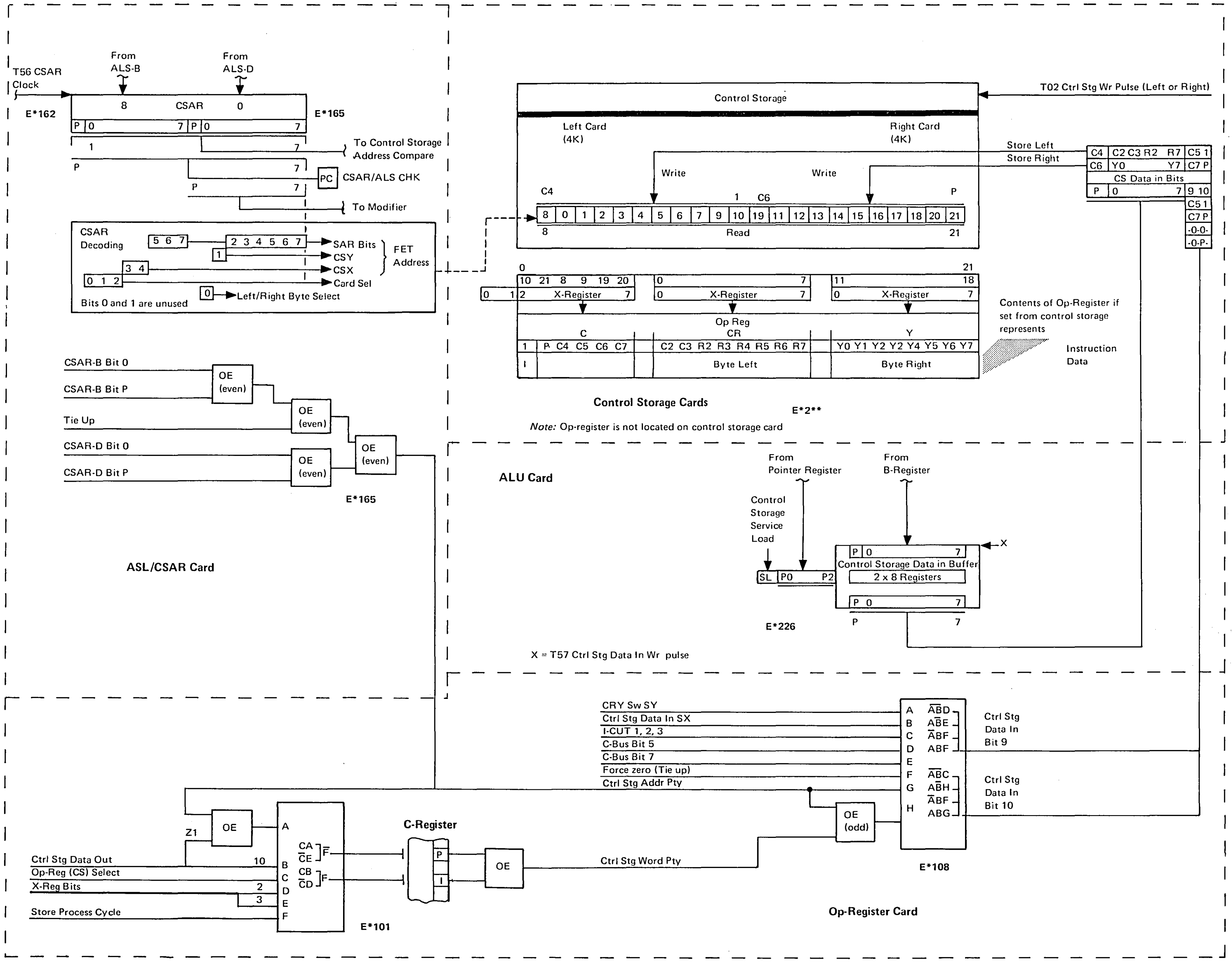
- * These addresses automatically activate the 'Req to MSC from IOP' line. Two addresses are assigned to the byte-left and byte-right registers. This allows loading of these registers either with 'Request to MSC' active or not active.
- ** This address causes reset of the "MSC data checks" by activating the 'Reset Accum Data Check' line

Control Storage

- Control storage size is in the range 1K bytes to 8K bytes; the actual size depends on requirements.
- Control storage contains microprogram and is also used as data storage.
- If control storage is used as data storage, data is fetched and stored by the following:
  Load byte instruction
  (loads control storage data into a specified register).
  Store byte instruction
  (stores data from a specified register into control storage).
- The required number of cycles for data fetch and store operations is equal to the number of bytes to be handled plus one.
- Data that has to be stored into control storage is first set into the control storage data in buffer.
- The "from" register content of the "store byte" instruction is set into the control storage data in buffer, which is addressed by pointer bits 0, 1, 2 of the currently executed routine.
- The control storage data in buffer consists of two sets of eight one-byte-wide registers. Eight registers are used for the process area, and the other eight registers are used for the service area.
- The 8-byte process area is inactivated during service phases to prevent loss of data.
- Because each addressable control storage position is 11 bits wide, the nine control storage data in buffer bits have to be extended to 11 by use of either C-bus bits 5 and 7 or control storage word and address parity.

Treatment of Invalid Addresses

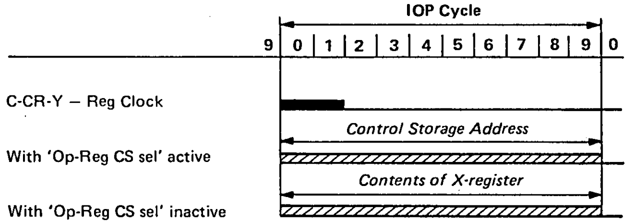
- Any attempt to address control storage outside its address range *does not* cause an invalid address indication.
- If the address used points to the area between actual storage size and the maximum storage size of 8K, an all ones bit pattern with invert bit is delivered. This results in a branch to control storage location 00.
- If the address used is above the 8K limit, the microprogram continues with the used address minus 8K.
- If during IMPL an address outside the actual storage size is used, the *stored byte* and the *re-fetched byte* do not match. This results in a message to the operator and IMPL is terminated.
- For details of control storage operations (LST-Ops) see pages 3-210 to 3-255 (Group 2 microinstructions).



Op-Register

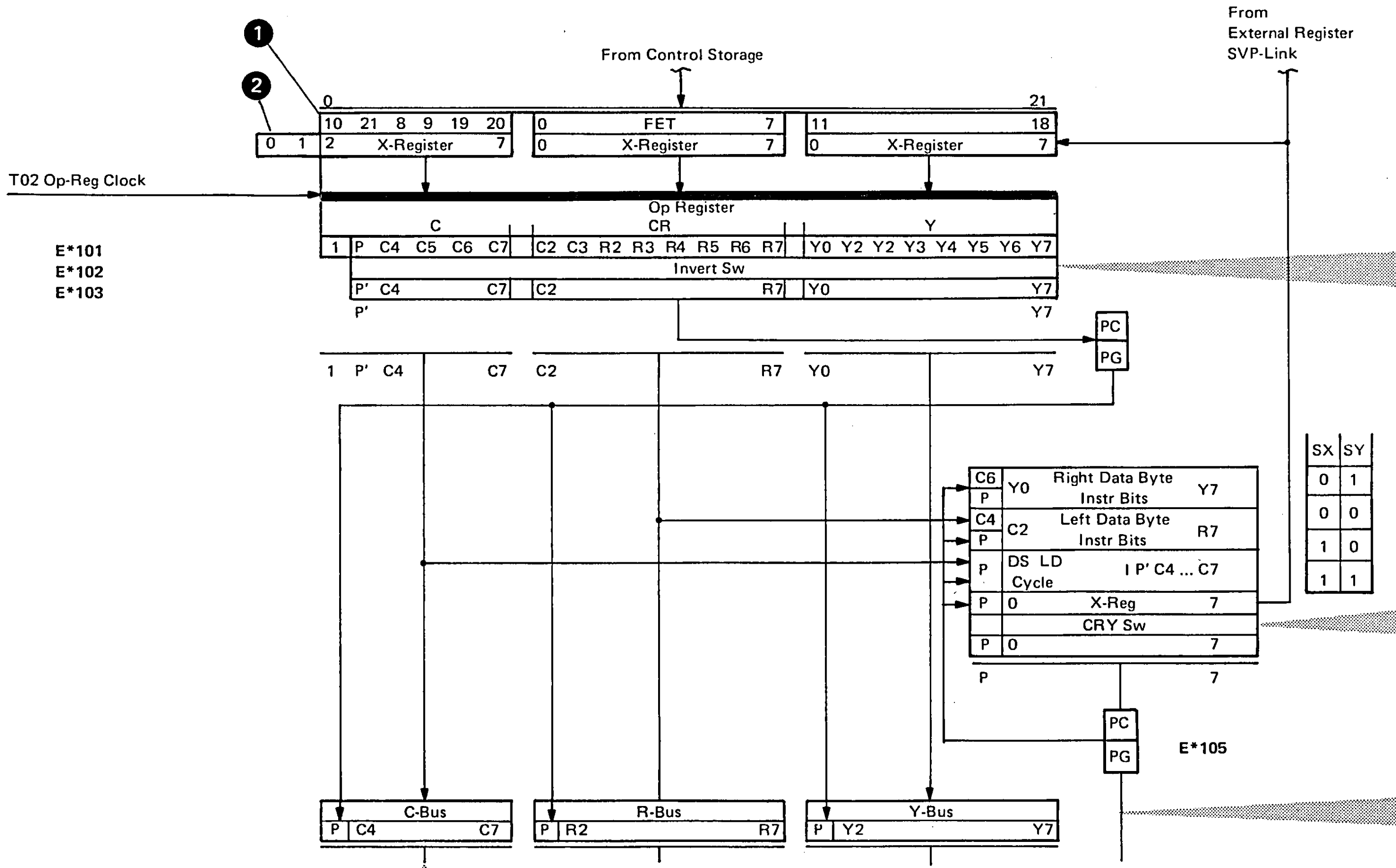
- The op-register consists of three registers:
  - C-register (5-bits)
  - CR-register (8-bits)
  - Y-register (8-bits).
- These three registers each have their own output:
  - C-bus (6 bits; C-register bits 4 to 7 and R-register bits 0 and 1)
  - R-bus (6 bits; R-register bits 2 to 7)
  - Y-bus (8 bits; Y-register bits 0 to 7).
- Parity is separately generated for each bus.
- Either a 22-bit control storage word or the contents of the X-register (SVP link) is read into the op-register according to the condition of the 'Op-Reg CS Sel' line.
- If data is read from control storage,
  - Y-register contains the right-hand byte of the halfword that is stored in the addressed control storage word.
  - CR-register contains the left-hand byte of the halfword that is stored in the addressed control storage word.
- For simplified error correction the op-register contents are inverted at the op-register output according to either
  - Invert bit (position 10) or
  - X-register bit 2 being active.

Op-Register Contents



❶ With the signal 'Op-Reg (CS) select' active, contents of addressed ctrl stg position is set into Op-register.

❷ With the signal 'Op-Reg (CS) select' inactive contents of X-Reg (located on SVP link card) is set into Op-register.



Invent Switch

- The invert bit is used for simplified error correction. It is set "on" when the inverse of a bit pattern is stored by the SVP during IMPL.
- The operation of the invert bit is as follows:
  Assume that the contents of an addressable storage word is .... 0111. If position 1° of this storage word is defective, the bit pattern read out of that strage word will be .... 0110.

  It is considered that this particular error condition will always generate the same bit pattern. Therefore, the bit pattern .... 0111 is stored inverted (that is, ... 1000) and the invert bit is set "on"

  During subsequent read operations, the bit pattern that is read from this storage word is .... 1000. However, the invert bit activates the invert switches, which invert the bit pattern to .... 0111 (that is, to the correct and desired bit pattern).

CRY Switch

- The CRY switch connects the C-, R-, or Y-bus together with additional information) via a byte-wide internal CRY-bus to the A and B input registers of the ALU.

CRY Bus

The CRY-Bus holds selected contents of op-register as shown above. CRY-Bus is connected to
- SVP link card
- ALU card

C-Bus

C-bus normally represents the op-code. A number of control signals are generated from the C-bus bits

Y-Bus

Y-bus represents either a "from" register address or immediate data. If it represents an address it is used to address either DLS or external registers. If it represents immediate data, this data is gated into the ALU, where it is used either to test or to alter bit patterns

R-Bus

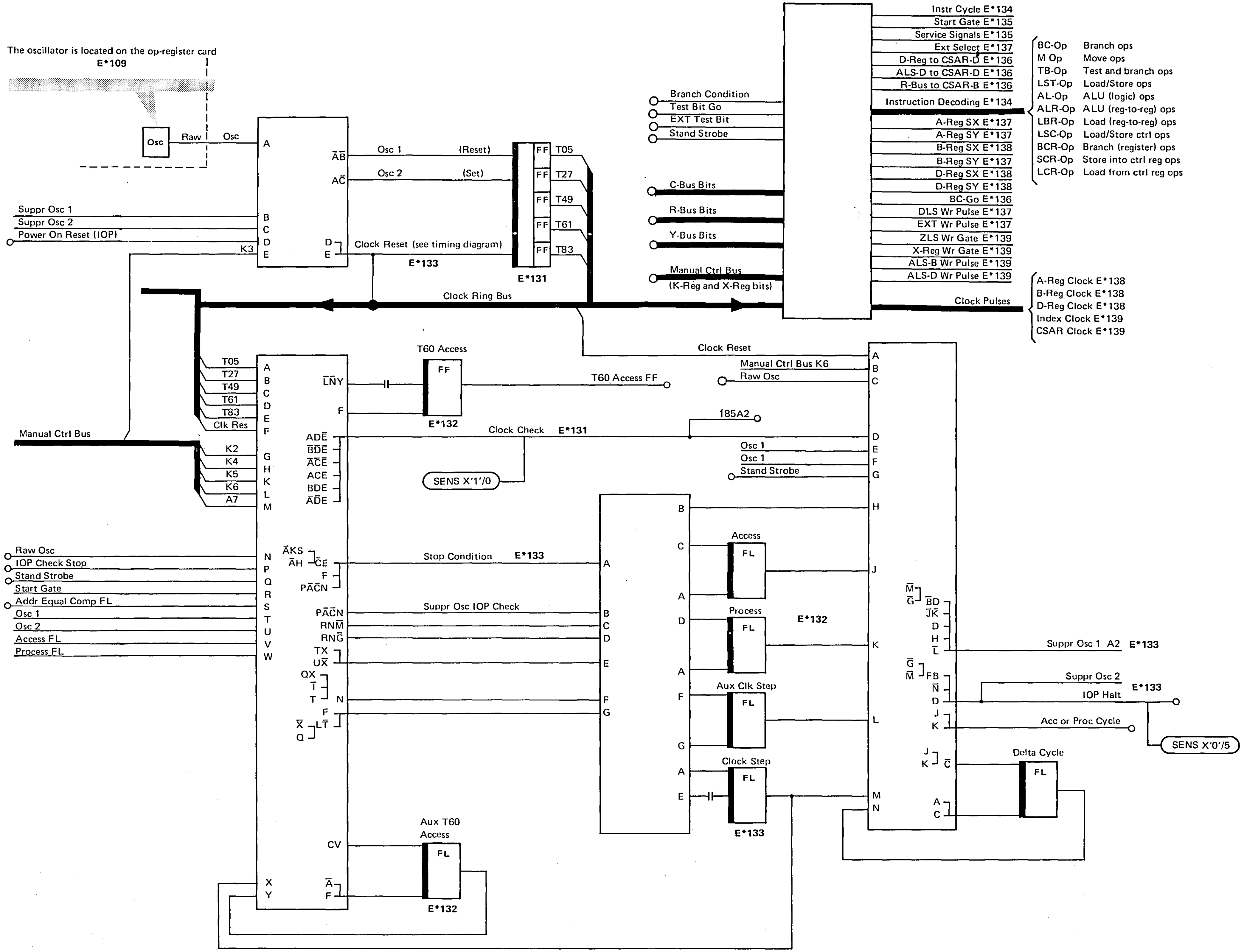
R-bus normally represents a "to" register address. R-bus is gated according to the op-code into ALS (if the address is to be stored), or into DLSAR, or into EXTAR (if used to address DLS or external registers)

C-Bus
R-Bus are connected to:
Y-Bus

- Op decode and run control card (analysis of Op-code)
- ALU card (definition of ALU function and increment/decrement amount into digit switch)
- ALS/CSAR card (with SAB and SAD instructions)
- DLS/ZLS card (definition of local or external zone and with SZ instructions)

Op-Decode and Run Control

- The microinstructions are read from control storage into the op-register. Decoding of the microinstructions (op-register output) takes place in AND/OR circuits. The outputs of which represent the required control signals.
- The IOP clock controls the timing of the control signals.
- The run control supervises the normal microinstruction sequence. The IOP starts and stops under the following conditions:
  (a) Under control of the SVP (for manual operations).
  (b) When the 'Clock Step' FF is set. (This condition allows stepping of clock ring FF positions.)
  The IOP stops under the following conditions:
  (a) On internal errors.
  (b) With 'Address Compare' signal active. (This stops the IOP on a successful address comparison.)



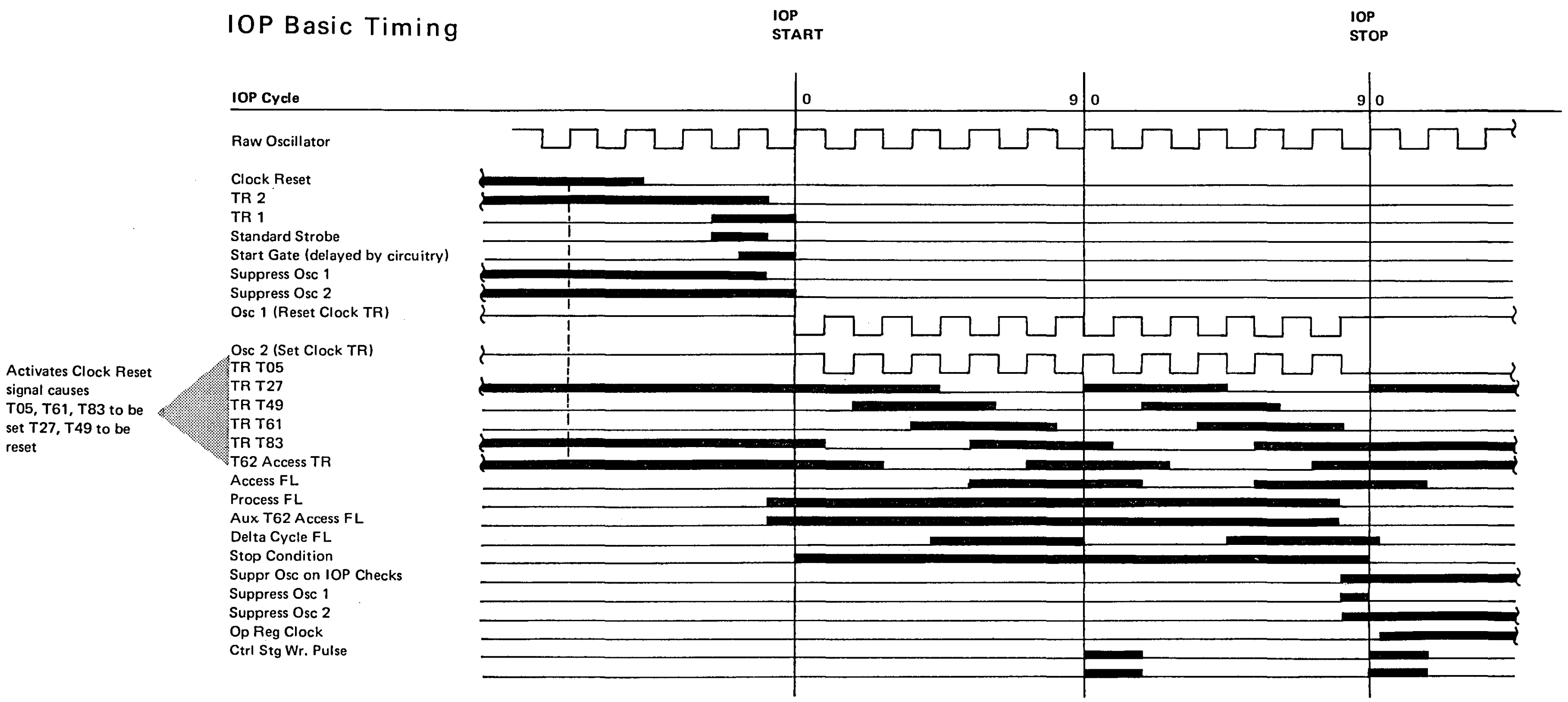
Basic Timing

IOP Start

Before an IOP can be started (after successful IMPL) a single cycle access operation has to be initiated by the SVP. This is a SVP control 'C'.

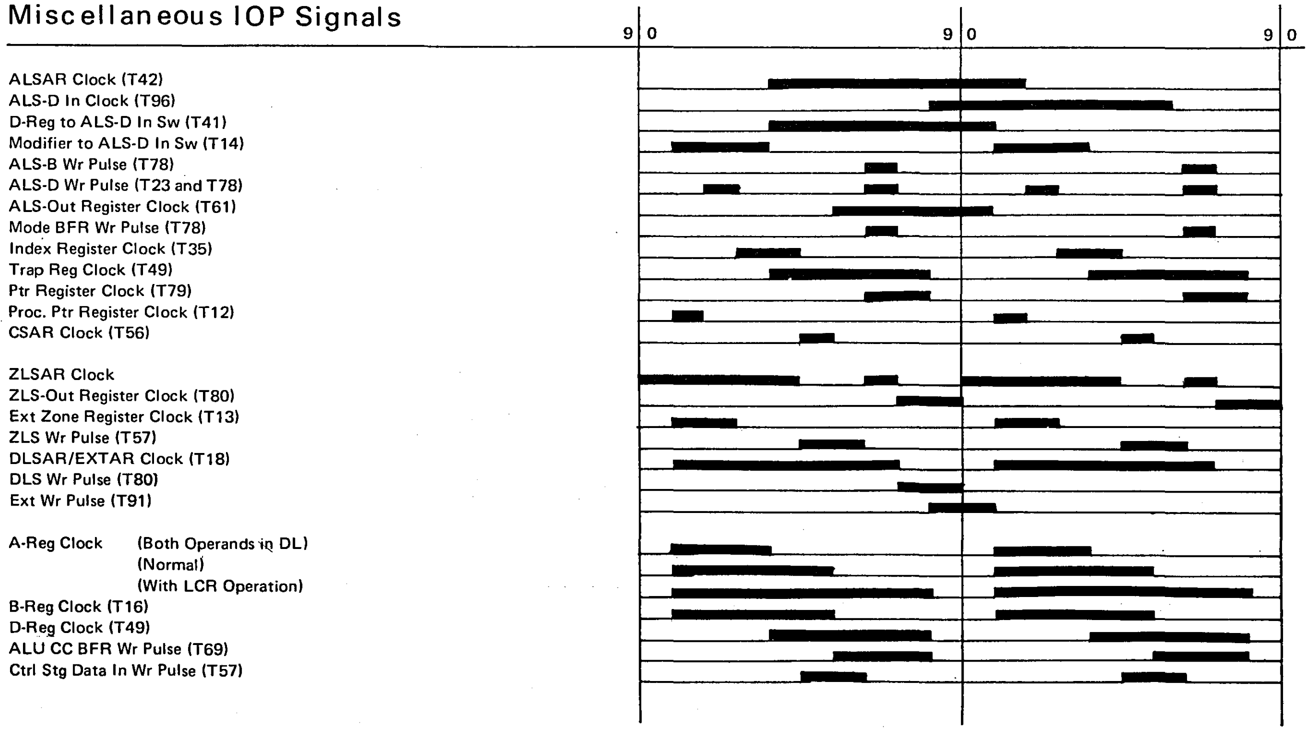
The operation sets up all controls for a service access cycle, which in turn causes the loading of the first instruction to be executed into op-register, (see also Pages 2-180 and 4-050).

IOP Basic Timing



Activates Clock Reset signal causes T05, T61, T83 to be set T27, T49 to be reset

Miscellaneous IOP Signals



This page is intentionally left blank

DLS/ZLS

Data Local Storage (DLS)

Data local storage (DLS) consists of 64 registers, each one-byte wide.

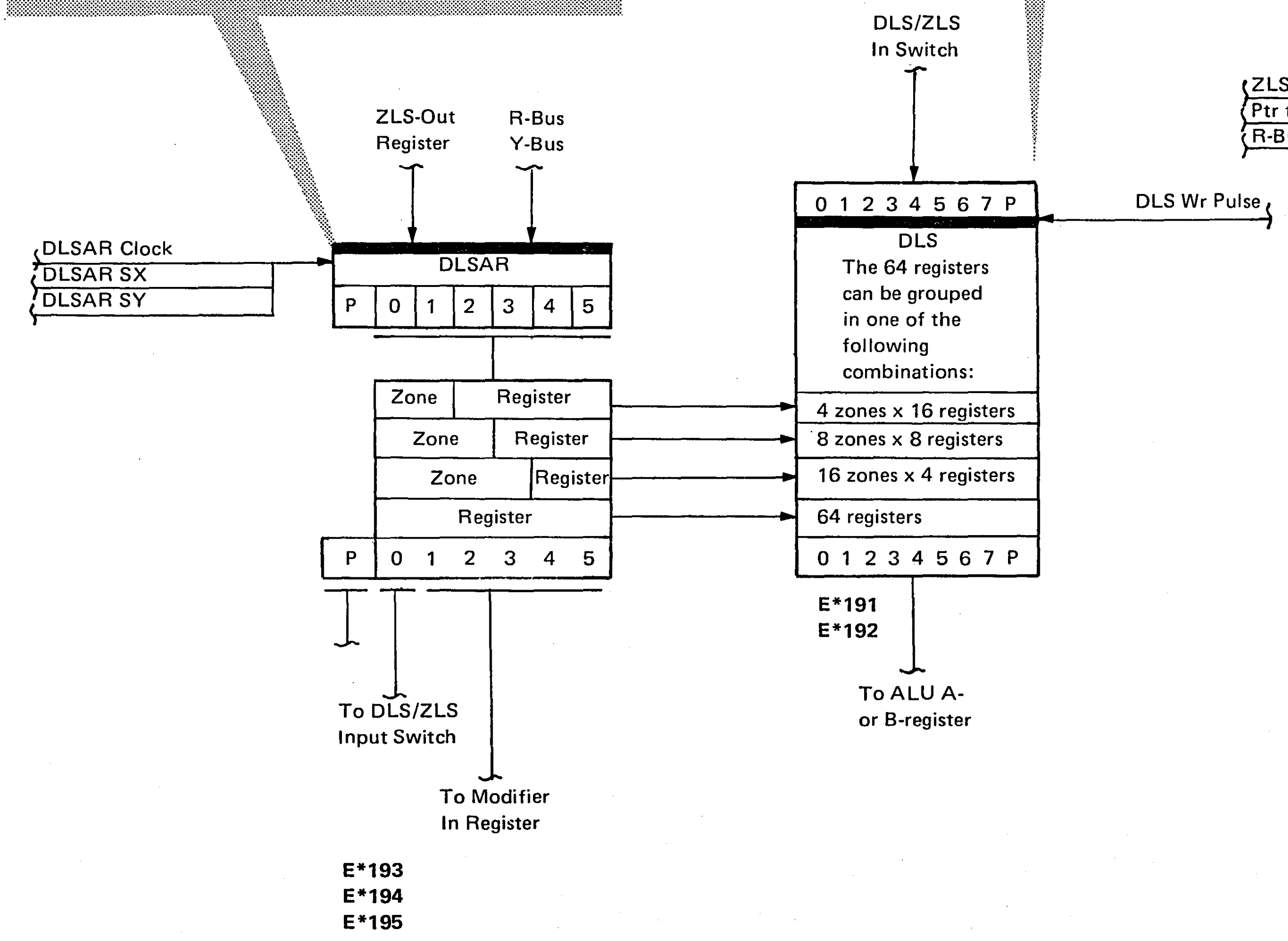
These registers can be grouped (in different zones) and are used as work registers (also called local registers)

These registers represent "To" or "From" registers and hold either data to be processed, or addresses.

During DS cycles, DLS address is taken directly from ZLS, because ZLS holds data addresses for LST operations.

- Data local storage address register (DLSAR) addresses DLS.
- DLSAR contents are constructed from R-bus bits, Y-bus bits, and ZLS out register bits (see Page 4-135).
- R-bus bits and Y-bus bits represent "to" and "from" register addresses.
- ZLS out register bits represent the "zone" within DLS.

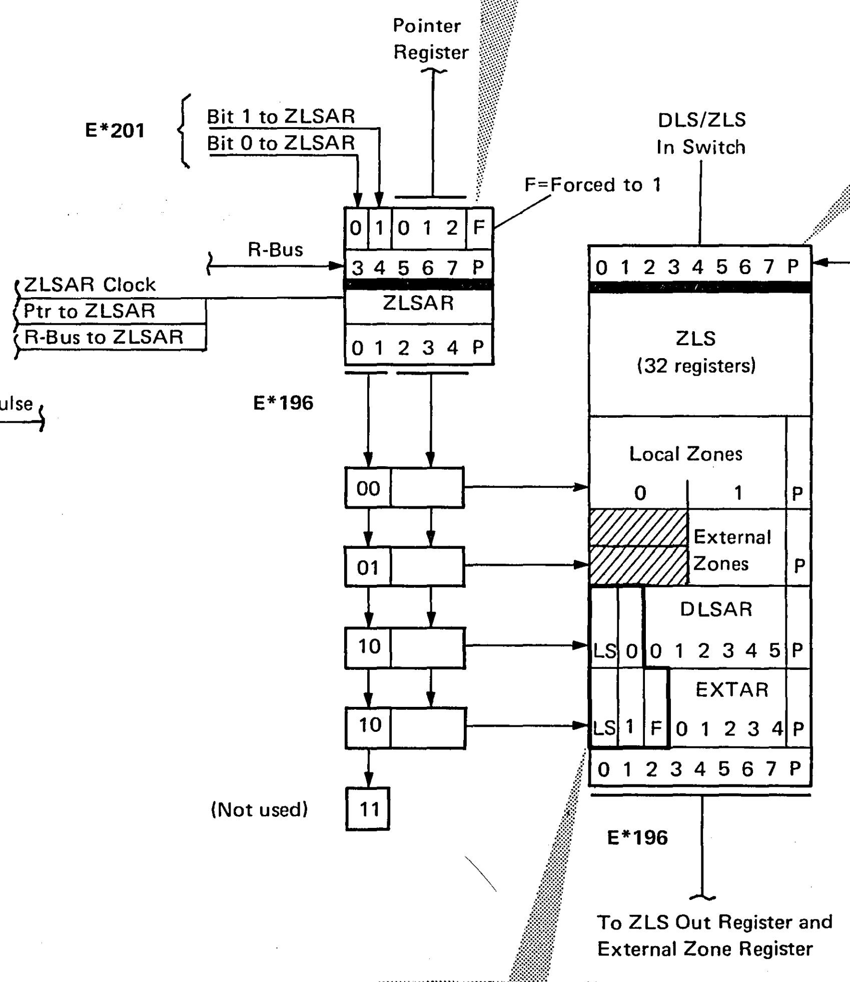
- DLS is addressed by DLSAR.
- Fast access to DLS is provided because the DLS addresses can change three times in one IOP cycle.



Zone Local Storage (ZLS)

- Zone local storage address register (ZLSAR) addresses ZLS.
- ZLSAR contents is normally the pointer, for SZ operations, R-Bus is set into ZLSAR.

- Zone local storage (ZLS) consists of 32 (4 x 8) registers, each one-byte wide.
- ZLS is used for storing addresses (see below).
- ZLS is addressed by ZLSAR.
- The address is the pointer of the currently executed instruction; therefore, each DLS location is assigned to a distinct micro-program routine



ZLS contains:

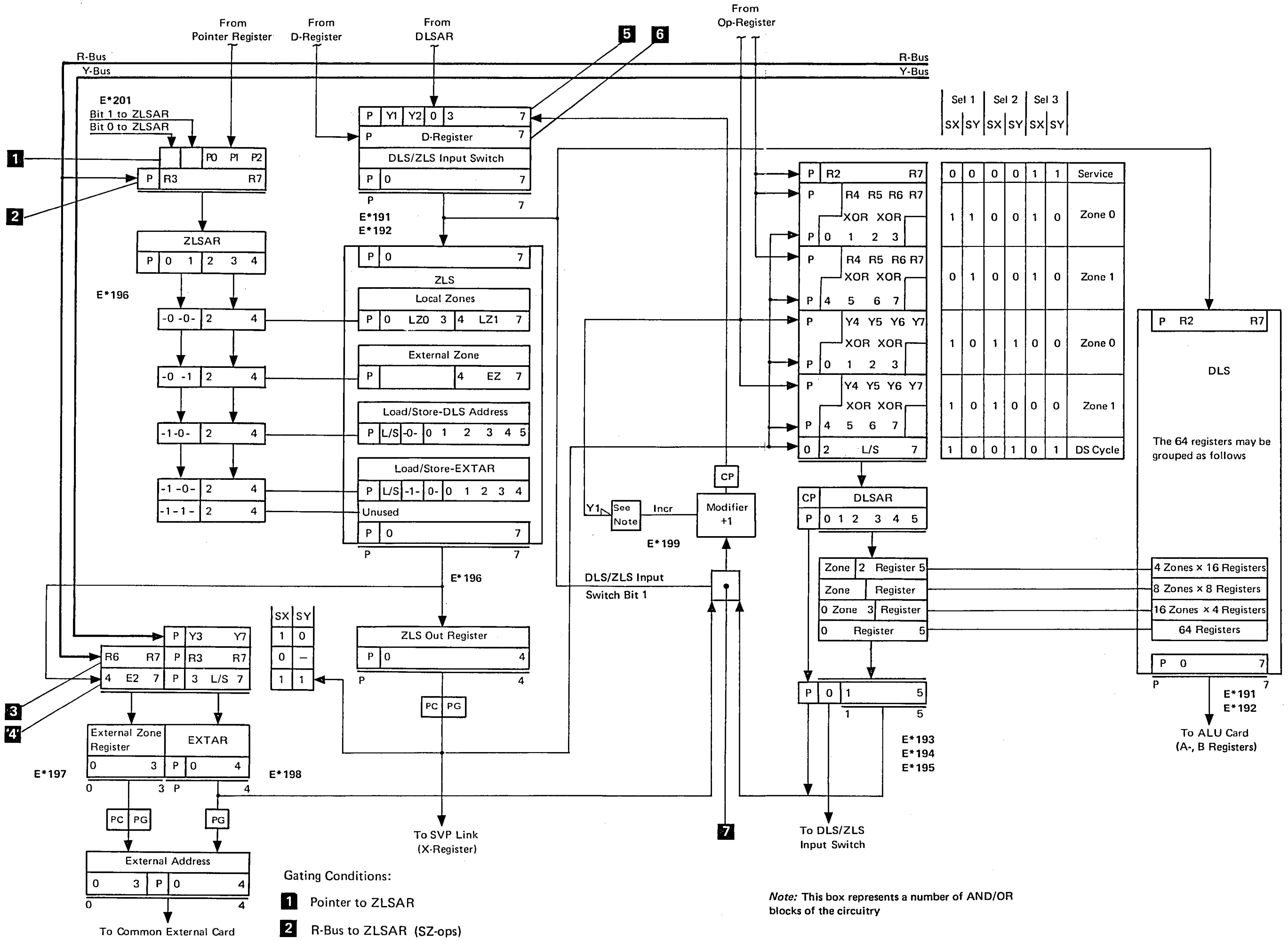
- Local zones for DLS addressing
- External zones for external register addressing.
  These zones represent register groups (see DLS) which may be assigned to different programs or jobs.
- DLS and external register addresses which are held here for the execution of LST operations.
  These addresses have to be held for two reasons:
  1. During DS cycles the actual LST operation, which defines the first data address is overloaded (lost) by data to be handled.
  2. If LST operations are not executed because time slicing becomes active during store access cycle, data register address has to be stored (also the displacement at T78 into ALS-D) until the pointer of the interrupted instruction is used next time.

Bit	Definition
0	If = 0: Store operation If = 1: Load operation
1	If = 0: DLS address If = 1: External address
2	Forced to 0 for external address

Legend

▨ = Not used

ALD PAGES E*19*, 20*

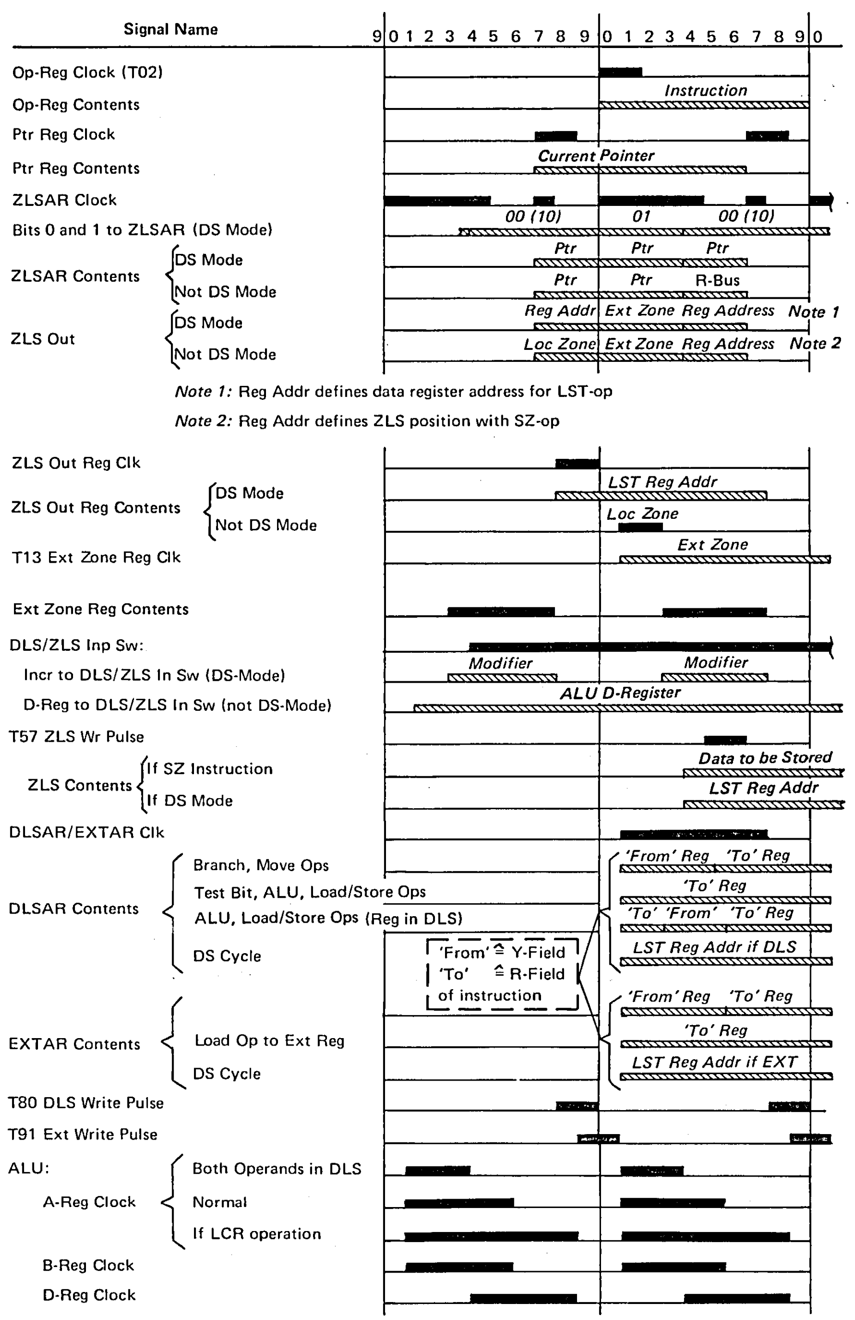


Note: This box represents a number of AND/OR blocks of the circuitry

Gating Conditions:

- **1** Pointer to ZLSAR
- **2** R-Bus to ZLSAR (SZ-ops)
- **3** K2 active (service)
- **4** K2 inactive external zone from ZLS
- **5** Increment to input switch
- **6** D-Register to input switch
- **7** With DLS/ZLS input switch bit 1 (which actually is bit Y2): Active, contents of EXTAR is gated to modifier. Inactive, contents of DLSAR is gated to modifier.

LST Address in ZLS:
Bit 0 – On defines load op
– Off defines store op
Bit 1 – On defines external address
– Off defines DLS address
Bit 2 – For external address, this bit is forced to zero



Arithmetic and Logic Unit (ALU)

- The ALU is used to update information such as data, addresses, and instructions.
- Two operands "A" and "B" are gated via the A-register and the B-register to the ALU. In the ALU, the two operands are logically connected according to the binary decode of the two lines: 'AND/XOR' and 'ADD/XOR'. These two lines are activated according to the op-code of an ALU microinstruction.
- The resulting ALU functions are:

Function	AND/XOR	ADD/XOR
ADD	Inactive	Active
AND	Active	Inactive
OR	Inactive	Inactive
XOR	Active	Active

These four functions may be performed:
(a) With the result being stored in the "To" register.
(b) Without the result being stored in the "To" register (for test purposes only).

- For data-fetch or store operations, operand B is represented by the contents of the digit switch, which contains the amount on increment and decrement. The remaining bit positions (0 to 4) are set to zero for increment, or set to one for decrement.
- Operand A (the displacement part of the data address) is set into the A-register.
- For test bit operations, the bit position that is to be tested is set from the C-bus (bits 5, 6, and 7) into the digit switch.
- The following branch conditions are generated from an ALU zero result and an ALU carry in conjunction with C-bus bits 6 and 7:
  - Carry
  - Zero
  - No carry
  - Not zero
  - Not zero *and* carry
  - Zero *or* no carry
- The ALU condition code buffer allows up to eight ALU conditions to be stored for subsequent branch purposes. It is addressed by a decode of the three pointer bits (process pointer) of the currently executed microinstruction. (ALU CC buffer is not addressable by microprogram.)
- Subtraction is performed by a "complement" ADD under microprogram control.

ALU Operation — Examples

ADD or Increment	**A + B (hex)**	**(A > B) ≅ True**
	A + 9	1 0 0 1
	B + 3	0 0 1 1
	+ C	1 1 0 0
ADD or Decrement	**A − B (hex)**	**(A > B) ≅ Complement**
	A + 9	1 0 0 1
	B − 3	1 1 0 0 (complement)
		①0 1 0 1
		1
	+ 6	0 1 1 0
	A carry out of the high-order position indicates that the result is true. No further complementation is required, but the carry has to be added to the units position.	
ADD or Increment	**A + B**	**(A < B) ≅ True**
	See first example.	
AND	A	1 0 0 1
	B	0 0 1 1
		0 0 0 1
OR	A	1 0 0 1
	B	0 0 1 1
		1 0 1 1
XOR	A	1 0 0 1
	B	0 0 1 1
		1 0 1 0

Specific ALU Signals

As well as the arithmetic result, the ALU generates internally the following signals:

ALU Carry from Bit Zero

This line is active if a carry occurs in the high-order position (bit 0) of the ALU result. The carry is stored in the ALU condition code buffer and is connected to an external pin for other control purposes.

ALU Zero

This line is active if all positions of the ALU result are zero. The zero condition is stored in the ALU condition code buffer.

Predicted ALU Parity

This line presents the correct ALU parity together with the result (that is, in the same cycle). Thus, no additional time is required to generate correct parity of the ALU result.

ALU Carry from Bit 1

This line is not used.

ALU Carry from Bit 4

This line is not used.

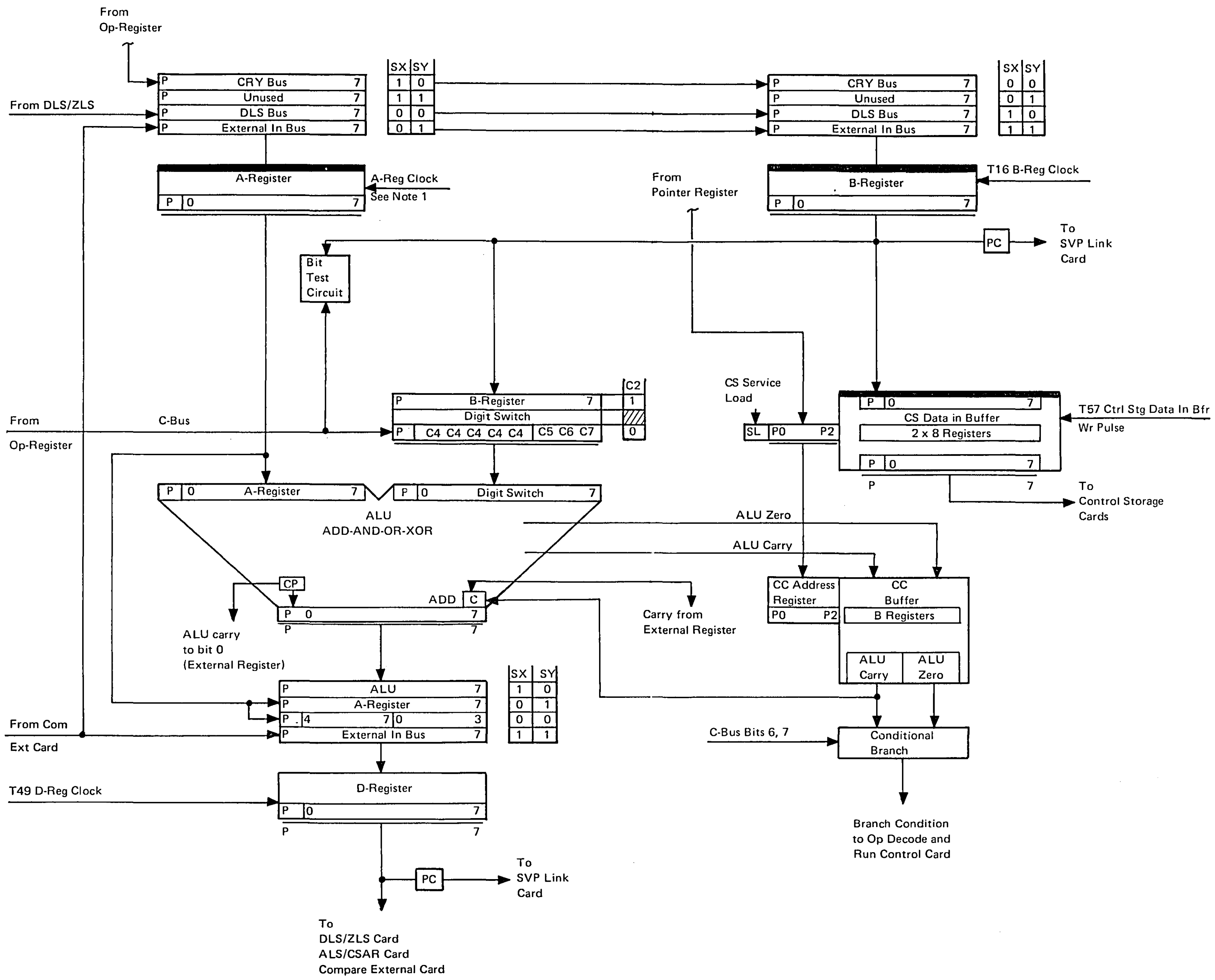
ALU Carry to Bit Zero (Ext)

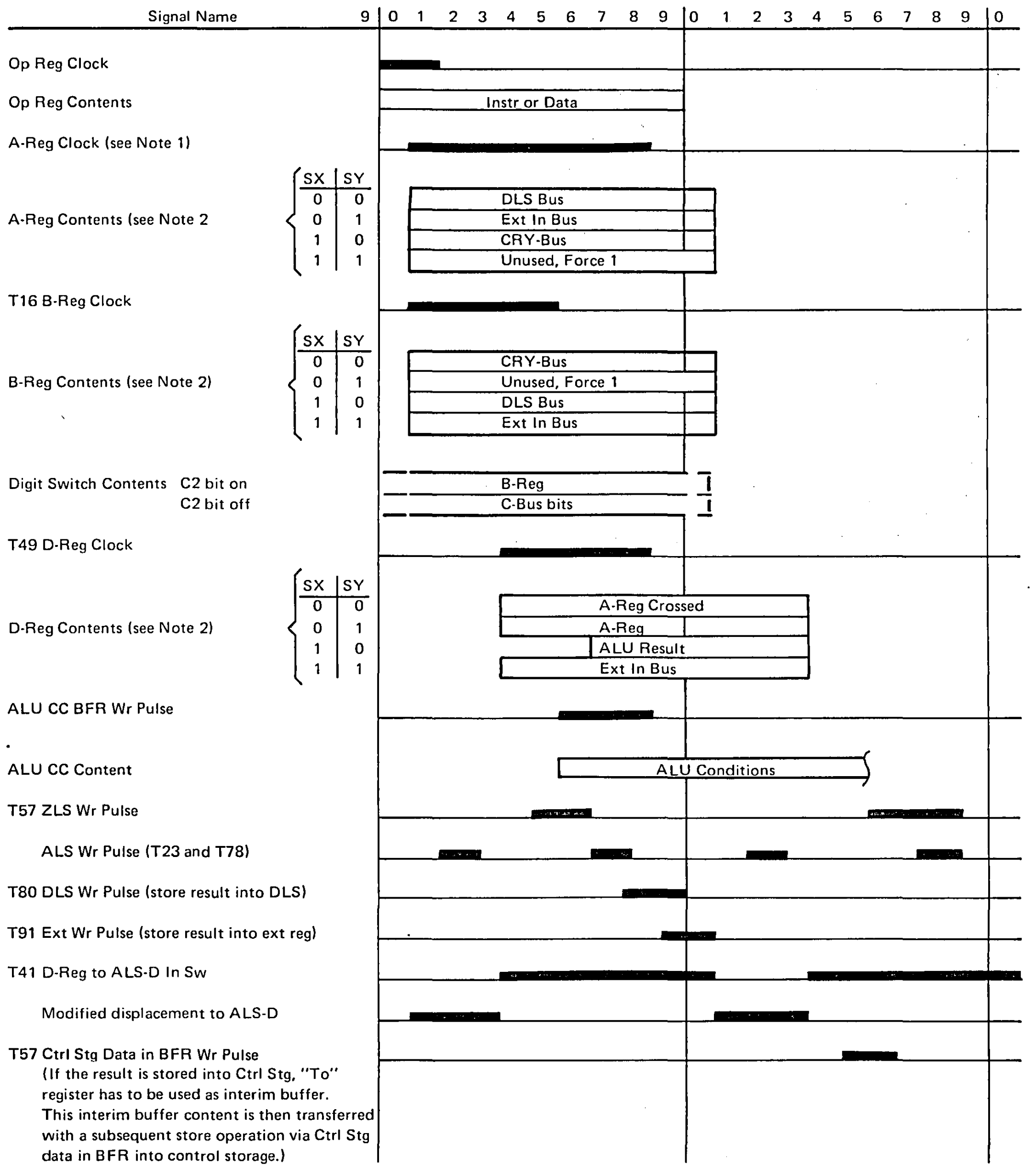
This line is not used.

ALU Carry from Ext

This line is not used.

ALU carry lines that are activated during an ADD operation keep their status until a new carry condition is generated.
A carry cannot occur with AND, OR, and XOR operations. Therefore, the previously activated carry line (which is not reset) causes a carry condition to be stored in the ALU condition code buffer. The carry conditon is stored in the position addressed by the pointer of the AND, OR, or XOR instruction.





Notes:

1. A-Reg clock is of different length.
   T14 If both operands are located in DLS
   T16 Normal
   T19 If LCR ops (SVP link external-Reg is loaded into other work register)
2. Contents of registers may depend upon operation type. For details refer to description of microinstructions in Chapter 3.

Chapter 5. Error Conditions

Reference to Central Test Manual (CTM)

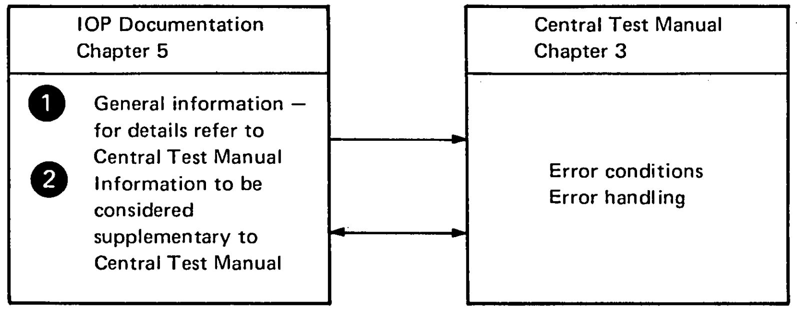
The main purpose of this page is to refer to detailed information on error conditions and error handling in the Central Test Manual Chapter 3.

As the CTM is an EC controlled manual, it always presents information at the latest level.

The contents of this Chapter of the IOP MLM provides:

❶ Only general information on:
  - Error types
  - Cause of errors
  - Error handling
  - Result of errors

❷ Information, that may be considered supplementary to CTM e.g.,
  - Generation of error signals in IOP circuitry.



Unusual or Exceptional Conditions

Unusual or exceptional conditions are caused either by IOP and Front End or I/O device malfunction and therefore are indicated in channel status or unit status.

The table below shows these unusual or exceptional conditions as they are set into either channel status or unit status (See also page 2-015 CSW layout).

Status bytes and sense bytes are delivered from addressed control units and are set into main storage. This information is analysed by the operating system being used, which also initiates corrective actions.

The last section of the table below shows part of sense byte 0, that is common for all I/O devices. Bits 6 and 7 are device specific. For information on the other sense bytes that also contain device specific information, refer to respective I/O or control unit documentation.

Byte	Bit Position	Condition	Cause of Error	Result
Channel Status	1	Incorrect length	Offered or requested data does not correspond with length or byte count	If SLI flag is not set to "on", chaining is suppressed
	2	Program check	One or more of the following conditions: Invalid CCW address; invalid command code; invalid count; invalid data address; invalid key; invalid CAW format; invalid CCW format; invalid sequence	Operation is terminated; chaining is suppressed
	3	Protection check	An attempt to fetch data from, or store data into, a protected storage	In connection with CCW, the operation is not initiated; in connection with data, the operation is terminated and chaining is suppressed
	4	Channel data check	Parity errors with the information transferred to or from main storage	Operation is *not* terminated; chaining is suppressed
	5	Channel control check	Any machine malfunction that affects channel controls. This condition includes parity errors on the CCW and on data addresses	Operation is terminated
	6	Interface control check	Device malfunction such as: Received address or status byte has invalid parity Device responds with an address other than the one specified Device appears not-operational during chaining A signal from the device appears at an invalid time or has an invalid duration	Operation is terminated
	7	Chaining check	Overrun condition during chain data	Operation is terminated; chaining is suppressed
Unit Status	6	Unit check	Unusual conditions in the I/O device (details are given with the information that is delivered by the sense command)	Chaining is suppressed
	7	Unit exception	Indicates a typical condition for any particular command and type of device	Chaining is suppressed
Sense Byte 0	0	Command reject	Device is not designed to execute the particular command that was issued (for example, read to a printer, rewind to disk file)	Program error operation is terminated after the initial selection
	1	Intervention required	A condition that requires some type of intervention at the device (for example, stacker full, hopper empty, printer out of paper, etc)	Operation is not executed
	2	Bus out check	Parity errors on the standard interface	If data parity error, operation is not terminated; if command parity error, operation is not executed
	3	Equipment check	Equipment malfunction (for example, print buffer parity error)	Operation is terminated
	4	Data check	Errors associated with recording medium (for example, reading an invalid card code)	Operation is *not* terminated
	5	Overrun	IOP does not respond in sufficient time to a "request for service" from an I/O device	Operation is *not* terminated

Error Handling

The flowchart shows the principle of handling IOP or Front End errors.

- *Hard* error conditions caused by either external register parity error or IOP circuit errors cause the "IOP halt" bit to be set into the SVP idle sense. (The IOP is stopped.)
- *Sensed* Front End error conditions cause the "PCR" bit to be set into the SVP idle sense. (The IOP is not stopped.)

After an error condition has been handled by the IOP, the IOP requests SVP service by activating a bit in its idle sense bit pattern (see page 3-920).

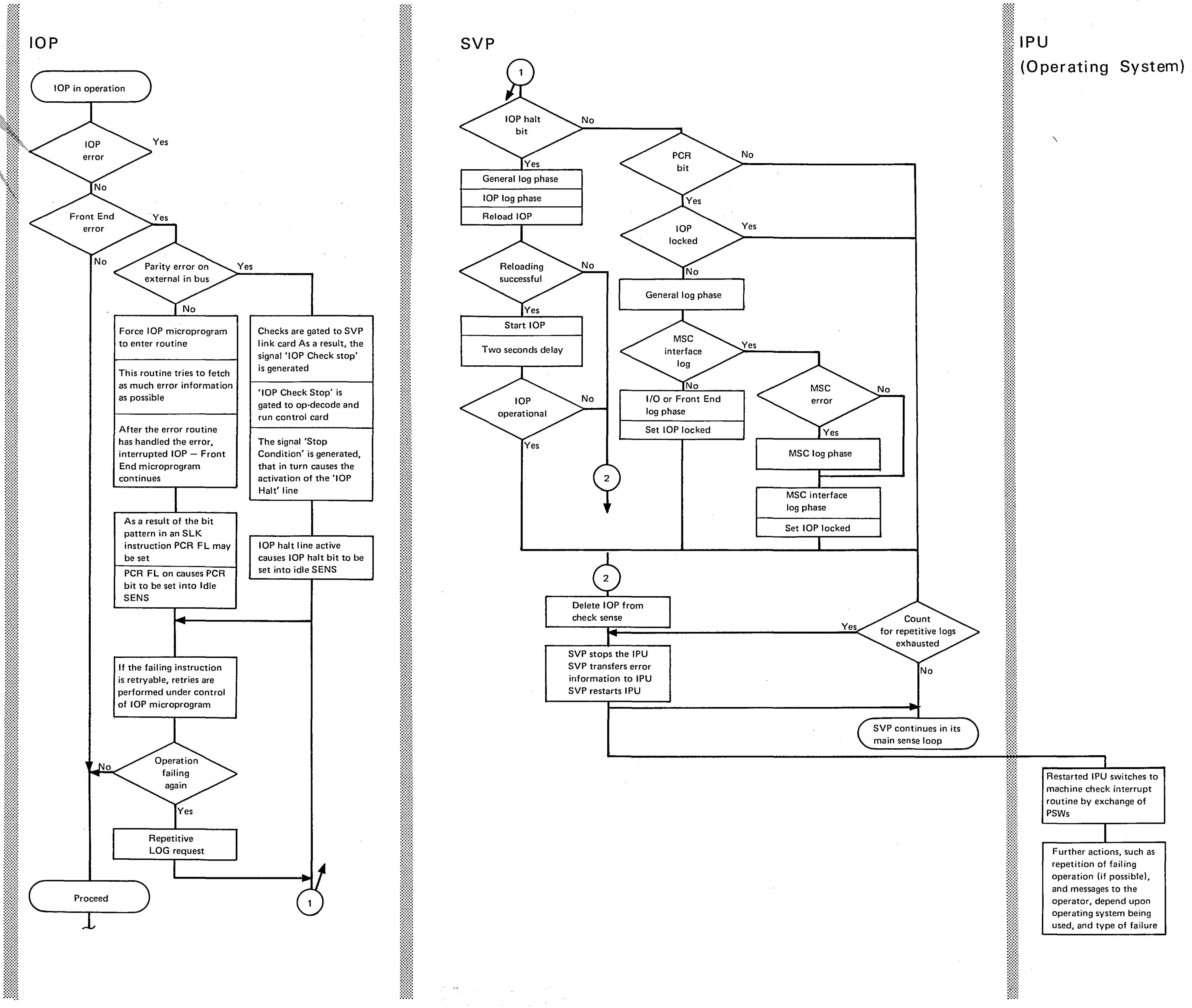
These idle sense bit patterns are periodically checked by the SVP, and the requesting IOP is served under control of the SVP microprogram. The SVP logs all error information.

During "log", the console keyboard remains locked.

Logging

- To ensure error data retention for system malfunction analysis, all solid and intermittent errors are recorded on the system diskette.
- For each IOP, separate log areas are provided to store:
  - Load log information
  - Run log information.
- The log information may be
  - a. Displayed
  - b. Evaluated by log analysis programs
  - c. Erased, if not longer required.

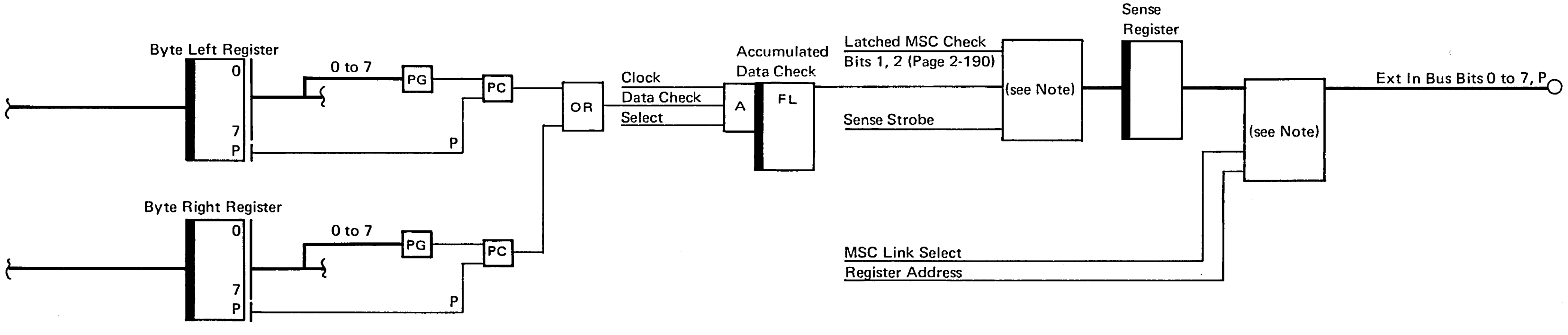
If the error condition cannot be handled by the IOP and SVP, control is transferred to the IPU and the operating system in use.



This page is intentionally left blank

IOP Error Circuits

MSC Data and Control



The sense register contents are gated into the ALU. The IOP microprogram analyzes the bit pattern and decides what actions are to be initiated to correct the error condition. (If possible, the operation during which the error occurred is repeated.)

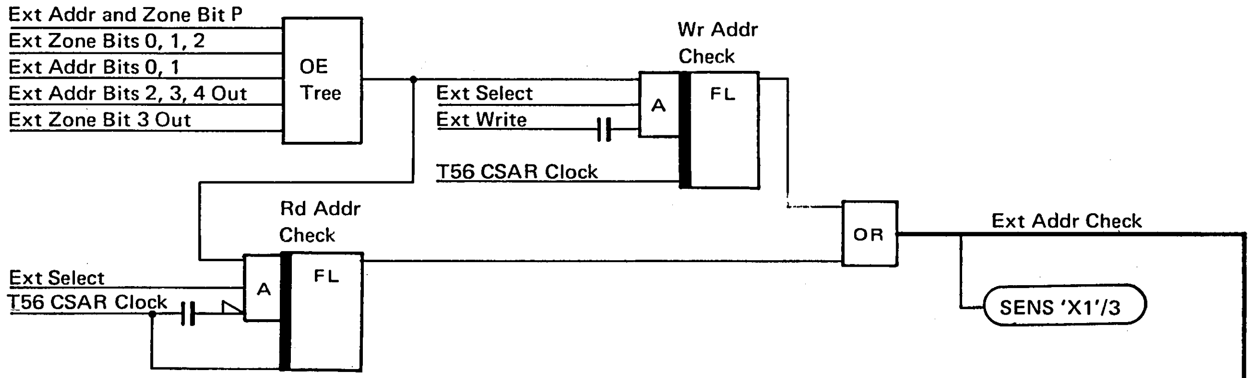
Front End	ALD
MPX	KA 25x, 26x
2560	MG 13x, 14x
3525	MP 13x, 14x
3504	See note below
5425	MM 13x, 14x
1403	See note below

ALDs for 3504 and 1403 have the prefix "MG", "MM", or "MP", according to the I/O configuration

Common External

Front End	ALD
MPX	KA 34X, 35X
2560	MG 10X, 11X
3525	MP 10X, 11X
3504	See note
5425	MM 10X, 11X
1403	See note

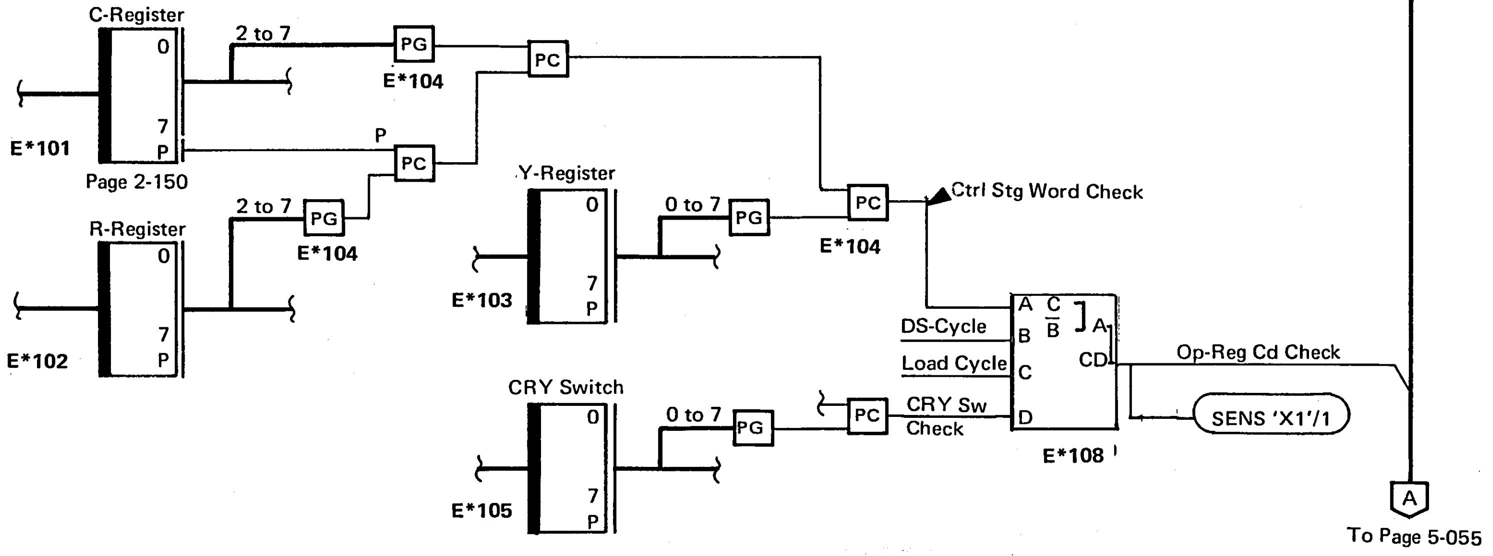
ALDs for 3504 and 1403 have the prefix "MG", "MM", or "MP", according to the I/O configuration



The leading edge of 'alternate sense strobe' signal first resets both flip latches. If the error condition still persists, the 'Rd Addr Check' FL is set again by the trailing edge of the 'Alternate Sense Strobe' signal

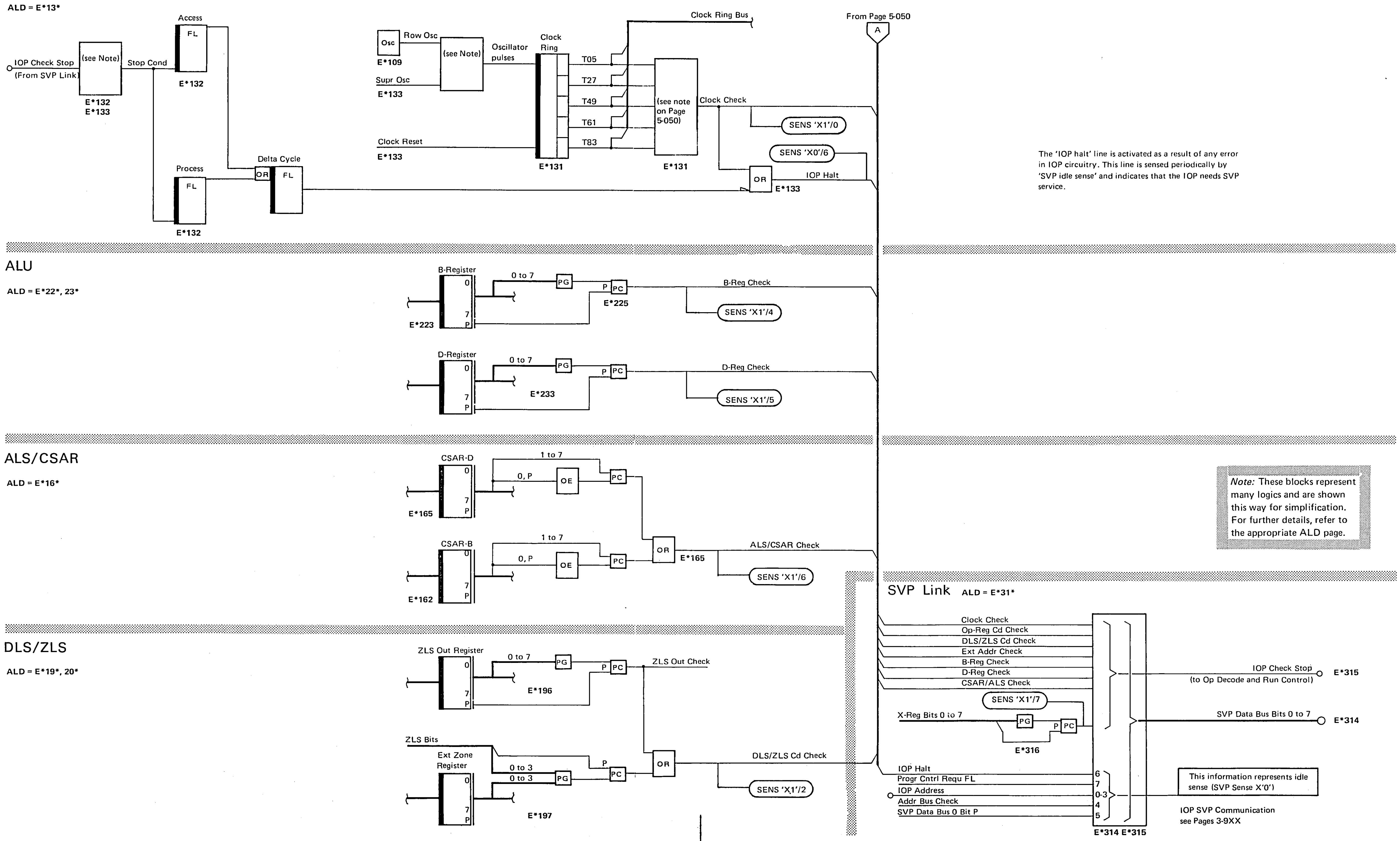
Op-Register

ALD = E*10*



Note: These blocks represent many logics and are shown this way for simplification. For further details, refer to the appropriate ALD page.

Op-Decode and Run Control



The 'IOP halt' line is activated as a result of any error in IOP circuitry. This line is sensed periodically by 'SVP idle sense' and indicates that the IOP needs SVP service.

Note: These blocks represent many logics and are shown this way for simplification. For further details, refer to the appropriate ALD page.

LCL Layout and Sequence Codes

LCL (or ECSW) Layout

The LCL contains model-independent information, that is related to equipment errors detected by the channel or the IOP.

Storage Location (decimal)	**176**				**177**			**178**		**179**				
Bit No	0	1 3	4 7	P	8 12	13 15	P	16 23	P	24	25	28	29 31	P

Bits 0 to 3, 13 to 15, 26 and 27 are set to zero.

Bits 4 to 7 are the *detect field,* which is used to identify the unit that detected the error
- 4 : IPU — detected error
- 5 : IOP — detected error (channel)
- 6 : MSC — detected error
- 7 : (Not used)

Bits 8 to 12 are the *source field,* which is used to indicate the most likely source of the error
- 8 : IPU
- 9 : IOP
- 10 : MSC
- 11 : (Not used)
- 12 : (Not used)

Bits 16 to 23 are the *field validity flags,* which are used to indicate the validity of the information that is stored in designated fields
- 16 : (Not used)
- 17 : (Not used)
- 18 : (Not used)
- 19 : Sequence code valid
- 20 : Unit status valid
- 21 : Command address and key valid
- 22 : Channel address valid
- 23 : Device address valid

Bits 24 and 25 identify *type of termination,* and are coded as follows:
- 00 : Interface disconnect
- 01 : Normal termination, stop, stack
- 10 : Selective reset
- 11 : System reset

Bit 28 is the *I/O error alert bit.* A malfunction reset is performed and the interface control check is set (for channels only, otherwise it is set to zero)

Bits 29 to 31 are the *sequence code.* Indicates the progress of an I/O operation at the time of the channel error. The evaluation of the sequence code depends upon the type of operating system being used. The generation of the sequence code is shown in the flowchart.

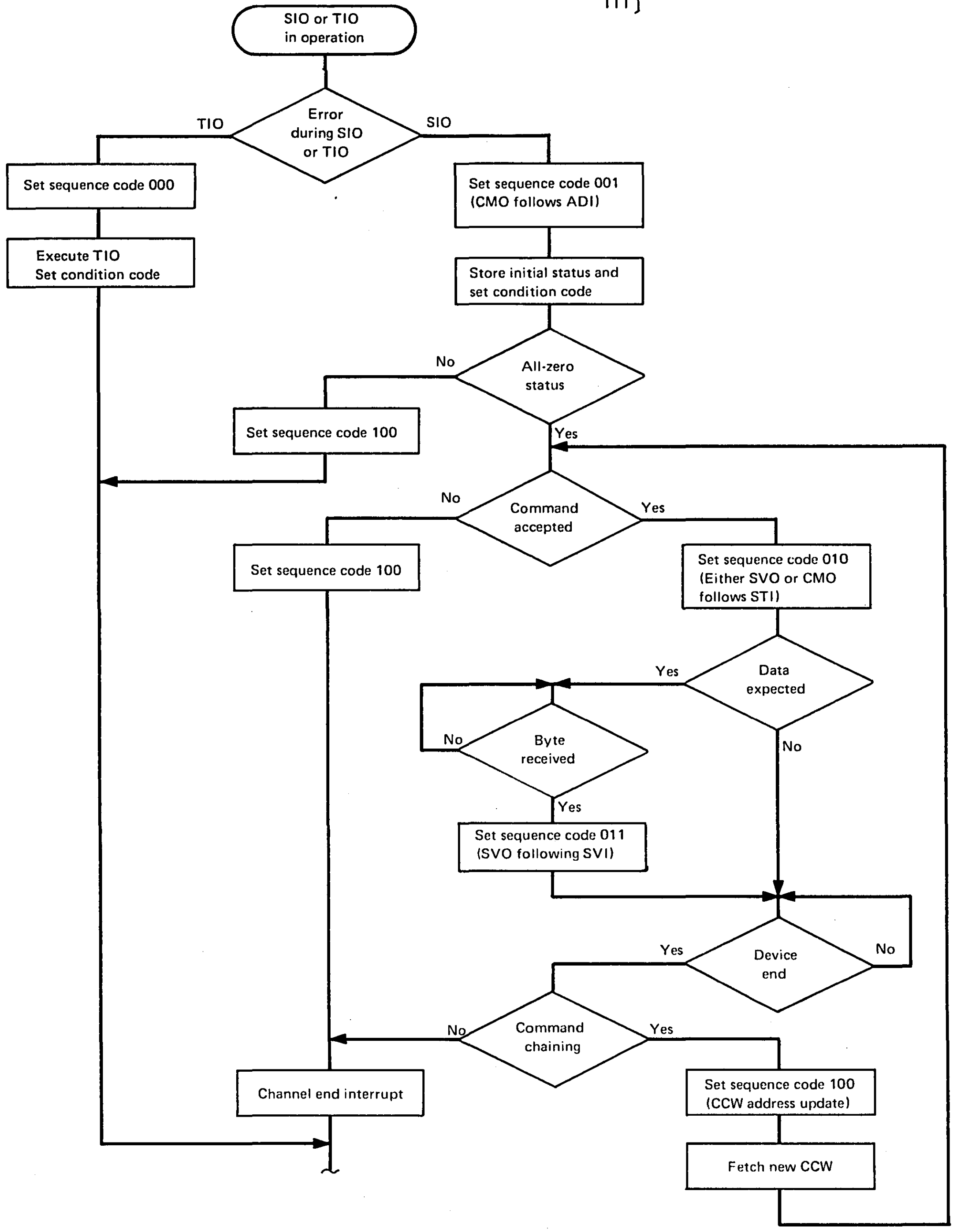
Sequence Code

The sequence code is stored in the LCL and allows correct determination of the point at which the error occurred, and allows correct positioning of the I/O devices for recovery.

Note: The evaluation of the sequence code depends upon the type of operating system being used.

Definition of Sequence Codes

000	Error during execution of TIO
001	Command sent out, status not yet received
010	Command accepted, data transfer not yet started
011	At least one data byte has been transferred
100	Command is either not yet sent out or sent out, but not accepted
101	Command accepted, but data transfer unpredictable
110 111	Reserved



Chapter 6. Maintenance Information

Reference to Central Test Manual (CTM)

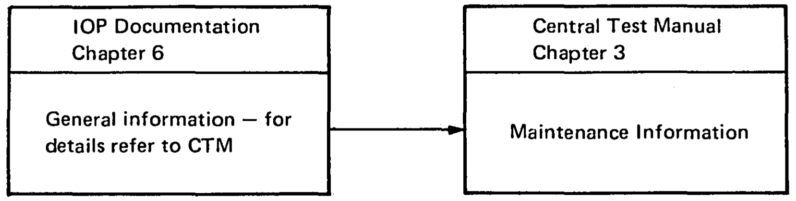
The main purpose of this page is to refer to the details on maintenance information in Chapter 3 of the Central Test Manual.

As the CTM is an EC-controlled manual, it always presents information at the latest level.

The contents of this Chapter of this IOP MLM provides:

Only general information on:

- Maintenance concept
- Diagnostic techniques
- Test programs



Maintenance Concept

- IOPs do not contain any circuits that require adjustments (for example, singleshots, time delays, etc).
- In the event of a malfunction, eliminate the cause by changing cards according to the instructions given on the display unit.
- Whenever cards are changed, check the socket pins.
- The information displayed on the display unit is the result of the log analysis program. The log analysis program analyzes and evaluates log information that was stored either during the IOP load phase or during normal IOP run phase.
- Where more than one card is suspected, the suspected cards are indicated according to the degree of probability for the cause of the fault. This indication is called the *replacement sequence.*
- The replacement sequence is coded as follows:
    1 = High probability
    2 = Low probability
    3 = Very low probability.

Diagnostic Techniques

- Test programs are designed for error detection and error location.
- These test programs either directly refer to the failing field replaceable unit (FRU) or they display detailed test results.
- For communication between the system and the user, use both the keyboard and the display unit.
- Besides normal (or run) mode, test programs may be applied in different modes, according to the setting of K-Register (K-bus) on SVP link card (see Page 4-050).

Manual Operations

IOP manual operations are also to be considered as CE aids.
They allow the CE:

IOP Restart	(Started a previously stopped IOP of a fixed address)
IOP Dump	(Display all informations held in control storage and registers of an IOP)
IOP Fill	(Load ALS, DLS, ZLS with any desired bit patterns)

Scope Sense

- *Scope Sense 1* and *Scope Sense 2* represent two groups of accessible pins on the SVP link card (see Page 2-185).
- Suspected signals may be connected to these pins.
- If the display unit is used as a digital oscilloscope (possibly under control of a special microroutine on the system diskette), selected signals can be displayed and compared with other reference signals.
- Signals that are connected to the scope sense pins are also logged. This allows additional conditions to be stored for later analysis.

Matrix

Two matrix types may be selected.

Both matrix types are represented by a 7X 10 X pattern.

Each X here represents a No Op, that may be replaced by either a SVP SENS or SVP CTRL operation.

This allows the composing of a specific routine to run an IOP under particular conditions.

After the selected IOP is started the IOP loops in that routine. If K-register was previously set, this routine may also be 'single cycled'.

The matrix selected by a Y requires 4 digits per instruction to be keyed in and is called "single matrix".

The use of this matrix allows the running of *one* IOP.

The matrix selected by a Y requires 5 digits per instruction to be keyed in and is called "multiple matrix". The use of this matrix allows the running of *several* IOP and/or subprocessors or adapters.

The first digit to be keyed in represents the address.

Chapter 7. Reference Information

Abbreviations and Glossary

A

A-Reg	A-register (ALU input)
access cycle	during this cycle an instruction is fetched from control storage
addr	address
ALS	Address local storage (HDB 32 x 18)
ALS B	contains block address or index word
ALS D	contains displacement or index word
ALS in switch	circuitry to gate data into ALS
ALS out register	register to gate ALS contents to the different users
ALSAR	address register for ALS
ALU condition code bfr	8-register-wide buffer to store ALU carry and zero condition
ALU	arithmetic and logic unit (performs arithmetic and logic operations)
ASCP	automatic system checkout program

B

B-Reg	B-register (ALU input)
bfr	buffer (fast interim storage device)
block	comprises 128 microinstructions or 256 data bytes in control storage
branch condition	a condition used for logical decisions
bus register	register that holds bus information. This actually comprises two registers called "byte right register" and "byte left register". IOPs communicate with the IPU/MSC via these registers

C

C-Bus	exit of C-register
C-Reg	part of op-register (usually represents op-code)
CAW	channel address word
CCL	chain control line
CCW	channel control word
CC	can have different meanings. If used in connection with CCW, it means chain command. Is sometimes used for condition code
cond code	condition code
CD	chain data
CIO	clear I/O
clk	clock
clock	generator of basic time intervals
clock ring bus	timing distribution
clock step FF	controls stepping of clock trigger ring
common register	register in the local storage of MSC
compare	circuitry that checks whether two operands are equal or unequal
CPL	control program load (load operation under control of SVP)
CS data register	control storage input register
CSAR	address register of control storage B-block, D-displacement
CSW	channel status word
CS	control storage. CS is also used for core storage, and for cycle steal
ctrl stg	control storage

D

D-Reg	D-register (ALU output)
DAR	data address register (in ALS)
DCD	decoder
decr	decrement (normally used for address updating)
delta cycle	a shifted access or process cycle
digit switch	allows selection in ALU input B-circuitry
displacement	low-order part of address for control storage or work register
DLS	data local storage
DLS/ZLS input switch	circuitry to gate information into either DLS or ZLS
DLSAR	address register for DLS
DS	data storage
DS mode	data store mode

E

ECSW	extended CSW
ext	external registers (up to 64, addressable)
ext zone reg	register in the ZLS used to address externals in connection with EXTAR
EXTAR	register for external addresses

F

FF	flipflop = trigger
FL	fliplatch

G

gt	gate

H

HDB	high density buffer (used as registers)
HIO/HDV	halt I/O or halt device

I

ID	identifier
IDA	indirect data address
IMPL	initial microprogram load
incr	increment (normally used for address updating)
index word	controls execution of microprogram and consists of pointer and link
index register	a byte-wide register that contains pointer and link
instruction cycle	one IOP cycle of 450 ns
instruction identifier	generated in IPU, represents op code
IPL	initial program load
IPU	instruction processing unit
IPU tag register	contains control information for IPU
irpt	interrupt

L

LCL	limited channel logout
link	part of the index word (linking to next index word)
load	data transfer from DS into "to" register
local store	part of the MSC storage

M

MIAR	main routine instruction address register (in ALS)
microinstruction	instruction of the IOP microprogram (22-bits wide)
MLM	maintenance library manual
mode bfr	contains IOP status
mode bits	stored in mode buffer (determines IOP status)
mode reg	contains IOP status read from mode buffer
modifier	used to modify addresses
MSC	main storage controller
MSC tag register	contains control information for MSC

N

NA	not available, or not applicable
NS	not shown
NSIA	next sequential instruction address (SVP)

O

OCL	octopus control line
op-decode	performs decoding of operation codes, which are part of an instruction
op-register	contains the currently executed instruction
OPSTAT	operation status (shows process of operation)

Abbreviations and Glossary (continued)

P

PCI	program-controlled interrupt
PCR	program-controlled request
pointer	part of the index word (pointing to the next instruction to be executed)
pointer reg	contains pointer
POR	power on reset
process cycle	during this cycle an instruction is executed
ptr	pointer

R

R-bus	exit of R-register
R-reg	part of op-register that usually represents "to" register
requ	request
RI	read-in (store operation)
RO	read-out (fetch operation)
Rt	right
run control	checks the conditions under which the IOP is allowed to run, or has to stop

S

sense register	a byte-wide register into which control signals from MSC-common are loaded
SIAR	subroutine instruction address register (in ALS)
SILI	suppress incorrect length indication
SIO (F)	start I/O (fast release)
SLI	see SILI
SR mode	subroutine mode (when SIAR is used instead of MIAR)
STIDC	store channel identifier instruction
store	data transfer from "from" register into control storage
SVP	service processor

T

test bit	information to be used as branch condition
TIC	transfer in channel
time slicing	comparable with multiplex mode. Allows more than one device to be run at one time
TIO	test I/O
trap register	contains trap bits (from the devices) ORed with the current link
trapping	allows switching to other index words if a device requires service
TS	time slice (mode) on time slicing
TSCON	time slicing and trapping control
TSR	time slice register (ZLS and ALS)

U

UCW	unit control word

W

wr	work register (DLS and external)
write gate	signal to control buffer on read/write operations

X

X-fer	transfer
X-register	transfer register in SVP link

Y

Y-bus	exit of Y-register
Y-RG	part of op-register. Usually represents "from" register or immediate data

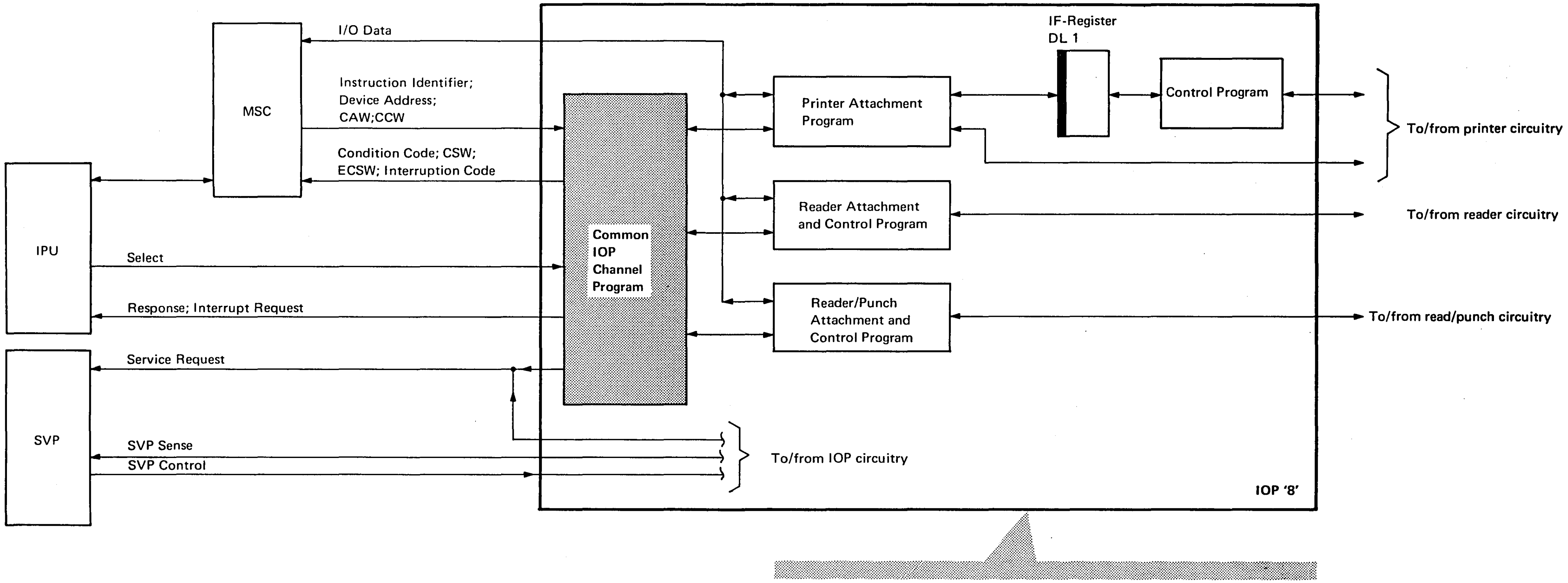
Z

ZLS	zone local storage
ZLSAR	address register for ZLS
zone	group (4, 8, 16) of DLS or external registers

Appendix A. Information Particular to IOP '8'

Common IOP Channel Program Link to IPU, MSC, SVP, and Attachment Programs

Connections to IPU, MSC, and SVP are set up by the common IOP channel program, with one exception: I/O data transfers are controlled by the individual attachment programs.



The following programs are stored in the control storage of IOP '8':

- Common IOP channel program (see Page 3-510)
- Attachment programs } documented in the appropriate Front End manual
- Control programs }

The common IOP channel program and the attachment programs directly communicate together, but the attachment programs and the control programs communicate via registers that are located in the DLS of the IOP control storage.

The multiplexer channel controlling microprogram is documented in *IBM 3125 Processing Unit, Multiplexer Channel,* Maintenance Library Manual, Order No. SY33-1067. The direct disk attachment microprogram is documented in *IBM 3125 Processing Unit, Direct Disk Attachment,* Maintenance Library Manual, Order No. SY33-1073.

UCW (Unit Control Word)

- The UCWs are located in the IOP control storage.
- The UCW format shown on this page is used in IOP '8' connected to card I/Os and printer.
- The multiplexer channel (served by IOP '9') uses a different format; see *IBM 3125 Processing Unit, Multiplexer Channel,* Maintenance Library Manual, Order No. SY33-1067.

UCWs can be considered as interim storages that hold device-oriented information necessary to run devices in byte (or multiplex) mode. This means that one UCW has to exist for each device.

If an 'SIO' instruction is issued to device A, device A is started and the UCW for device A is prepared. While the controlling IOP is waiting for a 'service request' from device A, the IPU continues the processing of its own programs.

If now, an 'SIO' instruction is issued to device B, device B is also started and the UCW for device B is prepared. The system is again released and the controlling IOP is now waiting for a 'service request' from device B, as well as from device A.

As soon as a device requires service, the device activates its 'request bit'. This request bit directly influences microprogram control. The UCW of the requesting device is loaded from control storage into working area and, the device is serviced, under control of this UCW. The UCW is then updated and stored again.

With the next 'service request' from another device the UCW of that device is loaded from control storage into the working area and, the device is serviced under control of this UCW, the UCW is then updated and stored again.

The starting of more than one device byte mode gives the impression that these devices are operating simultaneously. Actually, only the mechanical operations of the started devices are running simultaneously and one device only is serviced at one time. This is because all devices use the same circuitry and the same microprogram steps.

For further information, see Page 3-200, where "time slicing" is discussed.

00	01	02	03	04	05	06	07
Device Address	Flags of Previous CCW	Command Code	Data Address Medium	Data Address Low	1. Sense Byte 0 (from channel end to initial selection) 2. Data Address High (from initial selection to channel end)	Flags	Bit 0: TIC Allow Bits 1 to 3: Sequence Code Bit 4: Unit Check with TIO Bits 5 to 7: Op-status

08	09	0A	0B	0C	0D	0E	0F
Bits 0 to 4: Key Bits 5 to 7: Zero	Next CCW Address High	Next CCW Address Medium	Next CCW Address Low	Unit Status	Channel Status	Byte Count (see Note 1)	Byte Count (see Note 1)

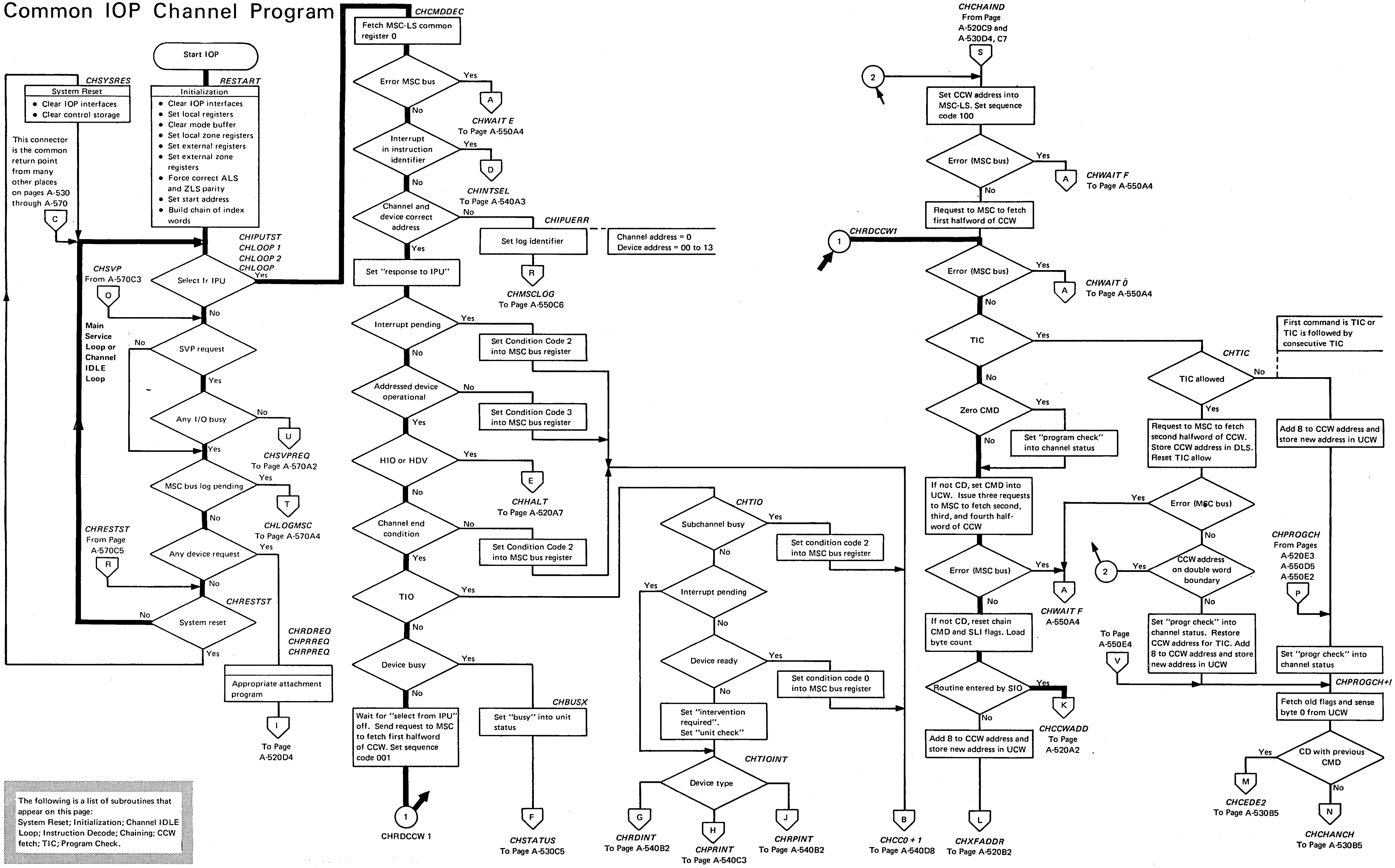
10	11	12	13	14	15	16	17
ECSW Detect field (Storage location 176 decimal)	ECSW Source field (Storage location 177 decimal)	ECSW Validity field (Storage location 178 decimal)	ECSW Sequence code (Storage location 179 decimal)	Saved Buffer Block Address (see Note 2)	Saved Buffer Display Address (see Note 2)	Saved I/O Count (see Note 3)	Saved Transfer Loop Identifier (see Note 4)

18	19	1A	1B	1C	1D	1E	1F
Saved MSC Tags	Saved I/O Data Byte (see Note 5)	Saved Sense Byte 0	Saved Work Register	(Not used)	IDA High	IDA Medium	IDA Low

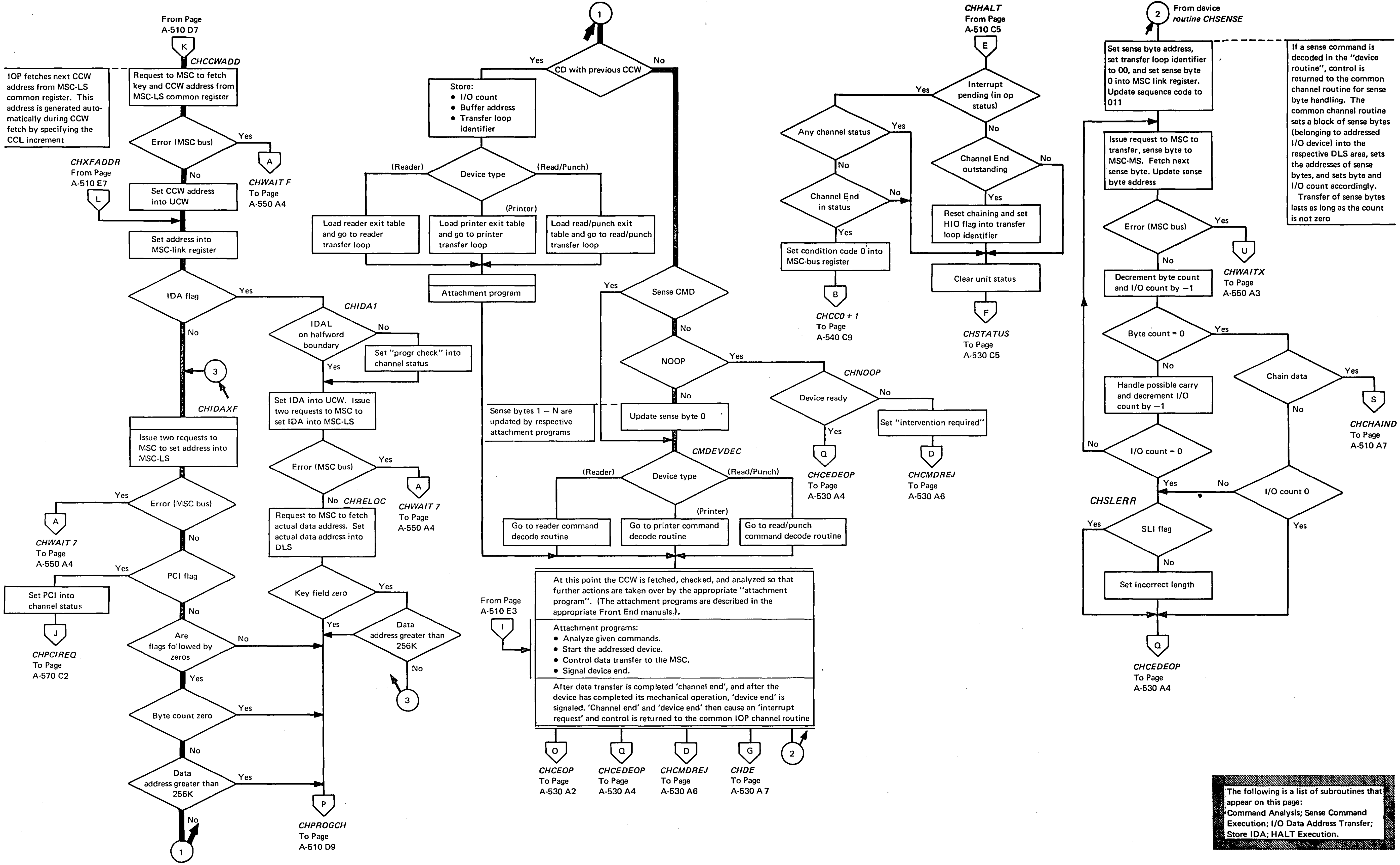
Notes:

1. *Byte Count* is defined as being the number of bytes that are to be transferred under control of the programmer.
2. *Buffer Address* contains the address of the IOP buffer location from where data is fetched on write operations, or into which data is stored on read operations. Data transfer is effected by the control program of the I/O device.
3. *I/O Count* is defined as being the number of bytes that can be processed by the I/O device.
4. *Transfer Loop Identifier* identifies the routine. It is used when urgent steps have to be performed first, and provides a return to the routine from which the microprogram left.
5. *Saved I/O Data Byte* is used as an interim storage for data bytes when urgent steps have to be performed first (for example, with relocation).

Common IOP Channel Program

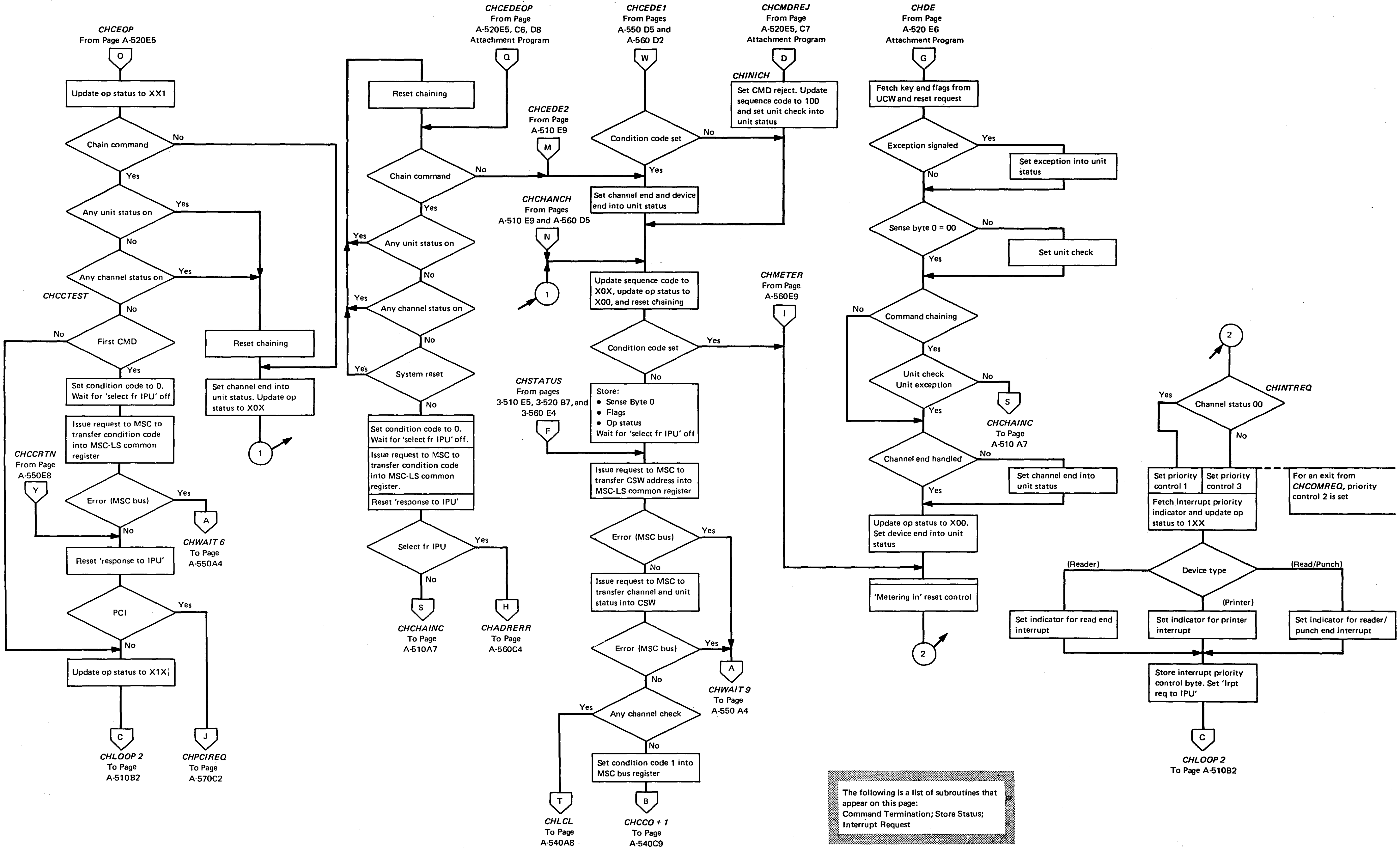


The following is a list of subroutines that appear on this page:
System Reset; Initialization; Channel IDLE Loop; Instruction Decode; Chaining; CCW fetch; TIC; Program Check.

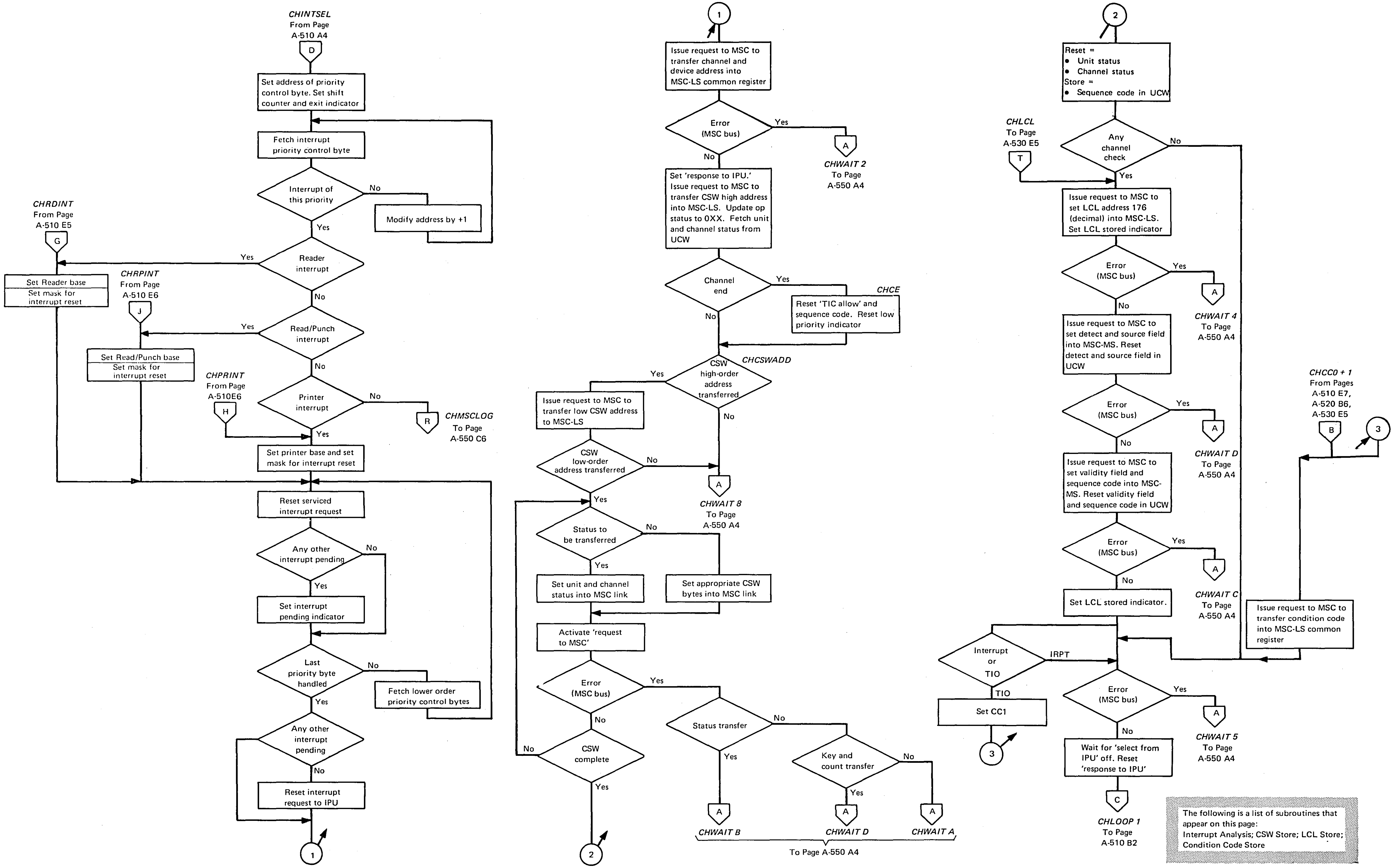


The following is a list of subroutines that appear on this page:
Command Analysis; Sense Command Execution; I/O Data Address Transfer; Store IDA; HALT Execution.

Common IOP Channel Program (continued)

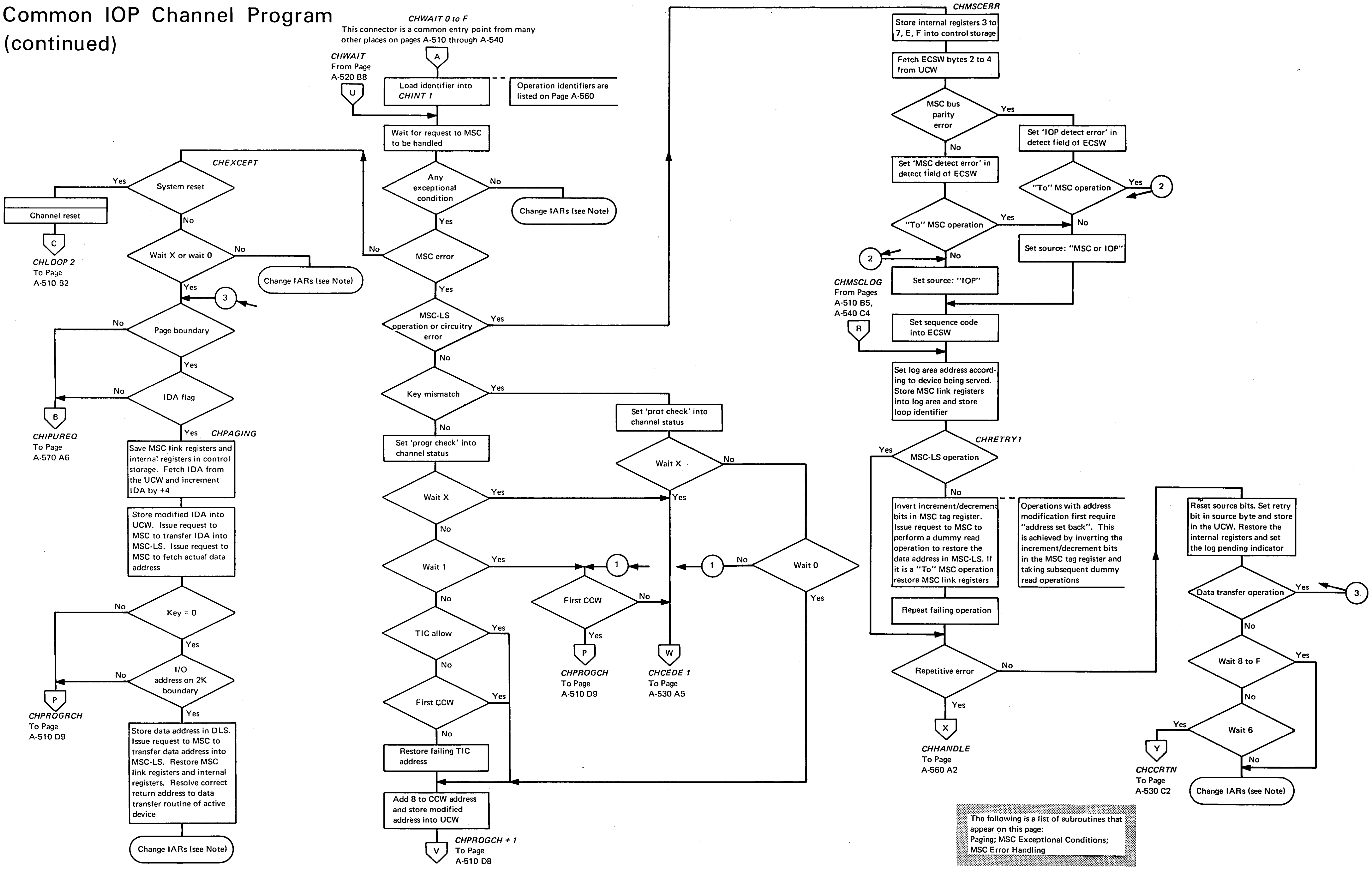


The following is a list of subroutines that appear on this page:
Command Termination; Store Status; Interrupt Request



The following is a list of subroutines that appear on this page: Interrupt Analysis; CSW Store; LCL Store; Condition Code Store

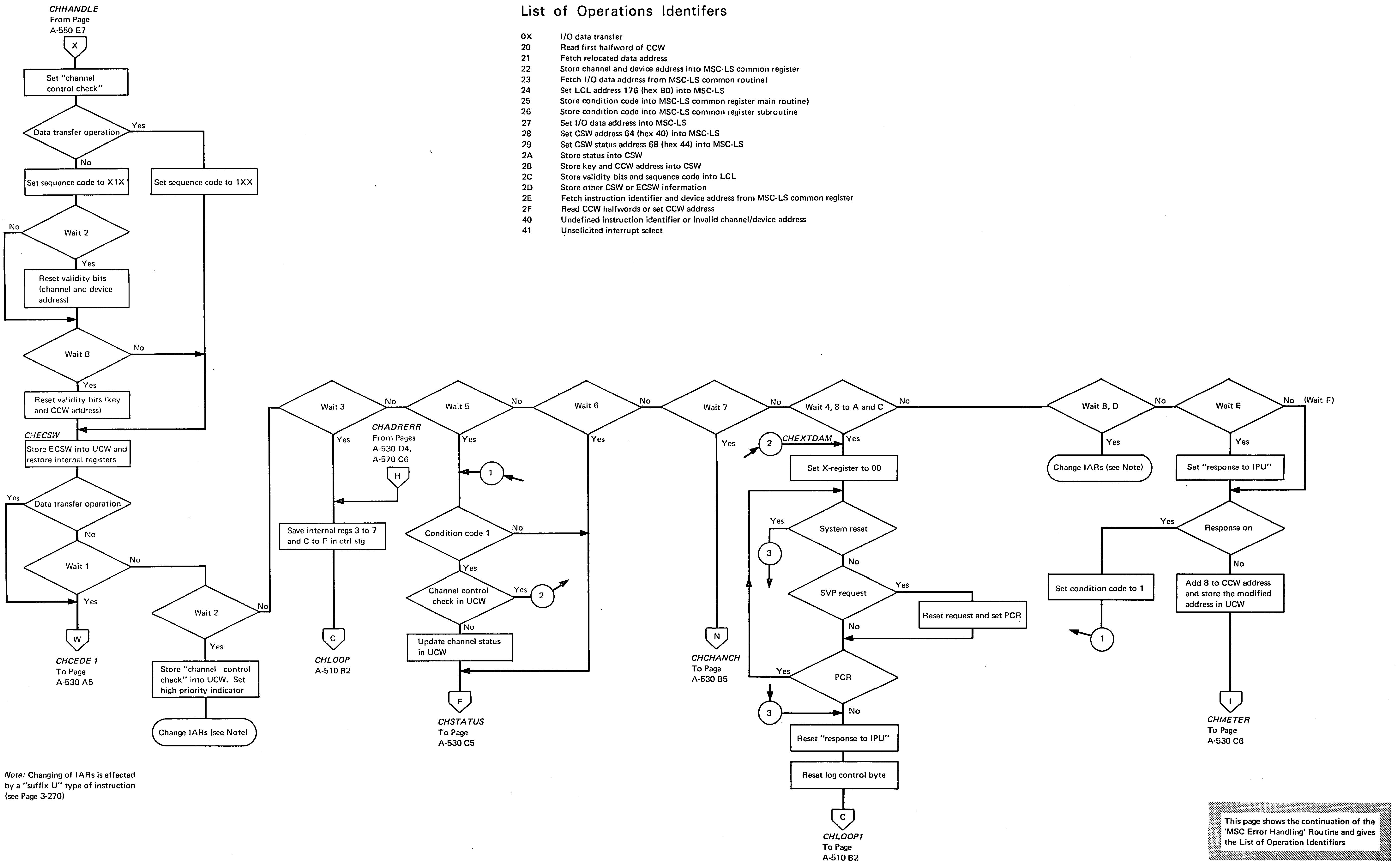
Common IOP Channel Program (continued)



The following is a list of subroutines that appear on this page:
Paging; MSC Exceptional Conditions; MSC Error Handling

List of Operations Identifers

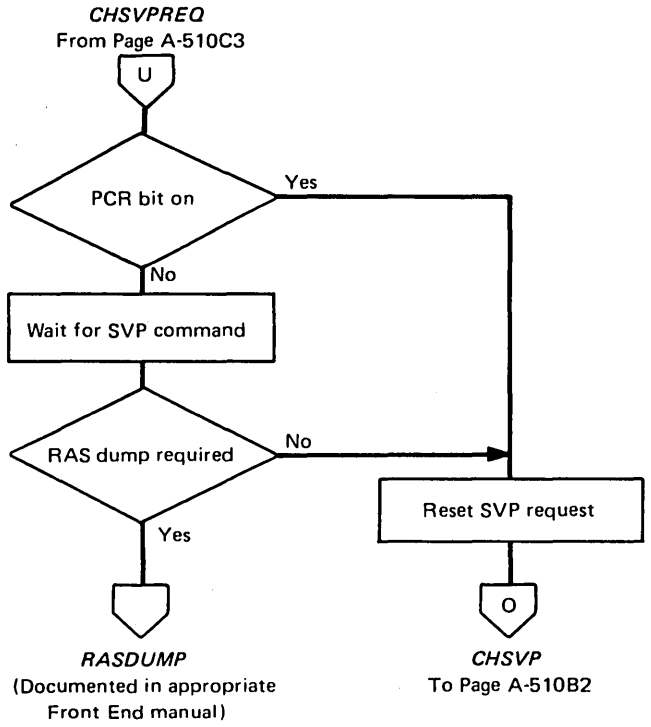
0X	I/O data transfer
20	Read first halfword of CCW
21	Fetch relocated data address
22	Store channel and device address into MSC-LS common register
23	Fetch I/O data address from MSC-LS common routine)
24	Set LCL address 176 (hex B0) into MSC-LS
25	Store condition code into MSC-LS common register main routine)
26	Store condition code into MSC-LS common register subroutine
27	Set I/O data address into MSC-LS
28	Set CSW address 64 (hex 40) into MSC-LS
29	Set CSW status address 68 (hex 44) into MSC-LS
2A	Store status into CSW
2B	Store key and CCW address into CSW
2C	Store validity bits and sequence code into LCL
2D	Store other CSW or ECSW information
2E	Fetch instruction identifier and device address from MSC-LS common register
2F	Read CCW halfwords or set CCW address
40	Undefined instruction identifier or invalid channel/device address
41	Unsolicited interrupt select

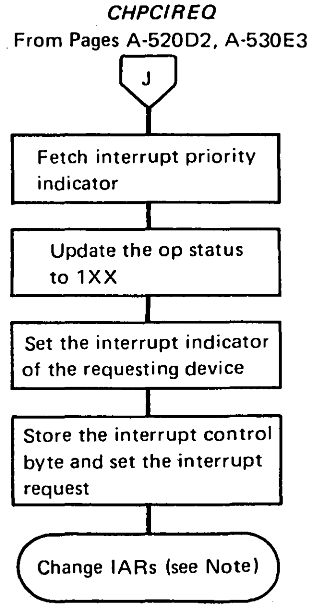


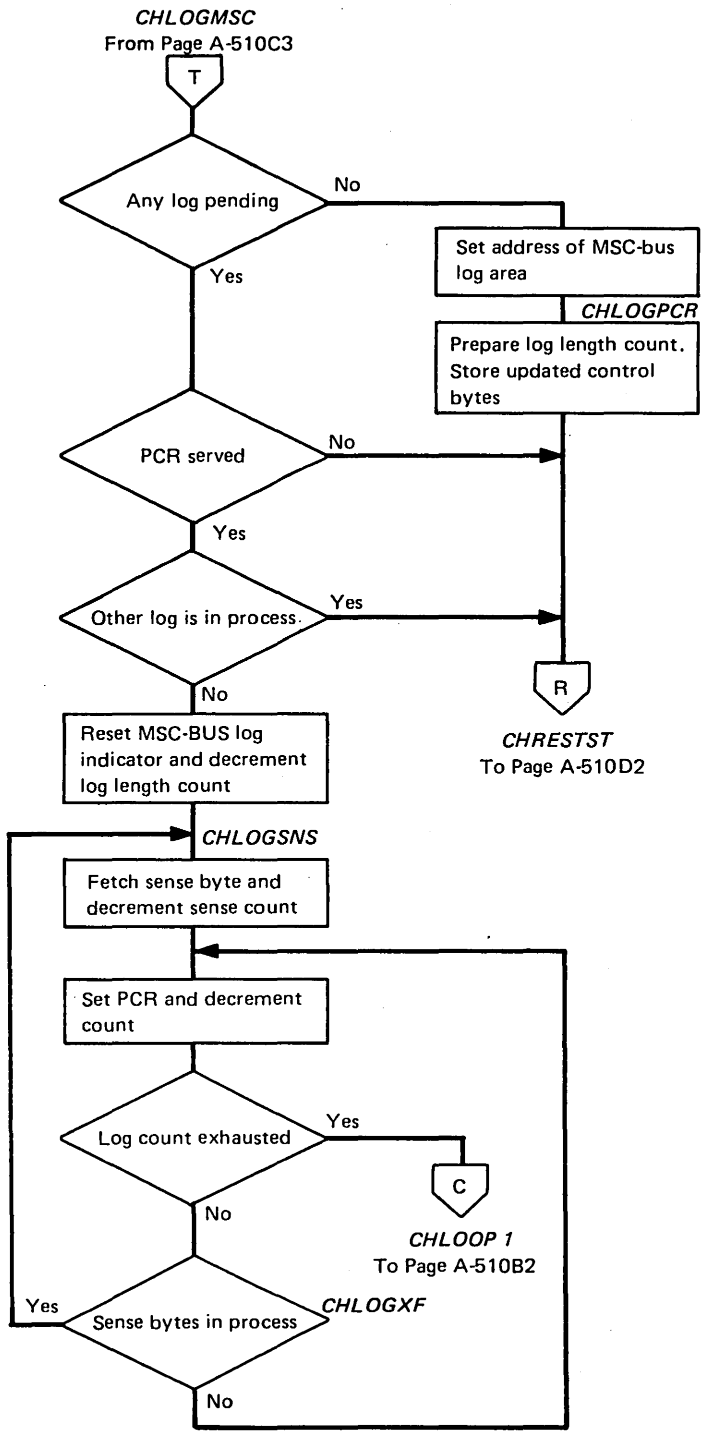
Note: Changing of IARs is effected by a "suffix U" type of instruction (see Page 3-270)

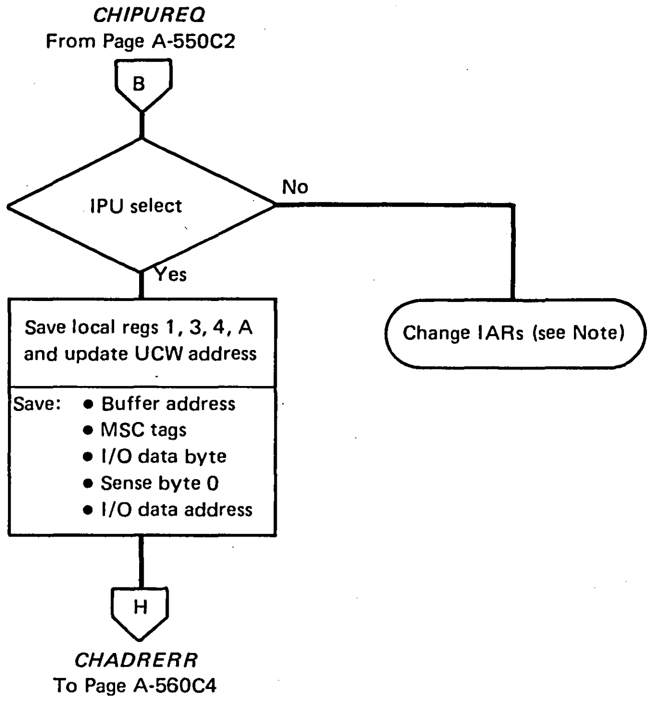
This page shows the continuation of the 'MSC Error Handling' Routine and gives the List of Operation Identifiers

Common IOP Channel Program (continued)









Label List

Label	Page No.	Coordinate
CHADRERR	A-560	C4
CHBUSY	A-510	E5
CHCC 0 + 1	A-540	C9
CHCCTEST	A-530	B2
CHCCEADD	A-520	A3
CHCCRTN	A-530	C2
CHCE	A-540	B6
CHCEDE1	A-530	A5
CHCEDE2	A-530	A5
CHCEDEOP	A-530	A4
CHCEOP	A-530	A2
CHCHAINC	A-510	A7
CHCHAIND	A-510	A7
CHCHANCH	A-530	B5
CHCMDDEC	A-510	A4
CHC'DREJ	A-530	A6
CHCSWADD	A-540	C6
CHDE	A-530	A7
CHDEVDEC	A-520	C6
CHECSW	A-560	C2
CHEXCEPT	A-550	A3
CHEXTDAM	A-560	C6
CHHALT	A-520	A7
CHHANDLE	A-560	A2
CHIDA1	A-520	B3
CHIDAXF	A-520	C3
CHINICH	A-530	A6
CHINTREQ	A-530	C8
CHINTSEL	A-540	A3
CHIPUERR	A-510	B5
CHIPUREQ	A-570	A6
CHIPUTST	A-510	B3
CHLCL	A-540	A8
CHLOGMSC	A-570	A4
CHLOGPCR	A-570	B5
CHLOGSNS	A-570	C4
CHLOGXF	A-570	E4
CHLOOP	A-510	B3
CHLOOP1	A-510	B3
CHLOOP2	A-510	B3
CHMETER	A-530	C6
CHMSCERR	A-550	A7
CHMSCLOG	A-550	B7
CHNOOP	A-520	C6
CHPAGING	A-550	C3
CHPCIREQ	A-570	C2
CHPRINT	A-540	C3
CHPROGCH	A-510	D9
CHRDCCW1	A-510	B6
CHRDINT	A-540	B2
CHRELOC	A-520	C3
CHRESTST	A-510	D3
CHRETRY1	A-550	C7
CHRPINT	A-540	B2
CHSENSE	A-520	A8
CHSLERR	A-520	D8
CHSTATUS	A-530	C5
CHSVPREQ	A-570	A2
CHSYSRES	A-510	A2
CHTIC	A-510	C9
CHTIO	A-510	C6
CHTIOINT	A-510	E6
CHWAIT (0 to F)	A-550	A4
CHWAITX	A-550	A3
CHXFADDR	A-520	B2
CHSVP	A-510	B2
RESTART	A-510	A3

Note: Changing of IARs is effected by a "Suffix U" type of instruction (see Page 3-270)

This page shows the following:
Miscellaneous Subroutines; Label List.

Internal Register Assignments — Common Channel Program (DLS)

- This table shows the parts of DLS that are used as internal work registers in the common IOP channel program.
- When an internal register is defined by *CHINT and its address* it means that the register is used for different purposes. Its individual use is shown in the microprogram list together with all the routines that are using that register.
- If an internal register contains "typical" information, a designation pointing to that typical information is used. The address of the register remains unchanged.

Register Address	Register Designation	Bit Positions 0	1	2	3	4	5	6	7
00 to 0F	CHINT 00 to 0F	These labels define DLS as internal registers for different purposes. Their usage is shown with the different microroutines.							
00	CHYADDR	UCW Byte Address							
02	CHYCHD	Command (First byte of CCW)							
03	CHYDATA 2	Data Address (third byte of CCW)							
04	CHYDATA 3	Data Address (fourth byte of CCW)							
05	CHYSENSE	Sense Byte 0						(Not used)	
		CMD Reject	Intervention Required	Bus Out Check	Equipment Check	Data Check	Overrun		
	CHYDATA 1	(Not used)						Data Address	
								6	7
06	CHYFLAG	Flags (fifth byte of CCW)							
		Chain Data	Chain Command	SLI	Skip	PCI	IDA	Zero	Zero
07	CHYOPSTA	(Not used)					Status of Operation (see also page 1-093)		
							5	6	7
	CHYSEQ		Sequence Code (see also page 5-050)			(Not used)			
			1	2	3				
	CHYTIC	TIC Allow Indicator 0							
08	CHY KEY	Key (first byte of CAW)							
		0	1	2	3	Zero	Zero	Zero	Zero
09	CHYCCW 1	Address of Next CCW							
0A	CHYCCW 2	Address of Next CCW							
0B	CHYCCW 3	Address of Next CCW							
0C	CHY UNITS	Unit Status (fifth byte of CSW; see also pages 3-010 and 5-020)							
0D	CHY CHANS	Channel Status (sixth byte of CSW; see also pages 3-010 and 5-020)							
0E	CHY CNT 1	Byte Count (seventh byte of CCW)							
0E	CHY CNT 2	Byte Count (eighth byte of CCW)							

Index

3125 Processing Unit
Input/Output Processor

Order No. SY33-1063-1

READER'S COMMENT FORM

Your views about this publication may help improve its usefulness; this form will be sent to the author's department for appropriate action. Using this form to request system assistance or additional publications will delay response, however. *For more direct handling of such requests, please contact your IBM representative or the IBM Branch Office serving your locality.*

Possible topics for comment are:

Clarity Accuracy Completeness Organization Index Figures Examples Legibility

What is your occupation? _

Number of latest Technical Newsletter (if any) concerning this publication: _ _ _ _ _ _ _ _ _ _ _

Please indicate in the space below if you wish a reply.

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments.)

Cut or Fold Along Line

3125 Processing Unit
Input/Output Processor

Order No. SY33-1063-1

READER'S COMMENT FORM

Your views about this publication may help improve its usefulness; this form will be sent to the author's department for appropriate action. Using this form to request system assistance or additional publications will delay response, however. *For more direct handling of such requests, please contact your IBM representative or the IBM Branch Office serving your locality.*

Possible topics for comment are:

Clarity Accuracy Completeness Organization Index Figures Examples Legibility

What is your occupation? _

Number of latest Technical Newsletter (if any) concerning this publication: _ _ _ _ _ _ _ _ _ _ _

Please indicate in the space below if you wish a reply.

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments.)

SY33-1063-1

Your comments, please . . .

This manual is part of a library that serves as a reference source for customer engineers. Your comments on the other side of this form will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Fold Fold

First Class
Permit 40
Armonk
New York

Business Reply Mail

No postage stamp necessary if mailed in the U.S.A.

Postage will be paid by:

International Business Machines Corporation
Department 813B
1133 Westchester Avenue
White Plains, New York 10604

Fold Fold

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

SY33-1063-1

Your comments, please . . .

This manual is part of a library that serves as a reference source for customer engineers. Your comments on the other side of this form will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Fold Fold

First Class
Permit 40
Armonk
New York

Business Reply Mail

No postage stamp necessary if mailed in the U.S.A.

Postage will be paid by:

International Business Machines Corporation
Department 813B
1133 Westchester Avenue
White Plains, New York 10604

Fold Fold

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)