IBM Field Engineering

Diagram Manual

System/360 Model 40

2040 Processing Unit

SY22-2842-3

PREFACE

This manual contains diagrams for use with System/360 Model 40 2040 Processing Unit, Field Engineering Maintenance Manual, Order No. SY22-2841, and also with the following manuals:

System/360 Model 40 Comprehensive Introduction, Field Engineering Theory of Operation Manual, Order No. SY22-2840.

System/360 Model 40 Functional Units, Field Engineering Manual of Instruction, Order No. SY22-2843.

System/360 Model 40 Theory of Operation, Field Engineering Theory of Operation Manual, Order No. SY22-2844.

System/360 Model 40 Power Supplies, Features, and Appendix, Field Engineering Manual of Instruction, Order No. S223-2845.

Power Supplies, SLT, SLD, ASLT, MST, Field Engineering Theory of Operation Manual, Order No. SY22-2799. This manual may be used for maintenance or instruction purposes. It contains Data Flow Charts, Simplified Logic Diagrams (SLD), Condensed Logic Flow Charts (CLF), Malfunction Analysis Procedures (MAP), and 1401/1460 Emulator Flow Charts.

The EC level of Control Automation System (CAS) Logic Diagrams referenced within this manual is 255263. ALD references are at EC level 254814 for all diagrams except the 1401/1460 emulator flow charts, which are at EC level 255264. The Mid-Pac power supply is at EC level 255055 and the 2.5 kHz HF Power Supply is at EC level 266316. Subsequent engineering changes may alter the contents of this manual.

Fifth Edition (January 1970)
This manual, Order Number SY22-2842-3, is a reprint of Y22-2842-2 incorporating changes released in FE Supplement Y22-6809, November 28, 1969.

Changes are continually made to the specifications herein; any such changes will be reported in subsequent revisions or FE Supplements.

This manual has been prepared by the IBM Systems Development Division, Product Publications, Dept B96, PO Box 390, Poughkeepsie, N.Y. 12602. A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be sent to the above address.

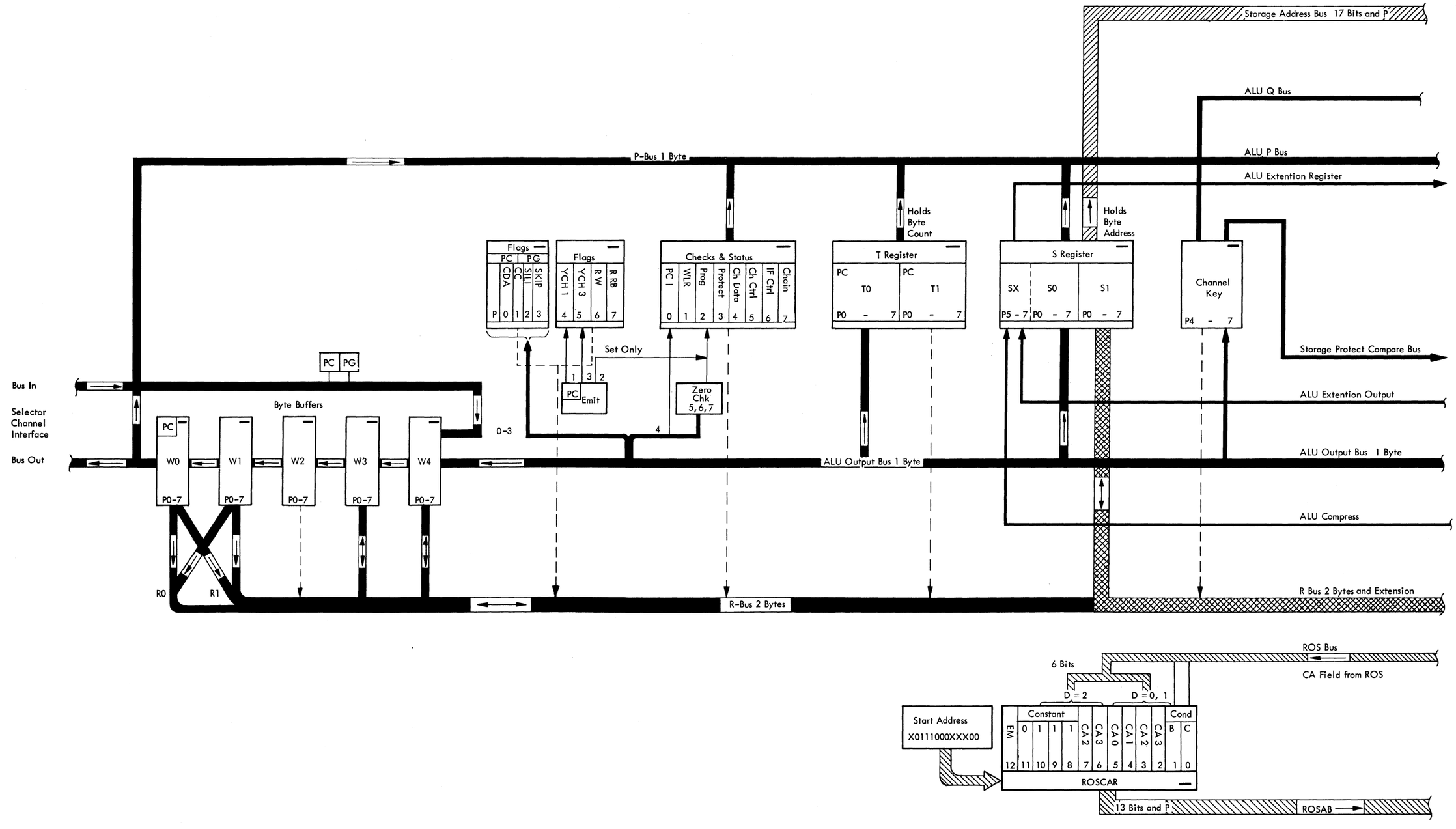


FIGURE 011. SELECTOR CHANNEL DATA FLOW

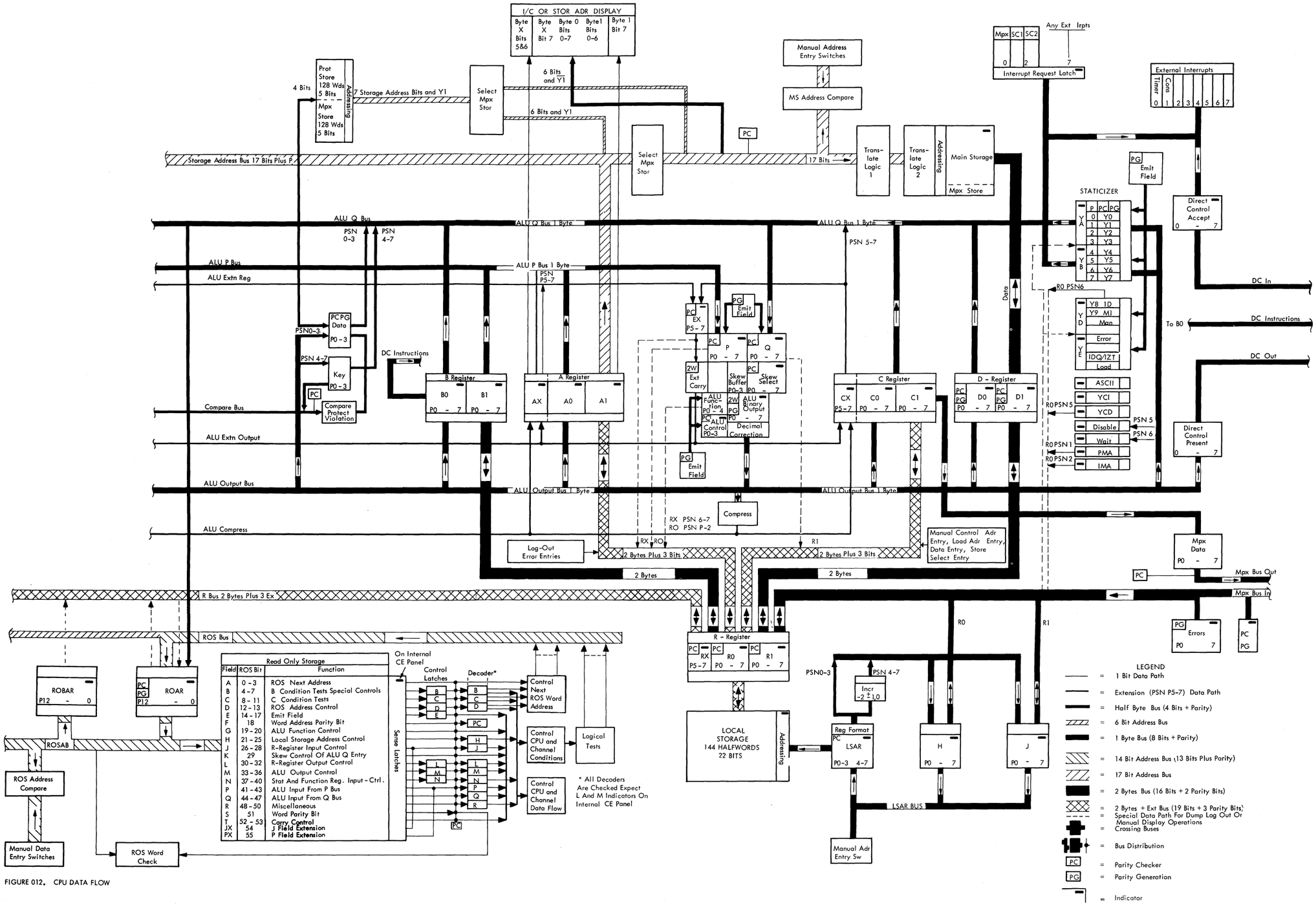


FIGURE 012. CPU DATA FLOW

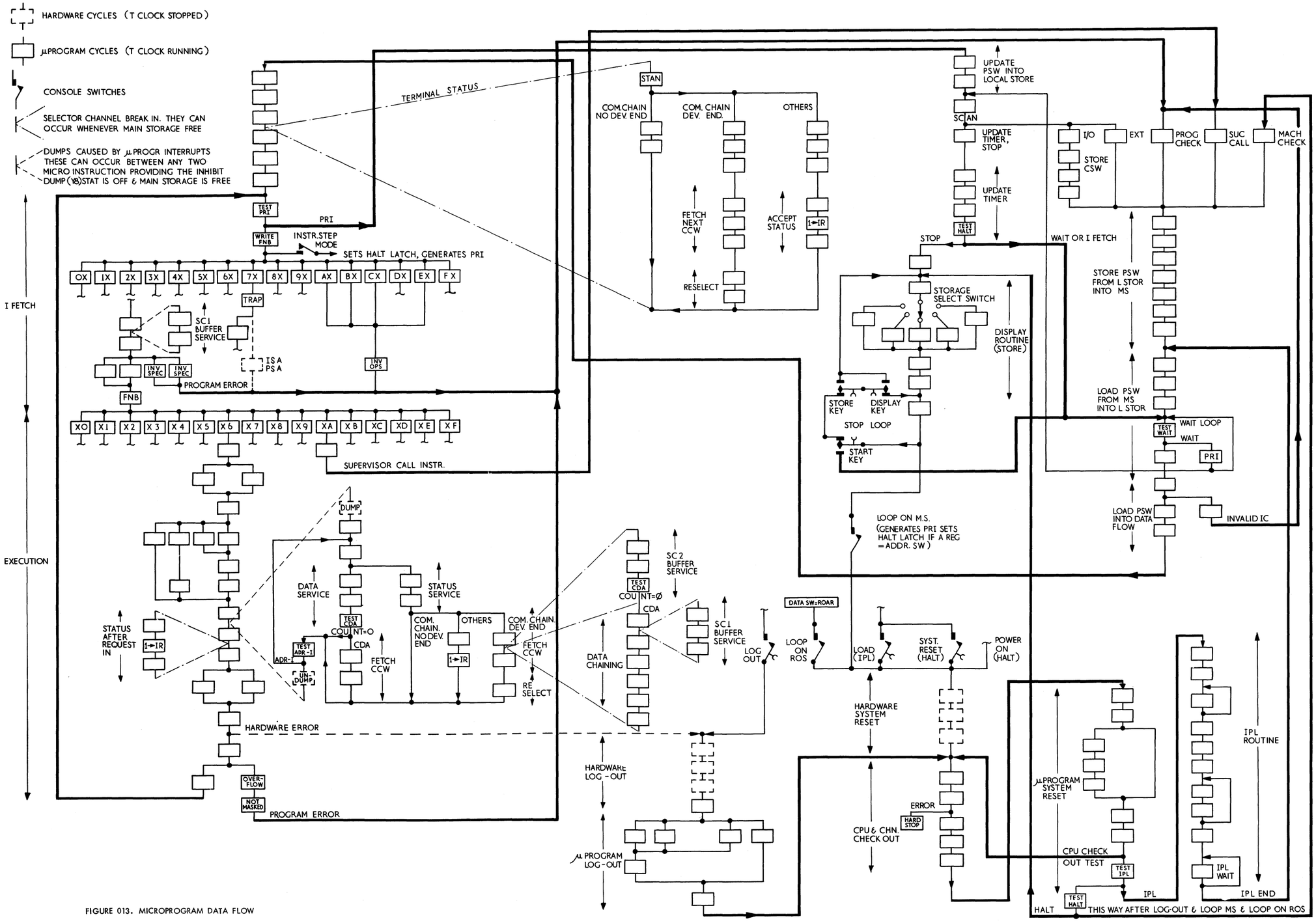


FIGURE 013. MICROPROGRAM DATA FLOW

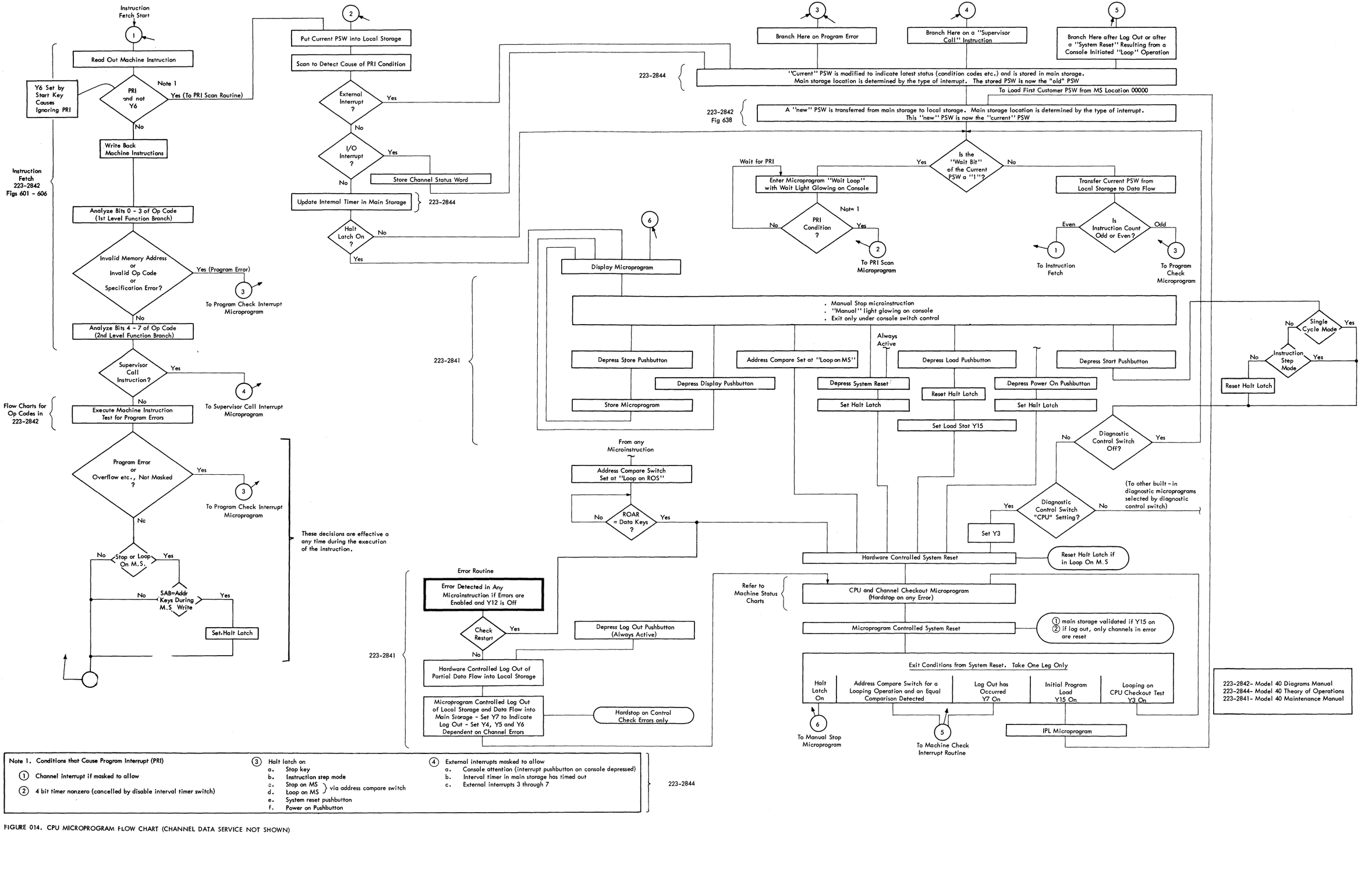
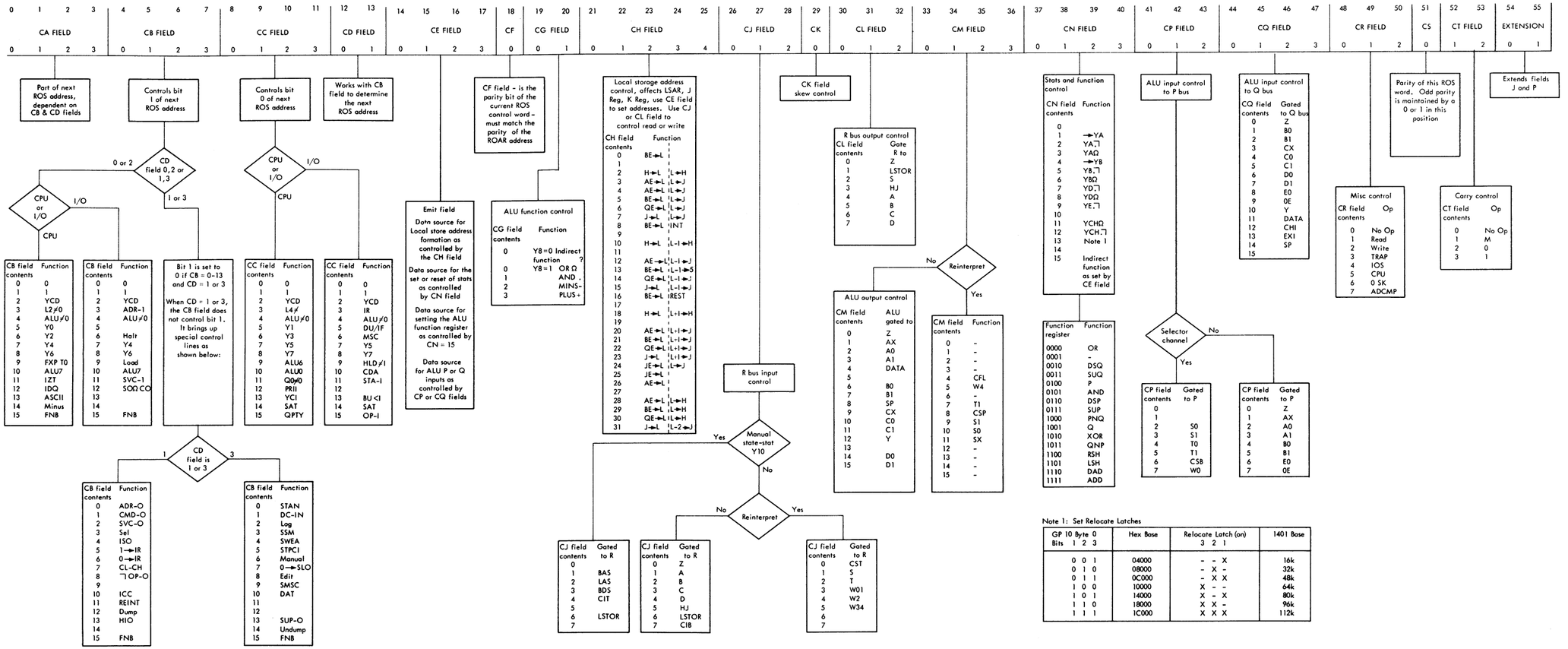


FIGURE 014. CPU MICROPROGRAM FLOW CHART (CHANNEL DATA SERVICE NOT SHOWN)



●FIGURE 015. ROS CONTROL WORD

Note 1: Set Relocate Latches

GP 10 Byte 0 Bits 1 2 3	Hex Base	Relocate Latch (on) 3 2 1	1401 Base
0 0 1	04000	- - X	16k
0 1 0	08000	- X -	32k
0 1 1	0C000	- X X	48k
1 0 0	10000	X - -	64k
1 0 1	14000	X - X	80k
1 1 0	18000	X X -	96k
1 1 1	1C000	X X X	112k

S1Y22-2842-3
FSS: S1Y22-0827

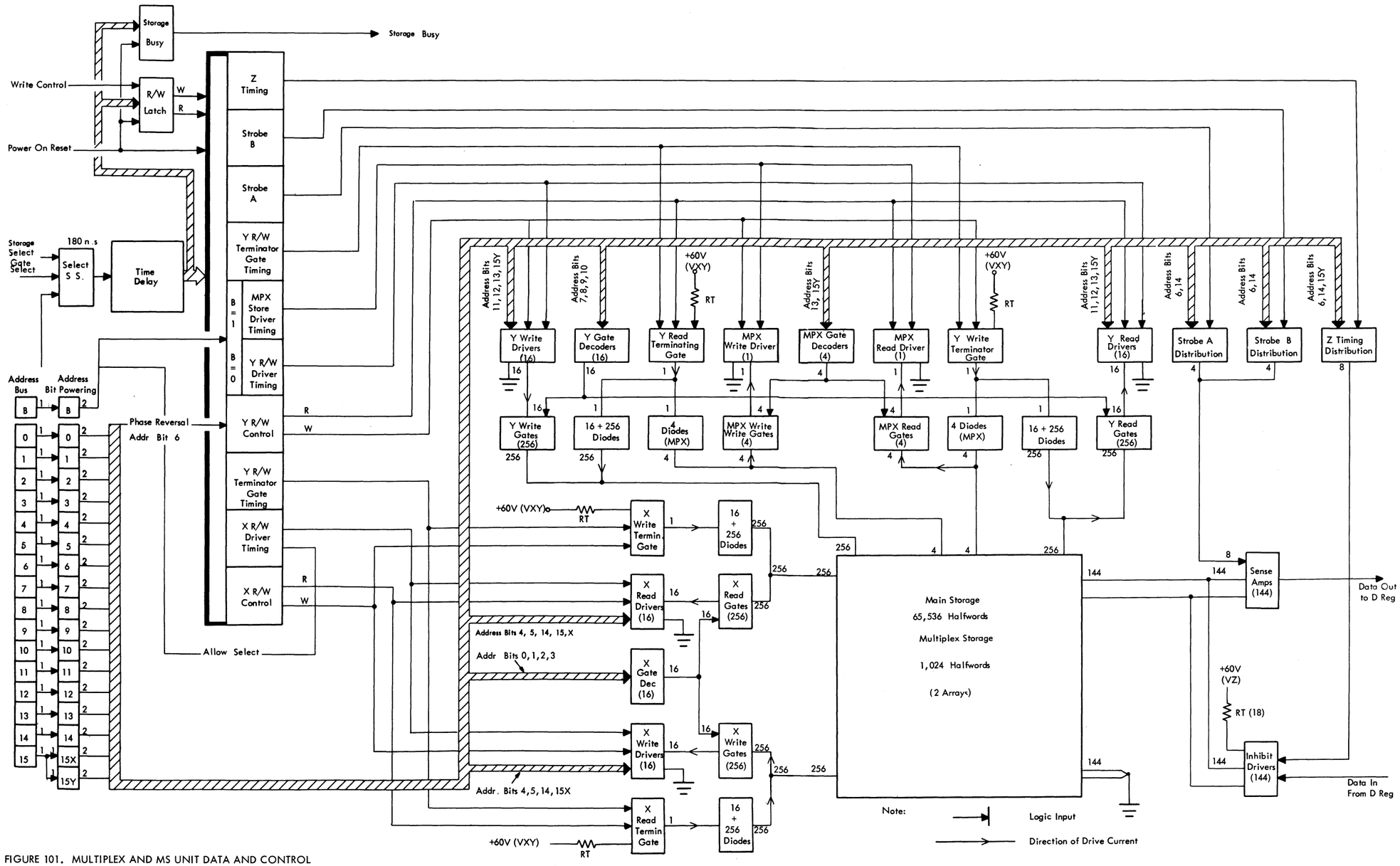


FIGURE 101. MULTIPLEX AND MS UNIT DATA AND CONTROL

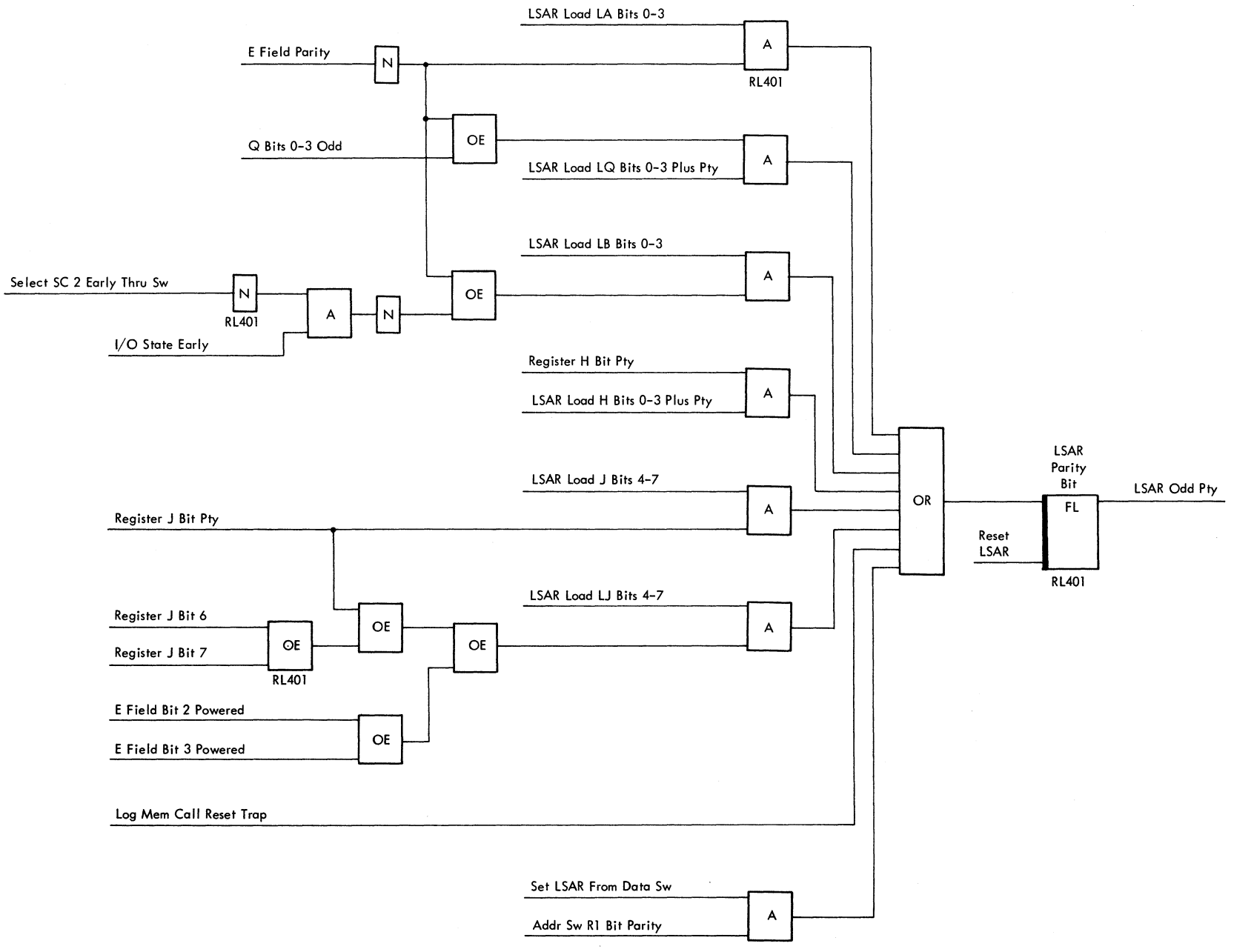


FIGURE 501. LSAR PARITY GENERATION

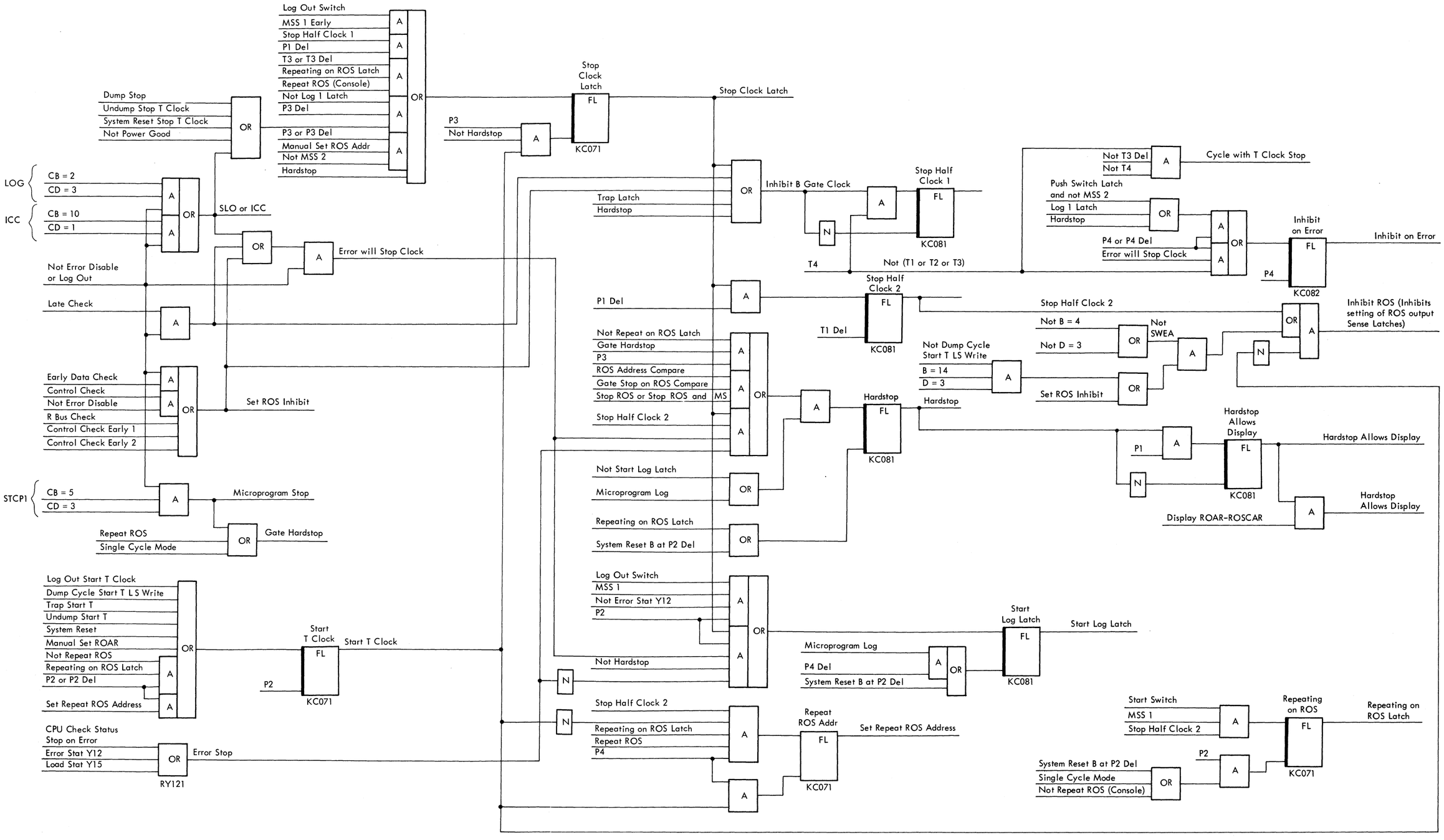
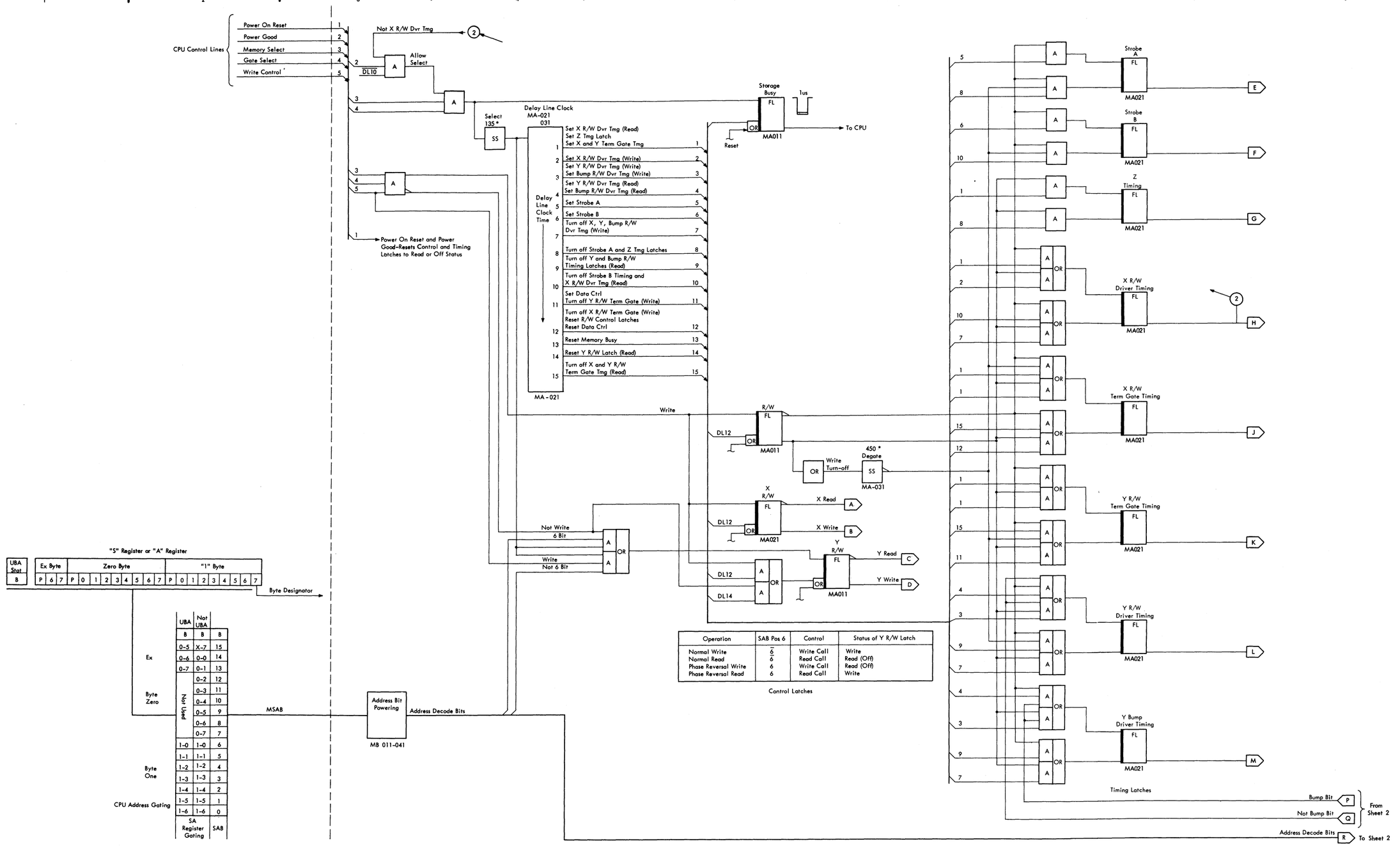


FIGURE 502. CLOCK CONTROL (KC071-KC081)



Operation	SAB Pos 6	Control	Status of Y R/W Latch
Normal Write	$\overline{6}$	Write Call	Write
Normal Read	$\overline{6}$	Read Call	Read (Off)
Phase Reversal Write	6	Write Call	Read (Off)
Phase Reversal Read	6	Read Call	Write

Control Latches

FIGURE 504. MAIN STORAGE CONTROL AND TIMING CIRCUITS (SHEET 1 of 2)

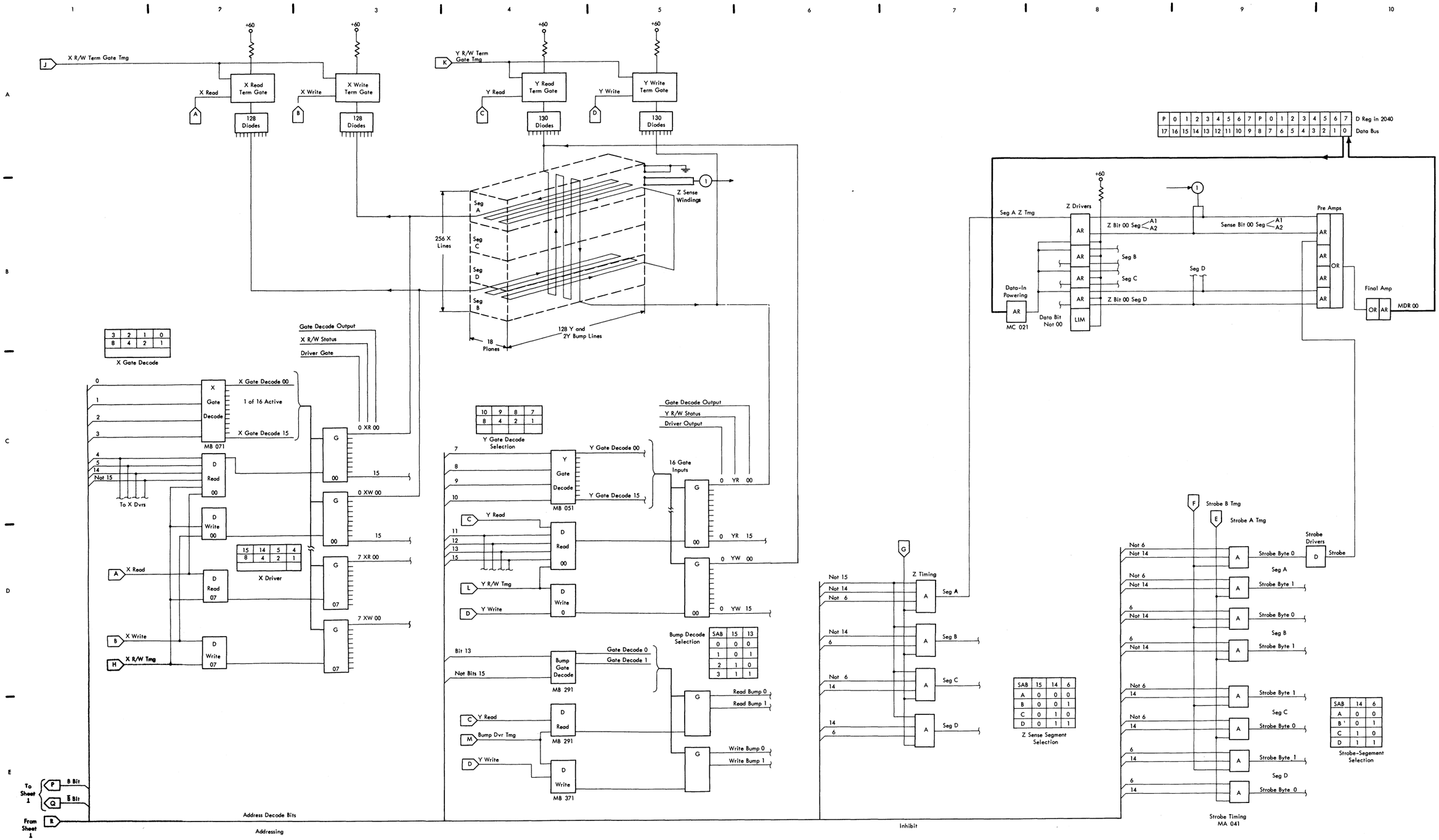
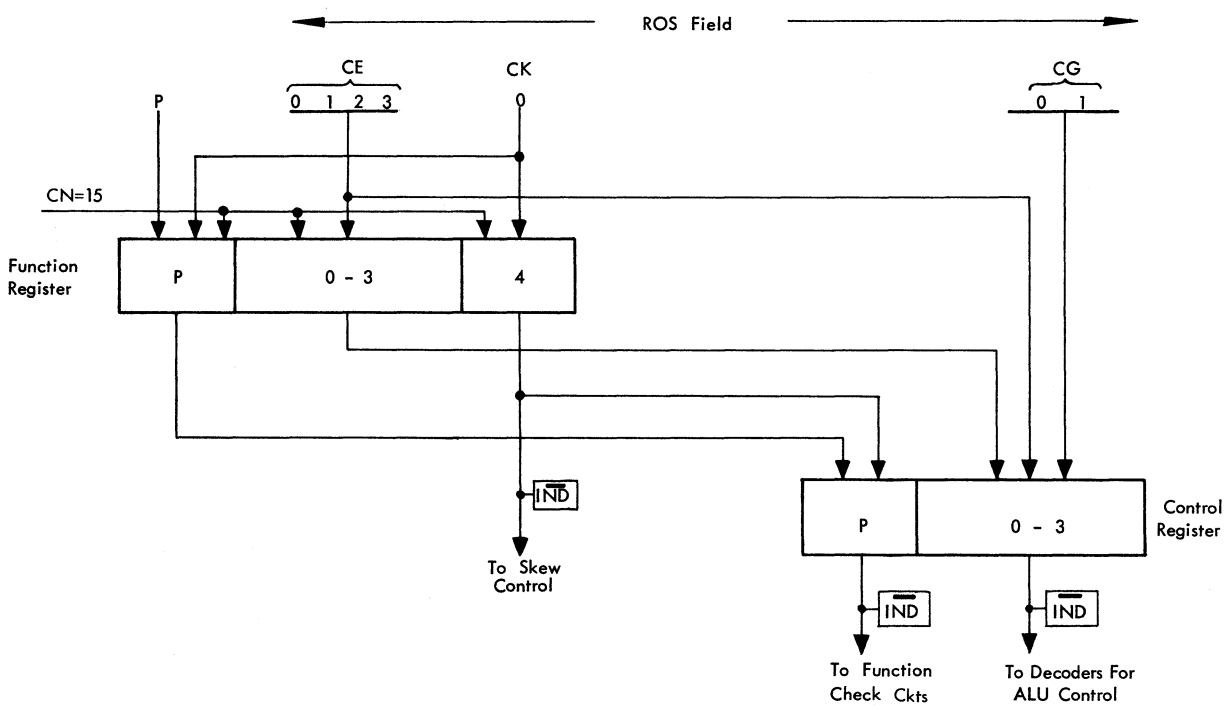
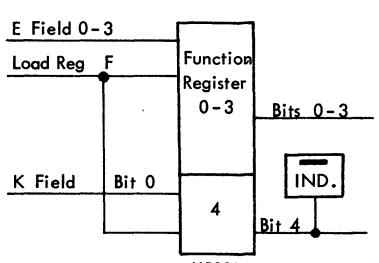
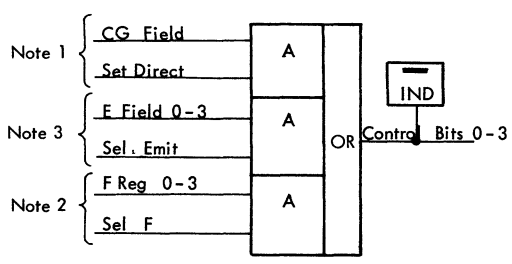


FIGURE 504. MAIN STORAGE CONTROL AND TIMING CIRCUITS (SHEET 2 OF 2)







KP001

Set By CN = 15 At T4 Del
Reset By Reloading G

Control Register

Control Latches Reset At T4 Del Special
If Not Inhibit ALU Control

	CG Field Bits 0	1
P Or Q	0	0
P And Q	0	1
P Minus Q	1	0
P Plus Q	1	1

	Control Bits 0	1	2	3
0	0	0	0	0
5	0	1	0	1
3	0	0	1	1
15	1	1	1	1

Note 1 Set Direct If CG = 0 (& Y8 =1)
Or CG = 1 - 3

Boolean Expression

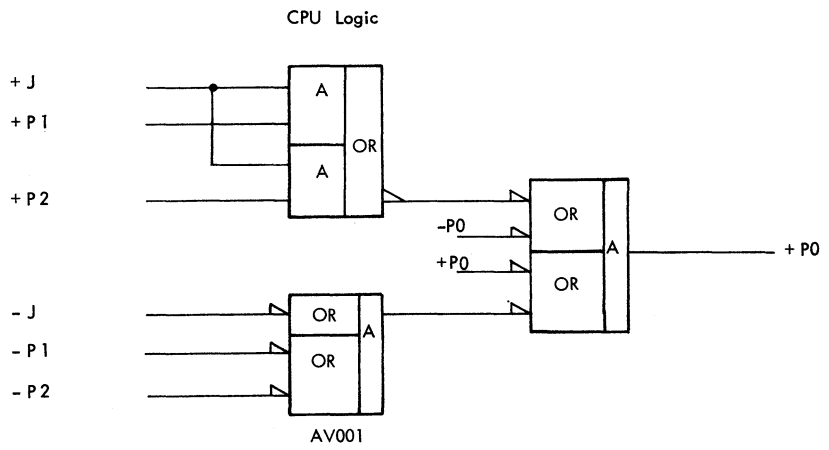
Position 0 G1 and G0
.. 1 G1
.. 2 G0
.. 3 G1 or G0
} See Chart Opposite

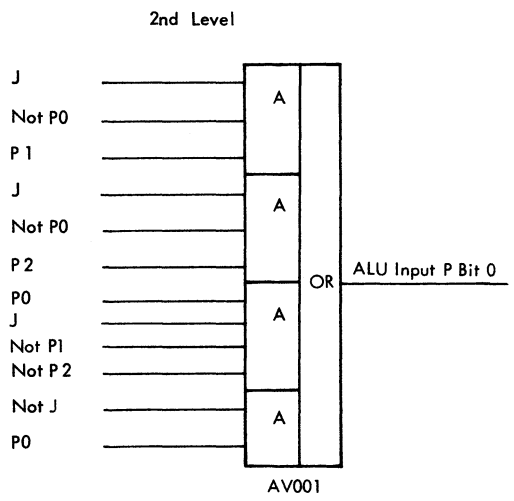
Note 2 Select F Register If CG = 0 (and Y8=0)
And CN ≠ 15

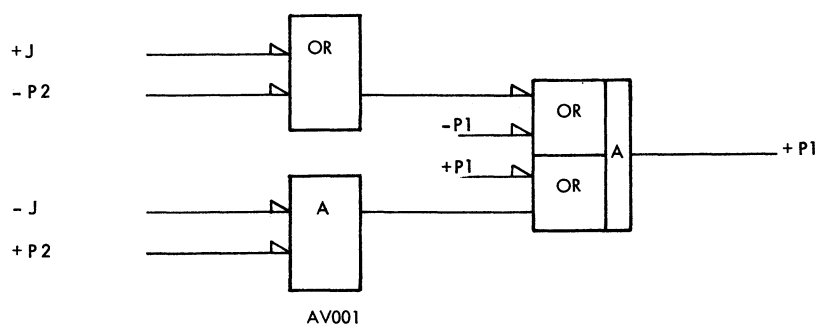
Note 3 Select Emit If CG = 0 (and Y8=0)
And CN = 15

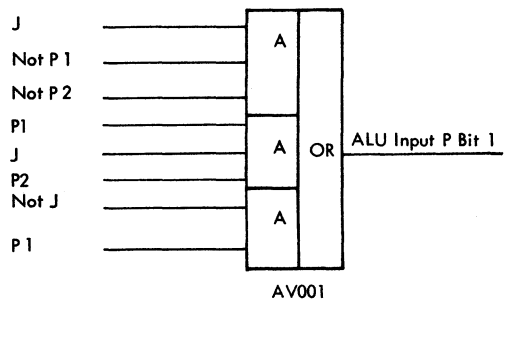
To Change Function Register & Use In Same Cycle,
Both Control And Function Resisters Set By Emit Field At T4 Delay.
Function Register Not Gated To Control Register Until After T4 Delay
During Normal Operation.
Calling Select Emit Avoids Waiting For Function Register To Be Set
And Then Setting The Control Register.

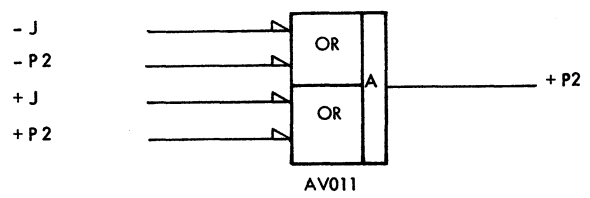
FIGURE 505. FUNCTION & CONTROL REGISTERS

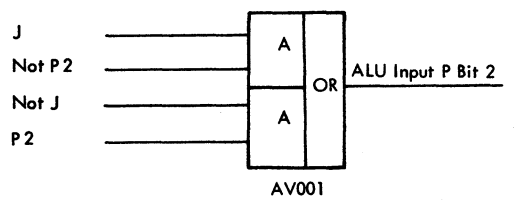










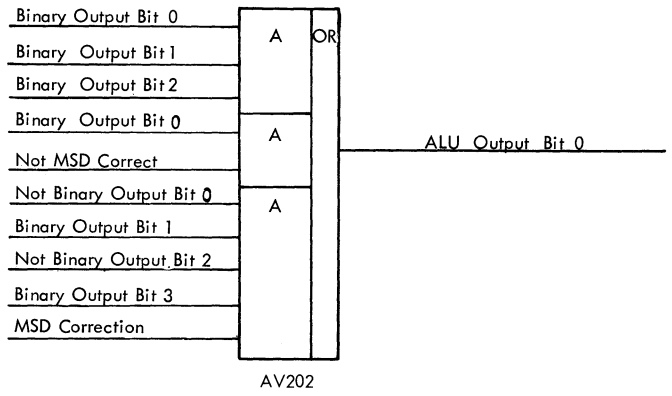


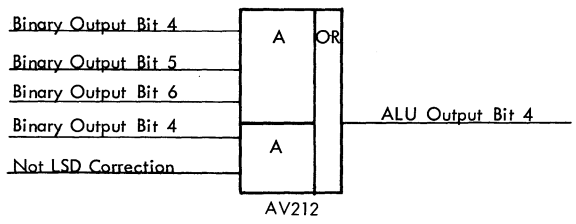
Boolean Expression

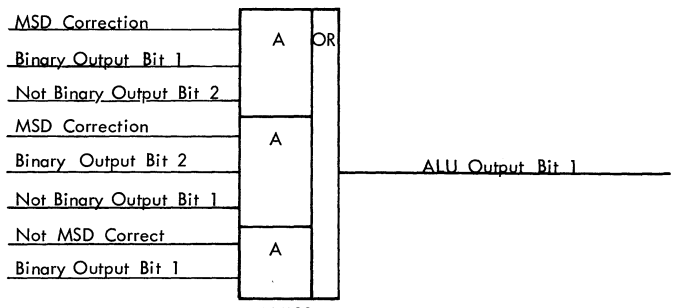
	Decimal Add Operation	Normal Operation
Bit Position 0	$J\overline{P0}P1$ or $J\overline{P0}P2$ or $JP0\overline{P1}\,\overline{P2}$	or $\overline{J}P0$
Bit Position 1	$J\overline{P1}\,\overline{P2}$ or $JP1\,P2$	or $\overline{J}P1$
Bit Position 2 (Inversion)	$J\overline{P2}$	or $\overline{J}P2$

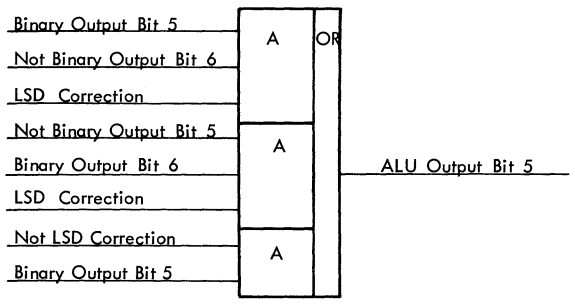
Note: Bit Position 3 Unchanged
Bits 4-7 Not Shown As
Logic Is Same As For
Bits 0-3

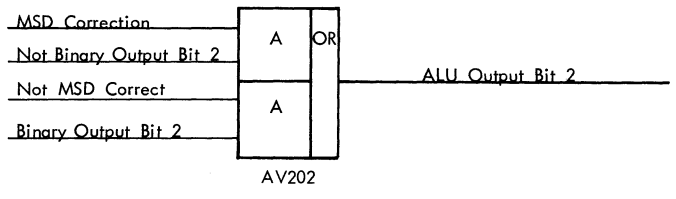
FIGURE 506. DECIMAL FILLER

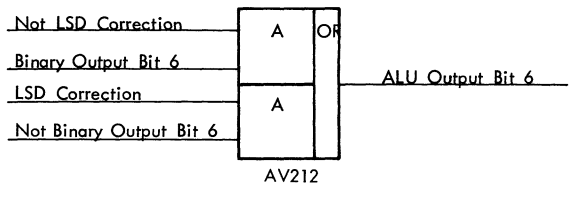


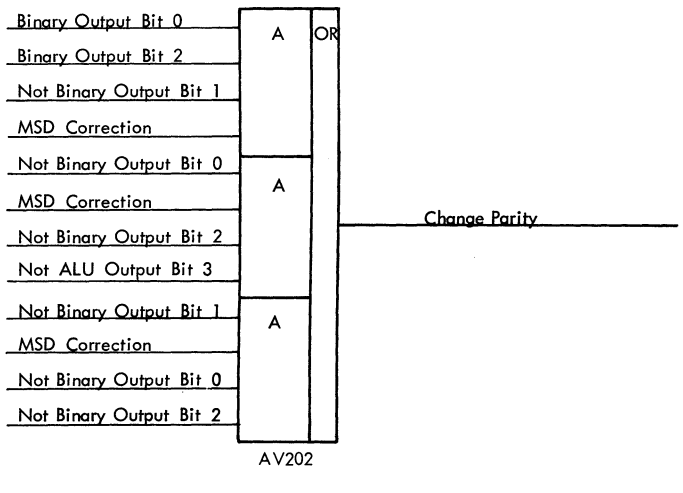


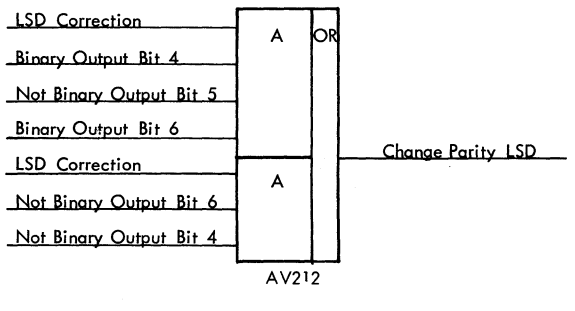


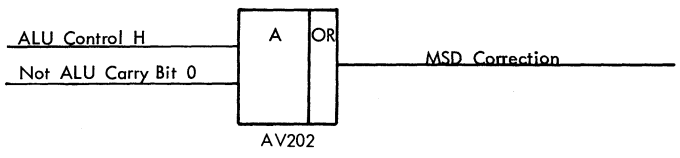






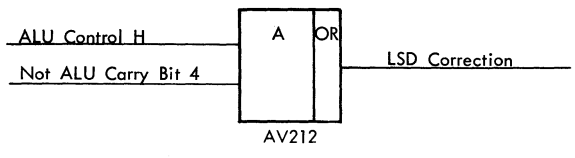






Boolean Expression MSD

Bit 0	$0.1.2 + \overline{0}.1.\overline{2}.3$
Bit 1	$1.\overline{2} + \overline{1}.2$
Bit 2	Invert
Bit 3	No Change (Not Shown)



Boolean Expression LSD

Bit 4	$4.5.6$
Bit 5	$5.\overline{6} + \overline{5}.6$
Bit 6	Invert
Bit 7	No Change (Not Shown)

FIGURE 507 DECIMAL CORRECTION

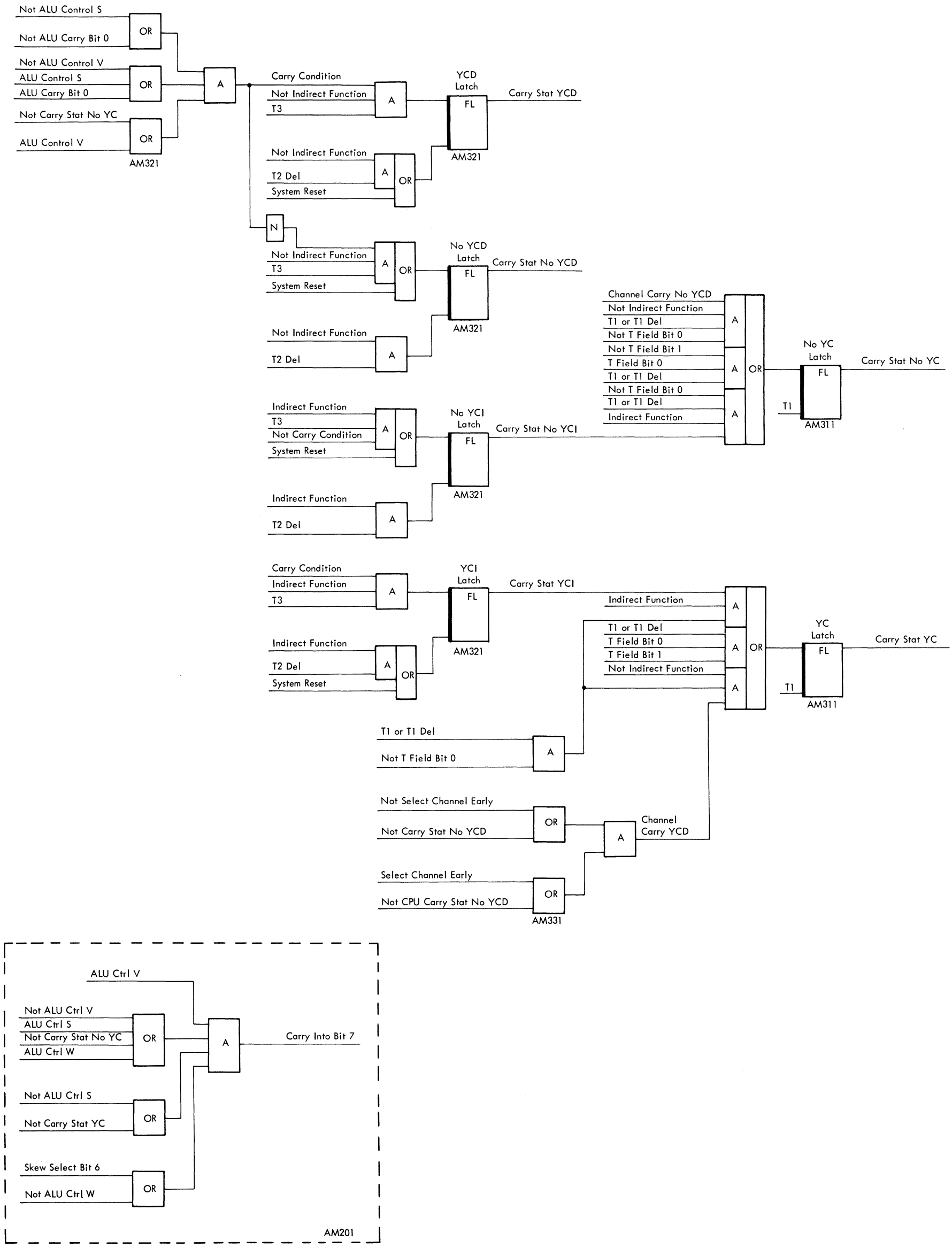


FIGURE 508. CARRY LATCHES

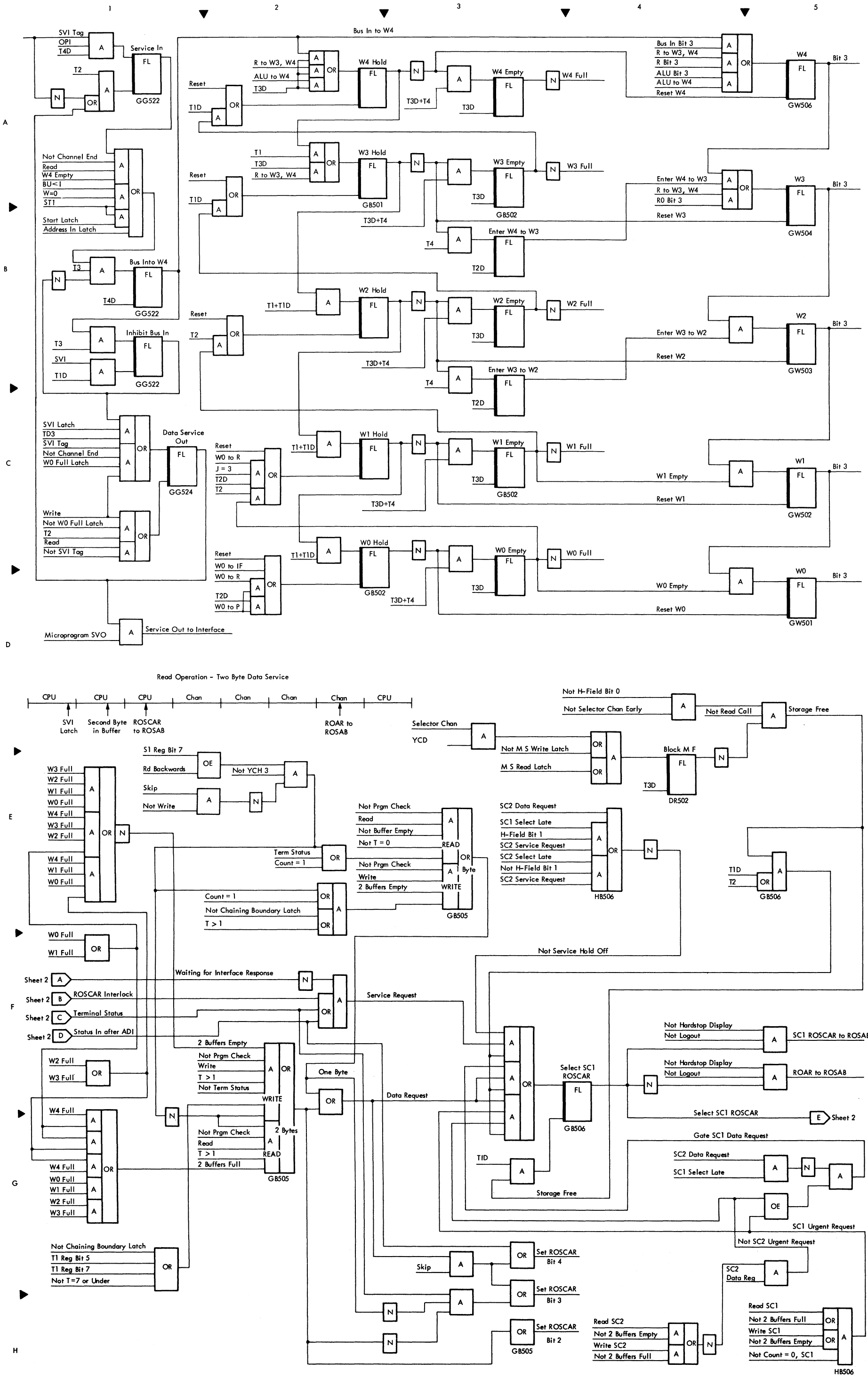


FIGURE 509. SELECTOR CHANNEL CONTROLS (SHEET 1 OF 2)

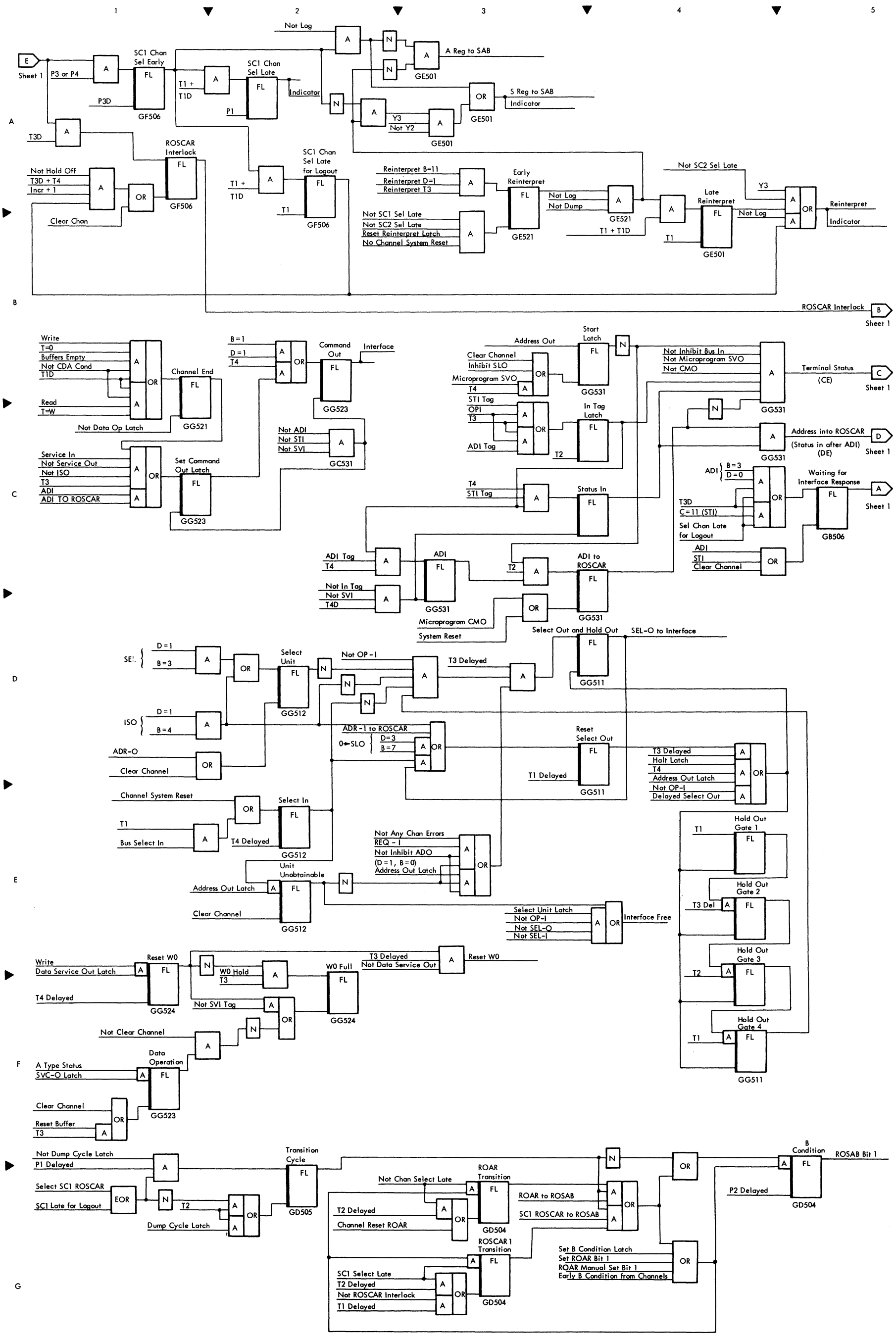


FIGURE 509. SELECTOR CHANNEL CONTROLS (SHEET 2 OF 2)

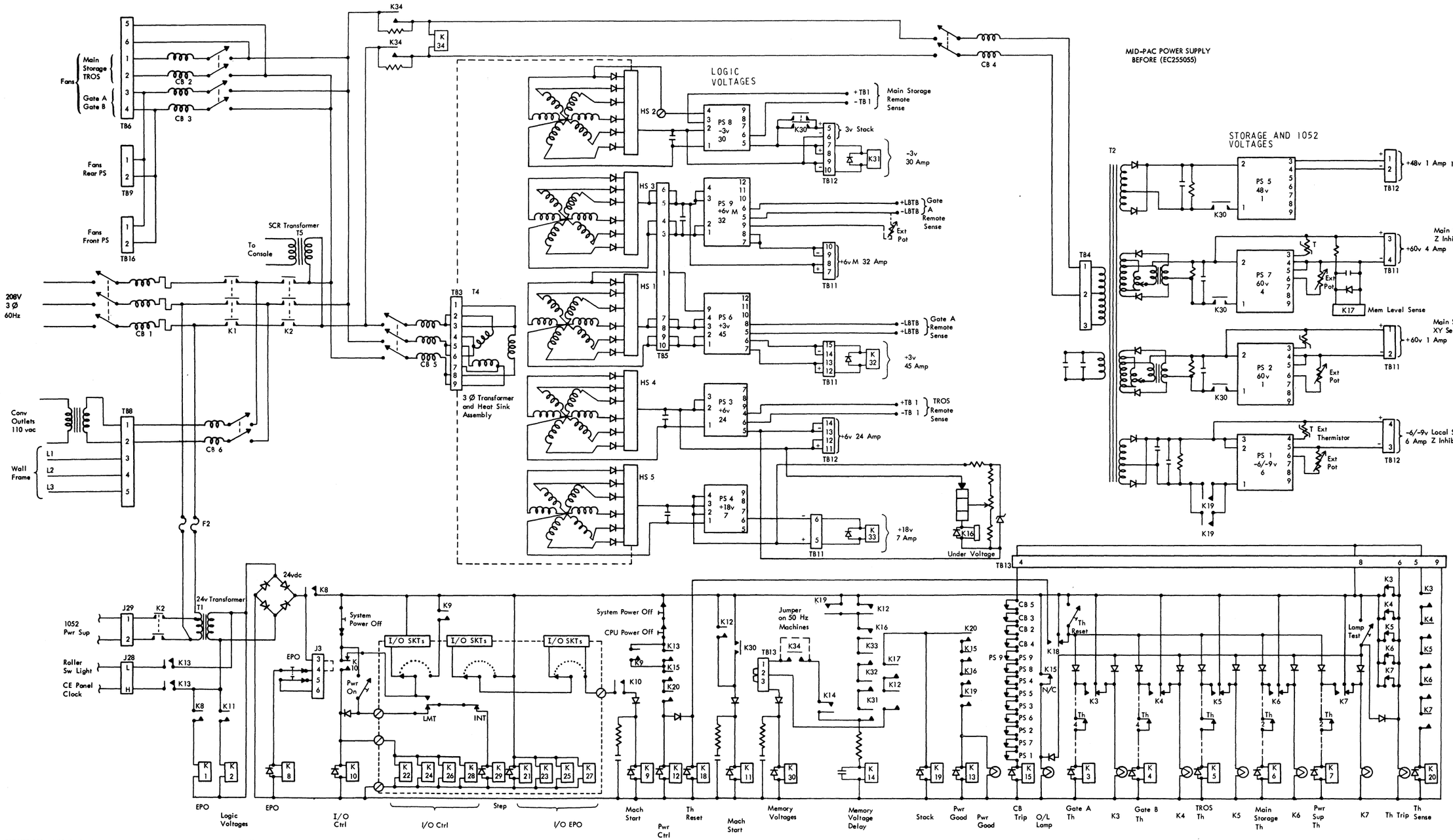
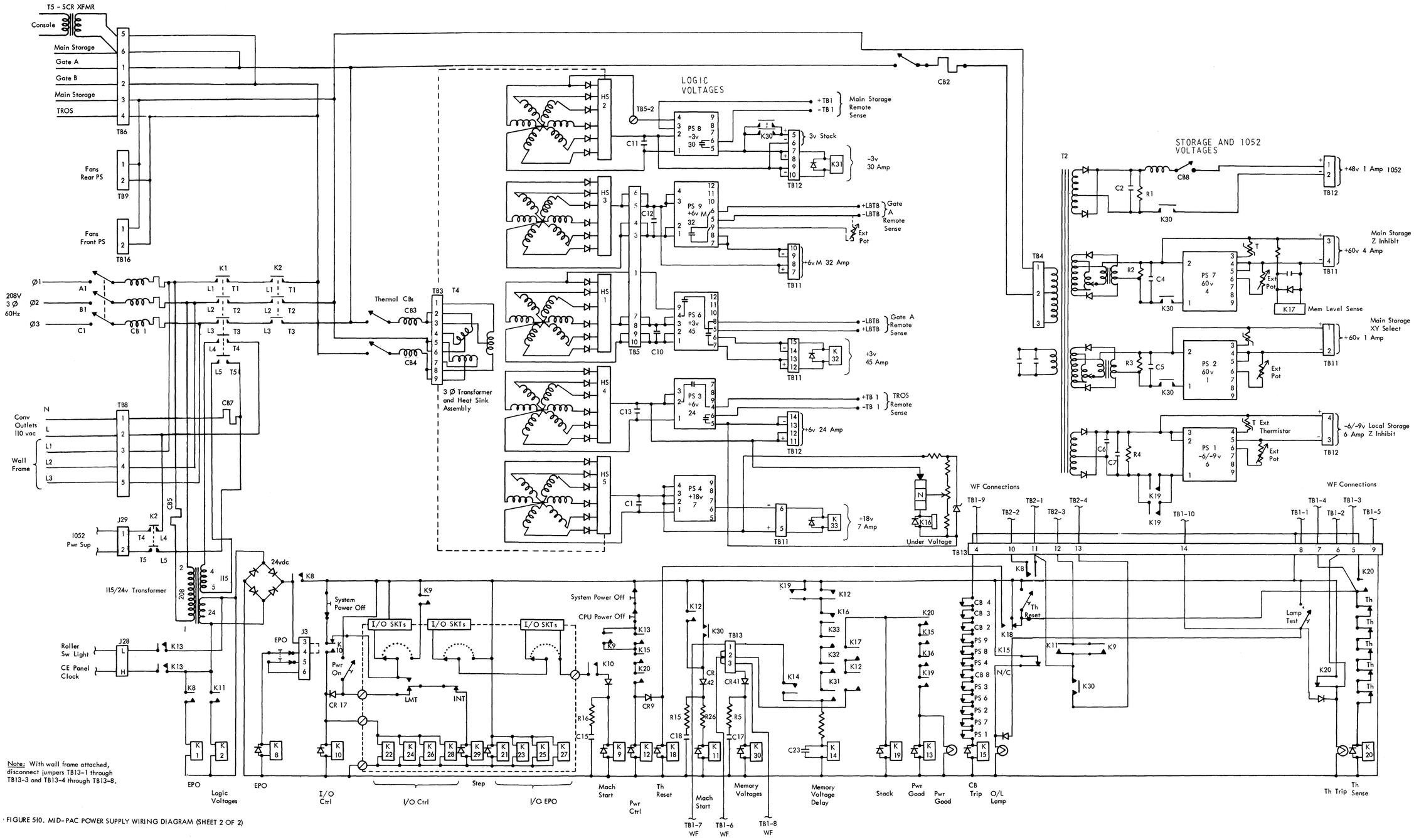
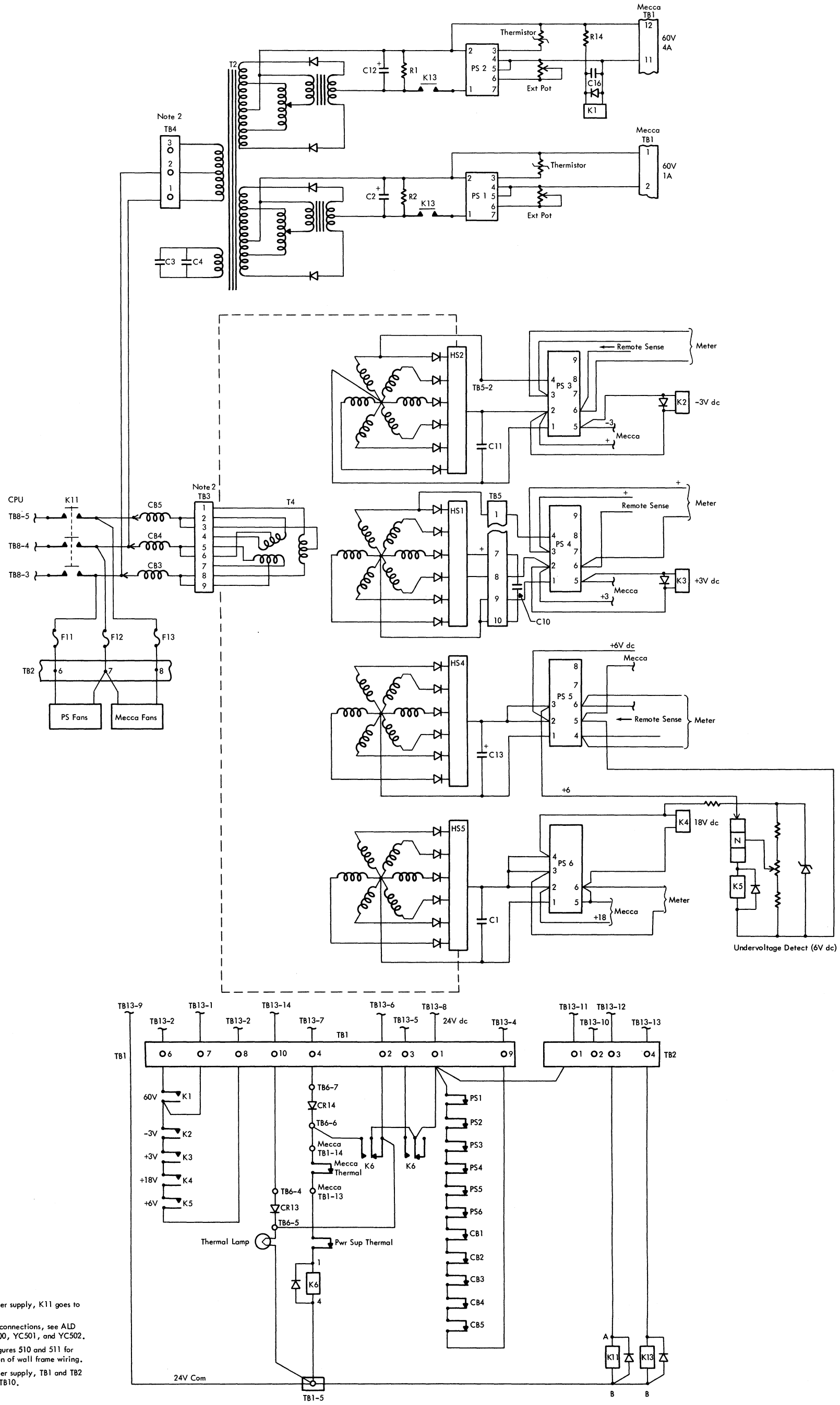


FIGURE 510. MID-PAC POWER SUPPLY WIRING DIAGRAM (SHEET 1 OF 2)



Note: With wall frame attached, disconnect jumpers TB13-1 through TB13-3 and TB13-4 through TB13-8.

FIGURE 510. MID-PAC POWER SUPPLY WIRING DIAGRAM (SHEET 2 OF 2)



Notes:

1. On HF power supply, K11 goes to CPU TB11.
2. For 50-Hz connections, see ALD pages YC500, YC501, and YC502.
3. Refer to Figures 510 and 511 for continuation of wall frame wiring.
4. On HF power supply, TB1 and TB2 go to CPU TB10.

FIGURE 510A. 2040 MID-PAC WALL FRAME WIRING DIAGRAM

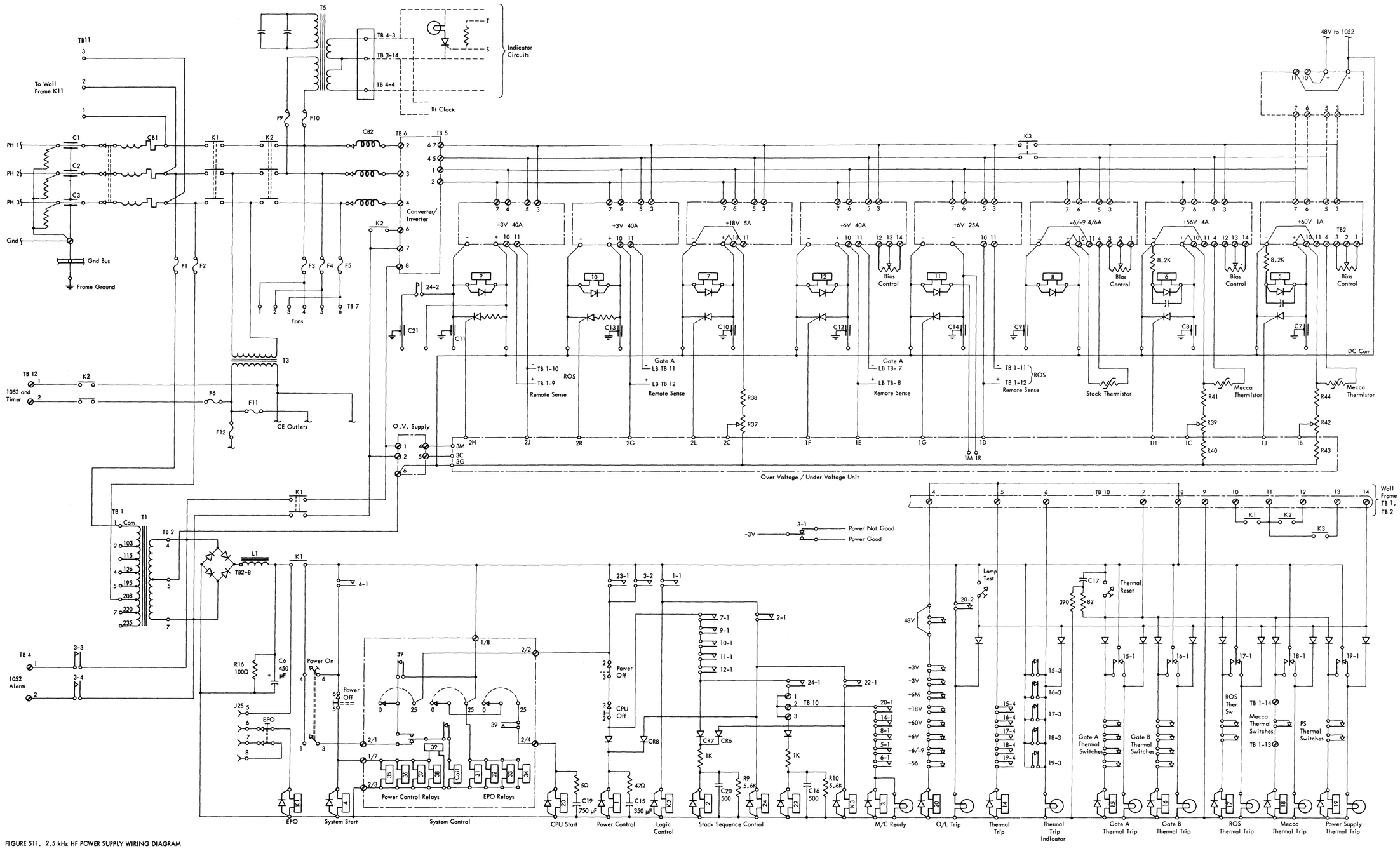
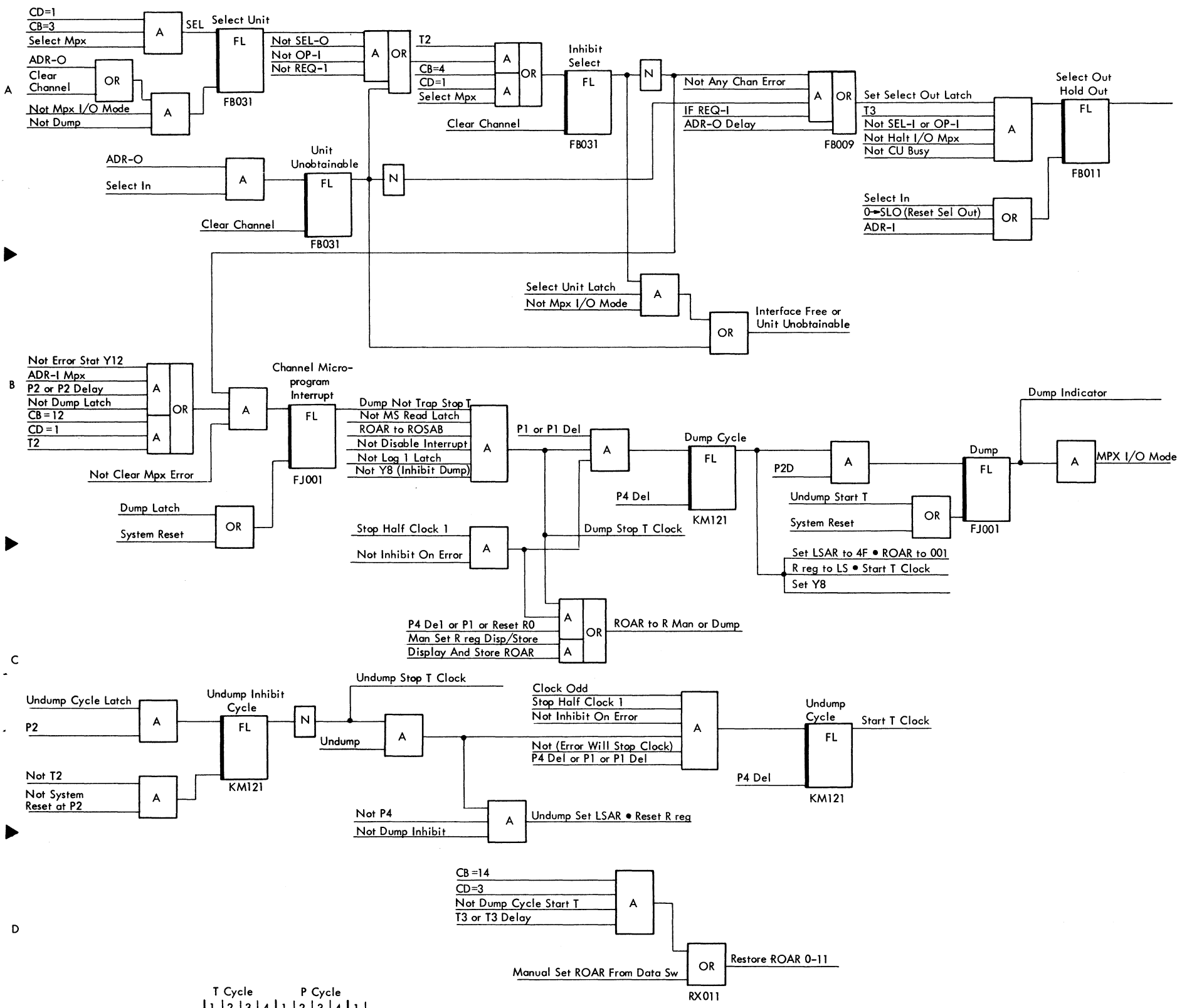


FIGURE 511. 2.5 kHz HF POWER SUPPLY WIRING DIAGRAM

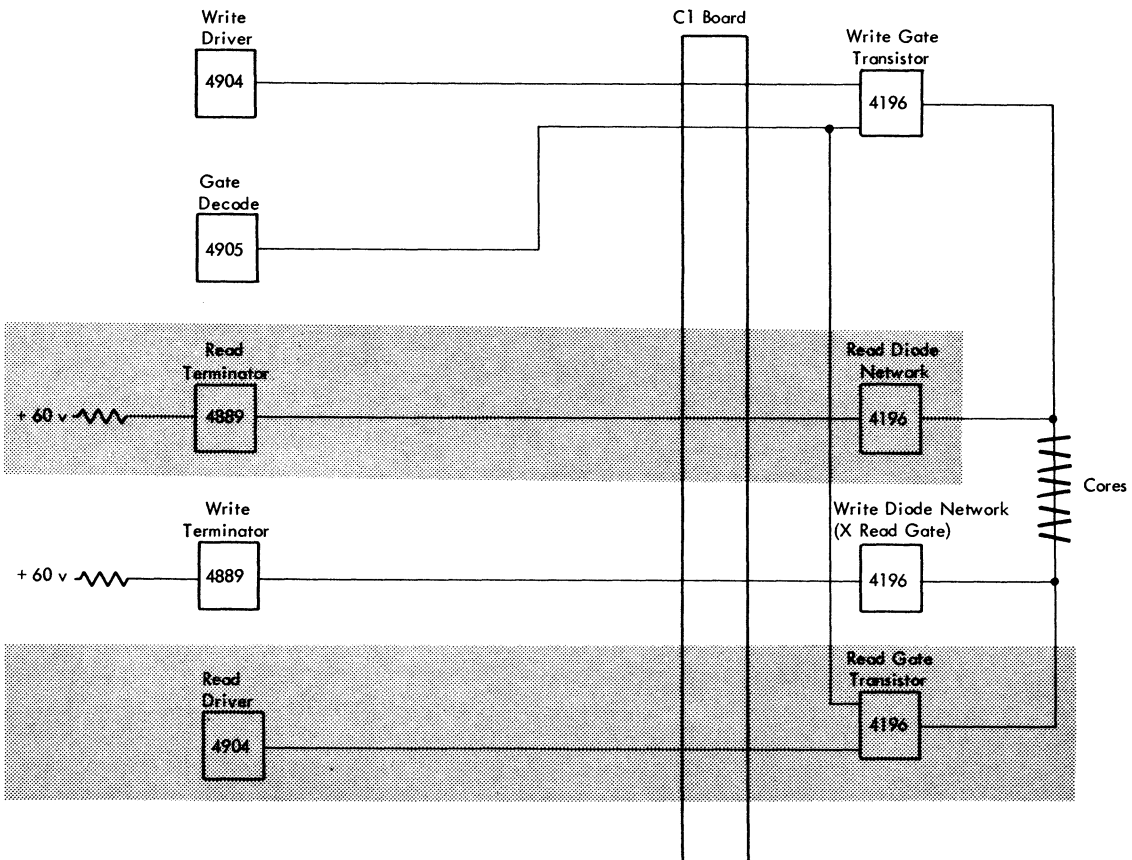


Contents of ROAR transfer to R reg at T4 delay of last machine cycle. T clock stops. During hardware cycle, LSAR is set to 4F; contents of ROAR (now in R reg) stored at address 4F in local storage. ROAR is reset, then set to 001. The T clock is restarted and the dump microprogram is executed.

FIGURE 512. MULTIPLEX CHANNEL CONTROLS

1 2 3 4

SIMPLIFIED DRIVE SYSTEM



POINT-TO-POINT EXAMPLE - GATE 05, DRIVER 04

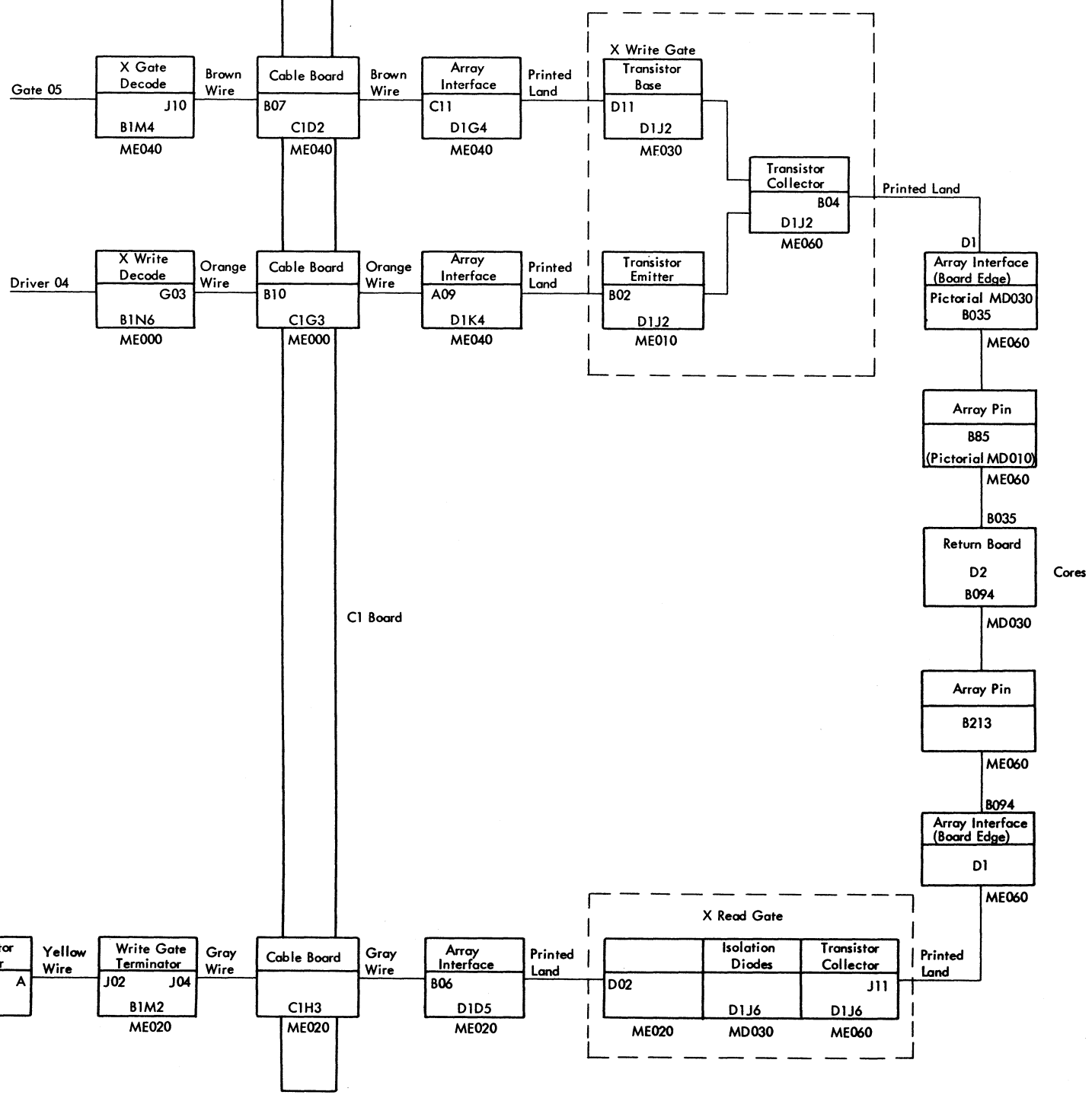


FIGURE 513. MAIN STORAGE X-DIMENSION DRIVE

Flow charts are provided for most machine instructions as tools to assist in diagnosing errors. To be used effectively, their basic philosophy and intended purpose must be understood.

Flow charts are a graphic representation of the complex sequence of loops or routines in CAS diagrams. Insignificant CAS blocks are not represented in the flow charts. In most cases, the notes in the decision blocks are taken directly from CAS diagrams for quick reference to actual CAS blocks. Flow charts guide the customer engineer to the correct CAS block and shorten diagnostic time.

Each figure contains a flow chart, objectives, instruction and data formats, the condition of STATS, the contents of registers at reference points, and the CAS page numbers. When an instruction has more than one format, all possibilities are shown. DMC version 4 was used to determine the data on these flow charts. The top right chart gives the section and routine numbers for each flow chart.

At several points (represented by circled numbers) in the flow chart, the current status of the input data is recorded for reference. These stop points can be duplicated with the use of Stop on ROS. The use of predetermined data aids in checking the machine operation quickly. If a machine failure is data-sensitive, the data can be changed to cause a failure.

Figure 618 is an example of how the flow charts and the DMC test procedure can be used to isolate a machine trouble. If a branch on condition failure that does not result in a hardstop occurs, Figure 618 may be used to locate the failure. First, load DMC and call in the test section and routine indicated in the chart (Section 120, Routine 10). When the DMC title prints on the console printer, enter the console keyboard L120/C.10/B/. This will read in and loop Section 120, Routine 10.

The flow chart (Figure 618) has four check points (circled numbers). These check points can be used by placing the proper ROS address in the data switches and the address compare switch to Stop on ROS. By comparing the actual data in the registers with the predetermined data, the failure can be isolated to a small section of CAS QE 031.

The flow charts are only a starting point; they do not replace the need for a thorough understanding of CAS symbols and diagrams. These flow charts also provide standard data for effective communication when discussing machine problems with area specialists or plant personnel.

FIGURE 599. HOW TO USE FLOW CHARTS

	-0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-A	-B	-C	-D	-E	-F
0-					04 SPM Set Program Mask 614 QC001 388	05 BALR Branch and Link 608 QE011 38A	06 BCTR Branch on Count 612 QE001 38C	07 BCR Branch on Condition 618 QE031 38E	08 SSK Set Storage Key 609 QC041 390	09 ISK Insert Storage Key 609 QC041 392	0A SVC Supervisor Call QC001 394					
1-	10 LPR Load Positive 611 QH071 381	11 LNR Load Negative 611 QH071 383	12 LTR Load and Test 611 QH071 385	13 LCR Load Complement 611 QH001 387	14 NR AND 615 QH001 389	15 CLR Compare Logical 615 QH001 38B	16 OR OR 615 QH001 38D	17 XR Exclusive OR 615 QH001 38F	18 LR Load 615 QH001 391	19 CR Compare 615 QH001 393	1A AR Add 615 QH001 395	1B SR Subtract 615 QH001 397	1C MR Multiply 620 QJ091 399	1D DR Divide 623 QJ241 398	1E ALR Add Logical 615 QH101 39D	1F SLR Subtract Logical 615 QH101 39F
2-	20 LPDR Load Positive (Flt Long) 625 QK011 A00	21 LNDR Load Negative (Flt Long) 625 QK011 A02	22 LTDR Load and Test (Flt Long) 611 QK011 A04	23 LCDR Load Complement (Flt Long) 611 QK011 A08	24 HDR Halve 625 QK011 A0B				28 LDR Load 625 QK011 A10	29 CDR Compare 626 QK021 A12	2A NADR Add Normalized 626 QK021 A14	2B NSDR Subtract Normalize 626 QK021 A16	2C NMDR Multiply (Flt Long) 628 QJ121 A18	2D NDDR Divide (Flt Long) 628 QJ121 A1A	2E AWR Add Unnormalized 626 QK021 A1C	2F SWR Subtract Unnormalized 626 QK021 A1E
3-	30 LPER Load Positive (Flt Short) 625 QK001 A00	31 LNER Load Negative (Flt Short) 625 QK011 A02	32 LTER Load and Test (Flt Short) 625 QK011 A04	33 LCER Load Complement (Flt Short) 625 QK011 A08	34 HER Halve 625 QK011 A0B				38 LER Load 625 QK011 A10	39 CER Compare 626 QK021 A12	3A NAER Add Normalized 626 QK021 A14	3B NSER Subtract Normalized 626 QK021 A16	3C NMER Multiply (Flt Short) 628 QJ121 A18	3D NDER Divide (Flt Short) 628 QJ121 A1A	3E AUR Add Unnormalized 626 QK021 A1C	3F SUR Subtract Unnormalized 626 QK021 A1E
4-	40 STH Store Halfword 615 QH031 301	41 LA Load Address QH031 303	42 STC Store Character QH031 305	43 IC Insert Character QH031 307	44 EX Execute QE001 309	45 BAL Branch and Link 608 QE011 30B	46 BCT Branch on Count 612 QE021 30D	47 BC Branch on Condition 618 QE031 30F	48 LH Load Halfword QH031 311	49 CH Compare Halfword 617 QH091 313	4A AH Add Halfword 616 QH081 315	4B SH Subtract Halfword 616 QH081 317	4C MH Multiply Halfword QJ001 319		4E CVD Convert to Decimal 613 QM001 31D	4F CVB Convert to Binary 610 QM021 31F
5-	50 ST Store 615 QH011 300				54 N AND 615 QH011 308	55 CL Compare Logical 615 QH011 30A	56 O OR 615 QH011 30C	57 X Exclusive OR 615 QH011 30E	58 L Load 615 QH011 310	59 C Compare 615 QH011 312	5A A Add 615 QH011 314	5B S Subtract 615 QH011 316	5C M Multiply 620 QJ091 318	5D D Divide 623 QJ241 31A	5E AL Add Logical 615 QH101 31C	5F SL Subtract Logical 615 QH101 31E
6-	60 STD Store (Flt Long) 627 QK001 A01								68 LD Load (Flt Long) 627 QK001 A11	69 CD Compare 626 QK031 A13	6A NAD Add Normalized 626 QK031 A15	6B NSD Subtract Normalized 626 QK031 A17	6C NMD Multiply (Flt Long) 628 QJ121 A19	6D NDD Divide (Flt Long) 628 QJ121 A1B	6E AW Add Unnormalized 626 QK031 A1D	6F SW Subtract Unnormalized 626 QK031 A1F
7-	70 STE Store (Flt Short) 627 QK001 A01								78 LE Load (Flt Short) 627 QK001 A11	79 CE Compare 626 QK031 A13	7A NAE Add Normalized 626 QK031 A15	7B NSE Subtract Normalized 626 QK031 A17	7C NME Multiply (Flt Short) 628 QJ121 A19	7D NDE Divide (Flt Short) 628 QJ121 A1B	7E AU Add Unnormalized 626 QK031 A1D	7F SU Subtract Unnormalized 626 QK031 A1F
8-	80 SSM Set System Mask 633 QC021 340		82 LPSW Load PSW 628 QC021 344	83 Diagnose 639	84 WRD Write Direct 637 QC151 348	85 RDD Read Direct 637 QC151 34A	86 BXH Branch Index High 632 QE041 34C	87 BXLE Branch Index Lo-Eq 632 QE041 34E	88 SRL Shift Right Single Logical 635 QL001 350	89 SLL Shift Left Single Logical 635 QL001 352	8A SRA Shift Right Single 635 QL001 354	8B SLA Shift Left Single 635 QL001 356	8C SRDL Shift Right Double Logical 635 QL001 358	8D SLDL Shift Left Double Logical 635 QL001 35A	8E SRDA Shift Right Double 635 QL001 35C	8F SLDA Shift Left Double 635 QL001 35E
9-	90 STM Store Multiple 634 QH041 341	91 TM Test Under Mask 631 QP001 343	92 MVI Move 636 QP001 345	93 TS Test and Set QP001 347	94 NI AND 636 QP001 349	95 CLI Compare Logical QP001 34B	96 OI OR 636 QP001 34D	97 XI Exclusive OR 636 QP001 34F	98 LM Load Multiple 631 QH041 351				9C SIO Start I/O 667 QB101 359	9D TIO Test I/O 668 QB101 35B	9E HIO Halt I/O 666 QB101 35D	9F TCH Test Channel 666 QB101 35F
A-																
B-																
C-																
D-		D1 MVN Move Numeric 647 QP041 C02	D2 MVC Move Character 648 QP041 C04	D3 MVZ Move Zone 647 QP041 C06	D4 NC AND QP041 C08	D5 CLC Compare Logical 649 QP041 C0A	D6 OC OR QP041 C0C	D7 XC Exclusive OR QP041 C0E					DC TR Translate 641 QP061 C18	DD TRT Translate and Test 642 QP061 C1A	DE ED Edit 643 QD131 C1C	DF EDMK Edit and Mark 643 QD131 C1E
E-																
F-		F1 MVO Move With Offset 656 QR101 C03	F2 PACK Pack 654 QR101 C05	F3 UNPK Unpack 655 QR101 C07					F8 ZAP Zero and Add 657 QD131 C11	F9 CP Compare Decimal 657 QD131 C13	FA AP Add Decimal 657 QD131 C15	FB SP Subtract Decimal 657 QD131 C17	FC MP Multiply Decimal 653 QD131 C19	FD DP Divide Decimal 650 QD131 C1B		

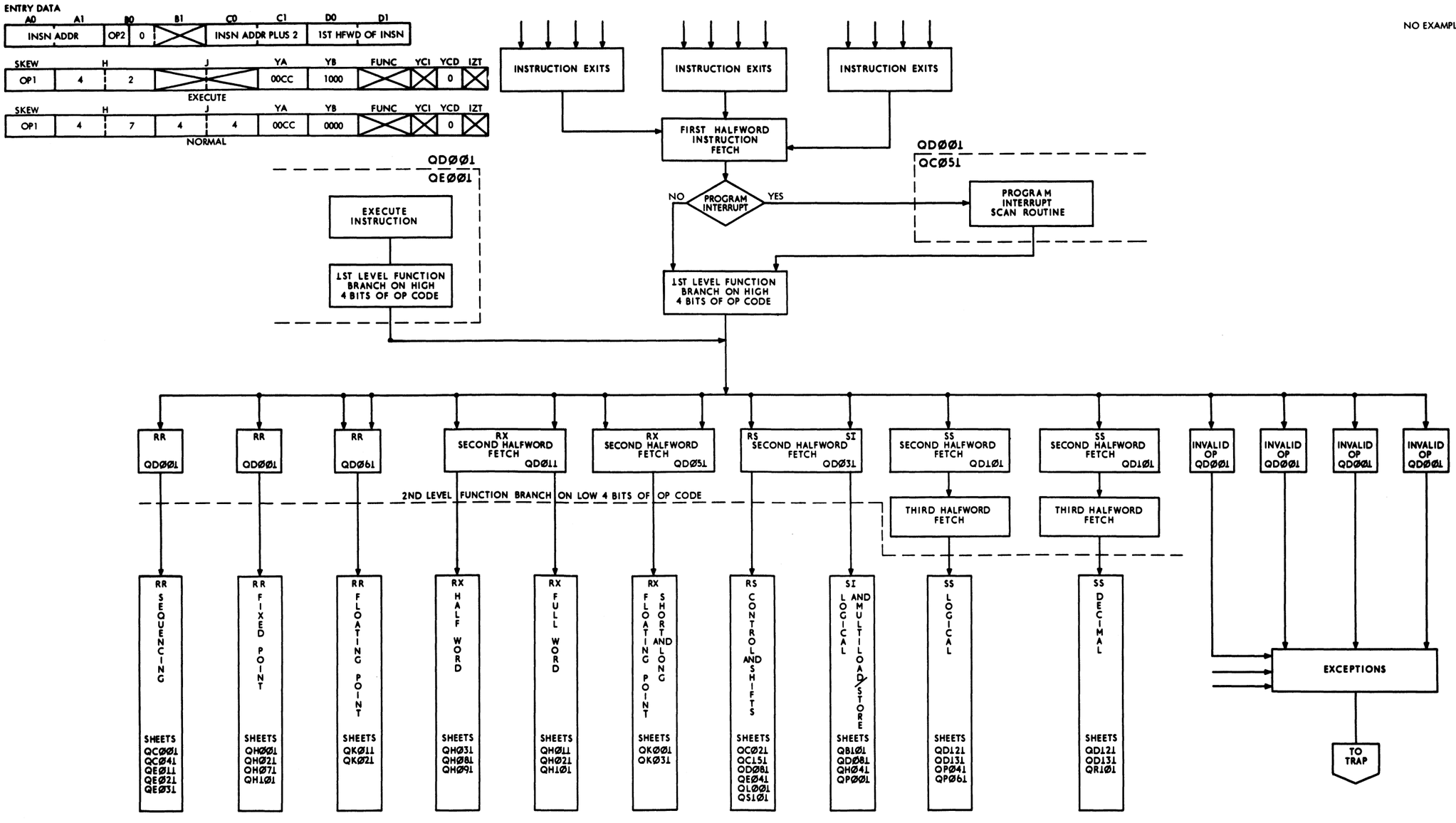
Op Code — Mnemonic

Instruction

Figure Number

CAS Page — CAS Address

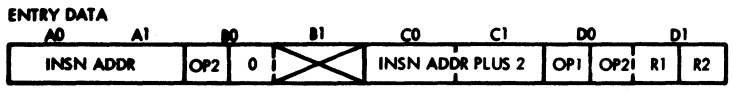
FIGURE 600. INSTRUCTION MATRIX

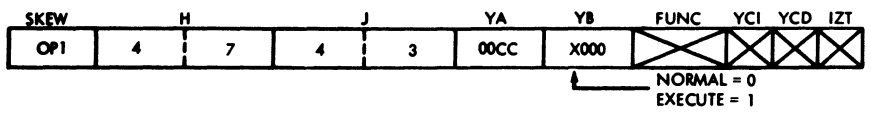


NO EXAMPLE SHOWN.

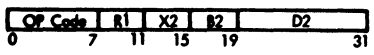
FIGURE 601. INSTRUCTION FETCH MICROPROGRAM

ENTRY DATA





RX



NO EXAMPLE SHOWN.

OBJECTIVE
TO READ 2ND HFWD OF
INSTRUCTION FROM MAIN STORE
AND PERFORM A FUNCTION BRANCH
ON THE LOW ORDER 4 BITS OF
THE OP CODE

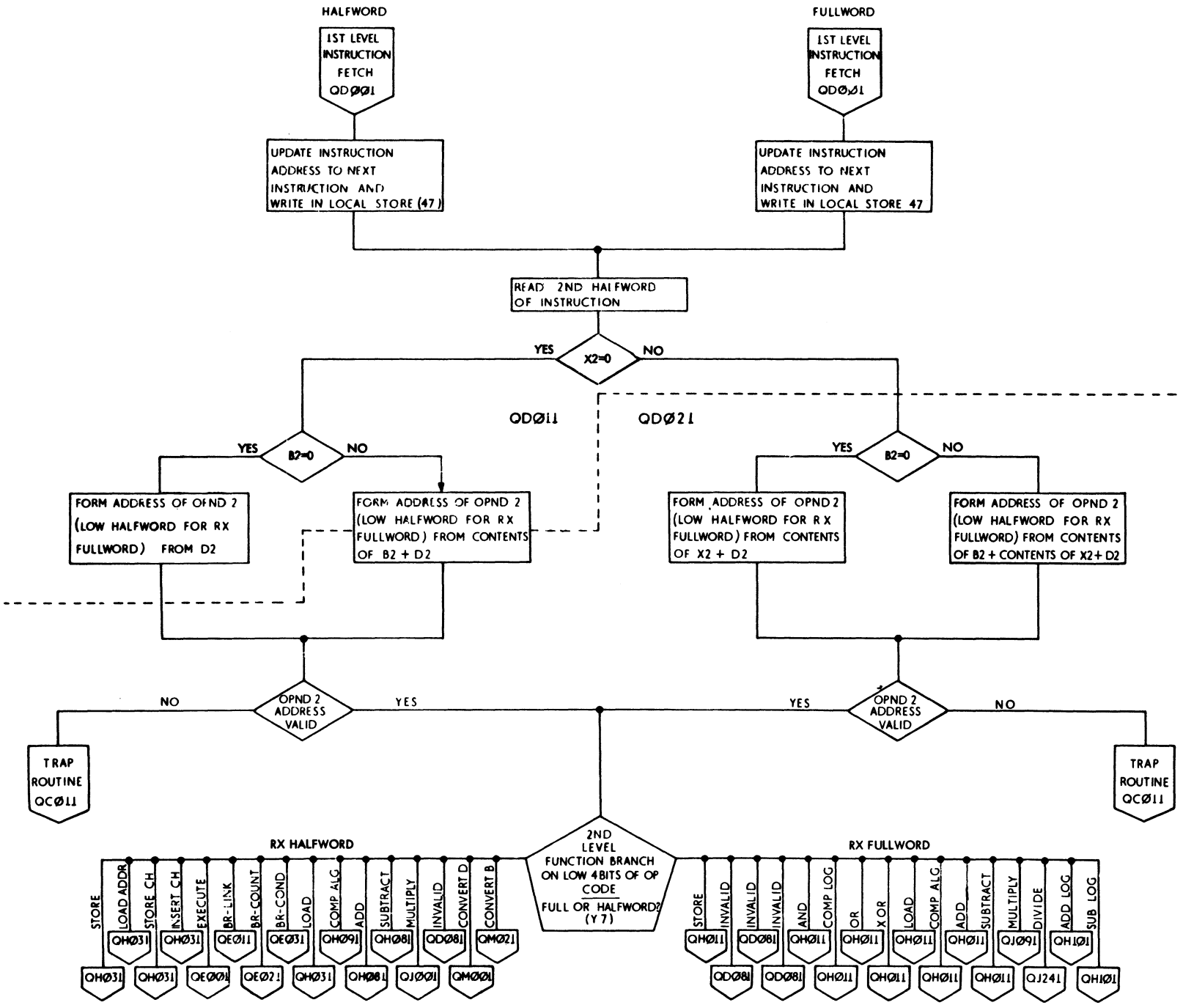
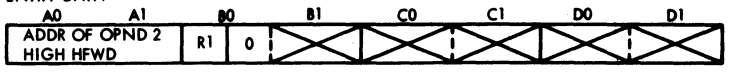
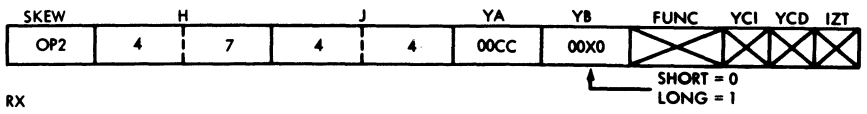


FIGURE 602. 2ND LEVEL INSTRUCTION FETCH, RX FIXED POINT

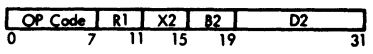
NO EXAMPLE SHOWN.

ENTRY DATA





RX



OBJECTIVES

FORM THE EFFECTIVE ADDRESS OF OPND 2 AND PERFORM A FUNCTION BRANCH ON THE LOW ORDER 4 BITS OF THE OP CODE

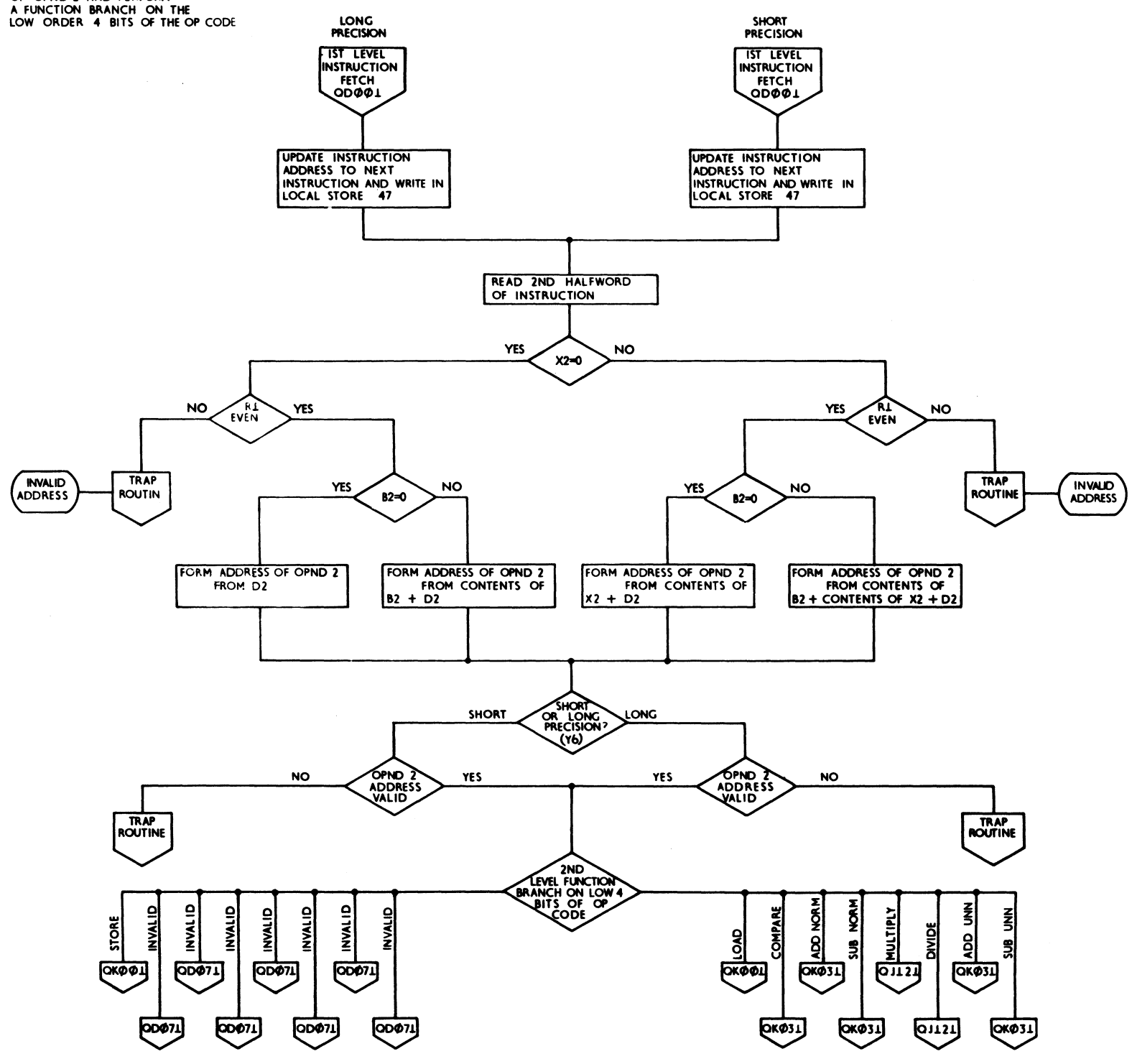
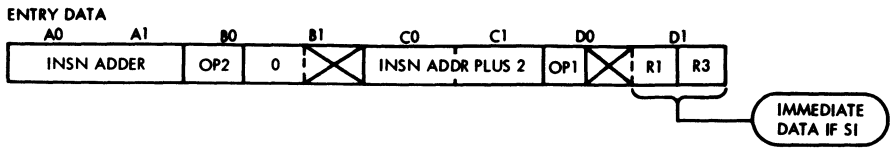
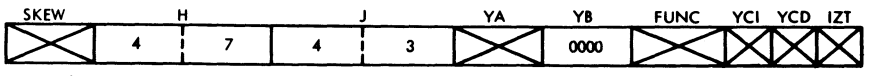


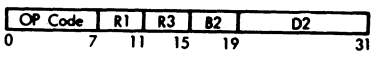
FIGURE 603. 2ND LEVEL INSTRUCTION FETCH, RX FLOATING POINT (QD051)

ENTRY DATA

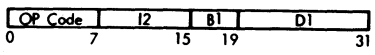




RS



SI



NO EXAMPLE SHOWN.

OBJECTIVE

TO READ 2ND HFWD OF INSTRUCTION FROM MAIN STORE AND PERFORM A FUNCTION. BRANCH ON THE LOW ORDER 4 BITS OF THE OP CODE.

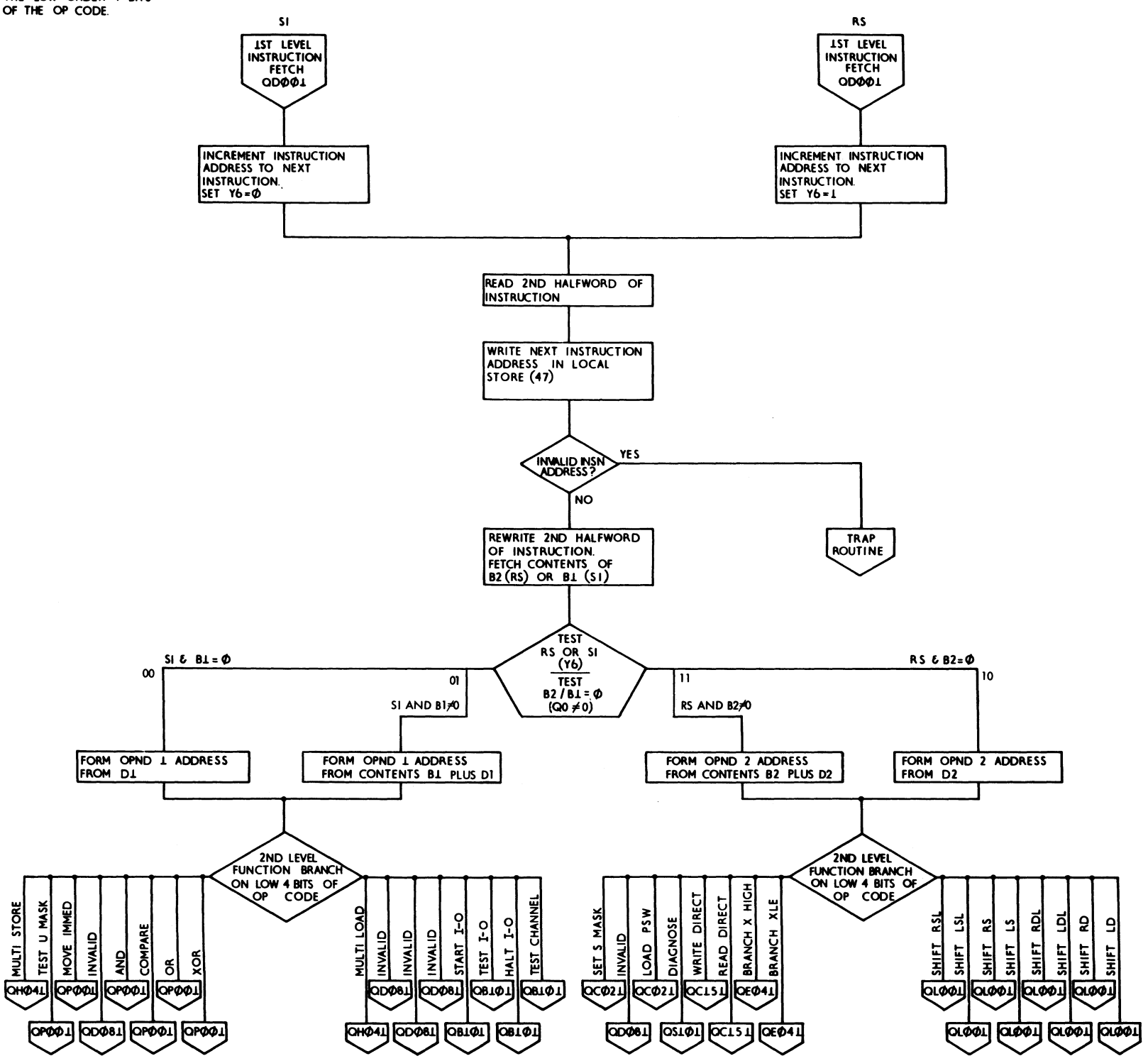


FIGURE 604. 2ND LEVEL INSTRUCTION FETCH, RS AND SI OPERATIONS (QD031)

NO EXAMPLE SHOWN.

ENTRY DATA

A0	A1	B0	B1	C0	C1	D0	D1
ADDR OF 1ST HFWD OF INSN		OP2	0	ADDR OF 2ND HFWD OF INSN		OP1	OP2	L1	L2

SKEW	H	J	YA	YB	FUNC	YCI	YCD	IZT
OP1	4	7	4	4	00CC	0000				

SS

OP Code	L1	L2	B1	D1	B2	D2
0 7	11	15	19	31	35	47

1st HFWD | 2nd HFWD | 3rd HFWD

OBJECTIVES

TO FORM THE EFFECTIVE ADDRESS OP OPND 1 AND OPND 2, AND PERFORM A FUNCTION BRANCH ON THE LOW ORDER 4 BITS OF THE OP CODE

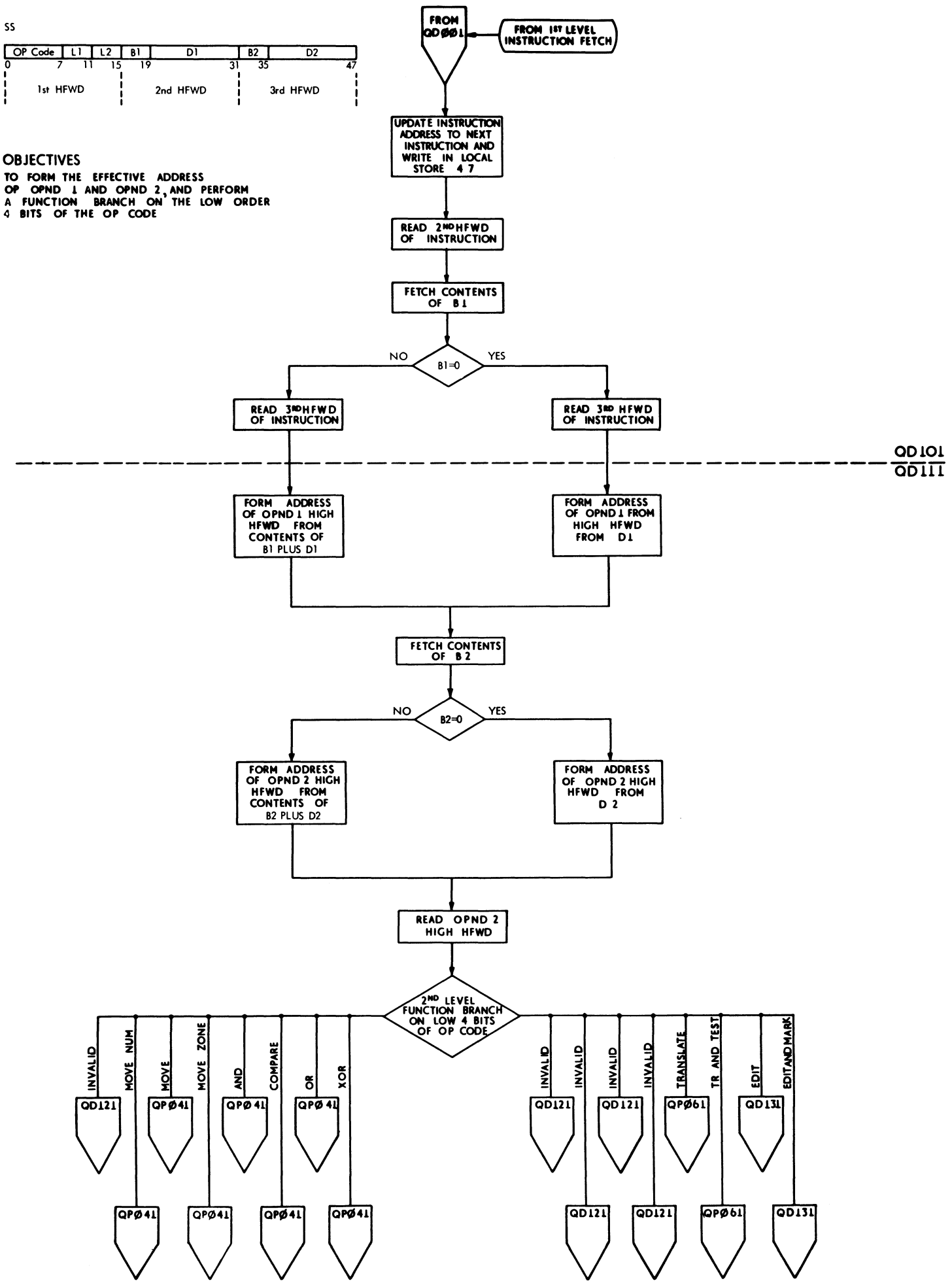
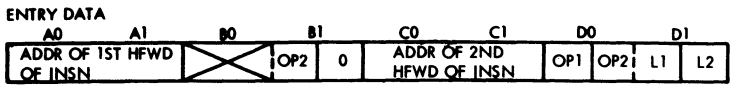
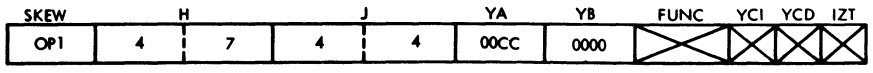


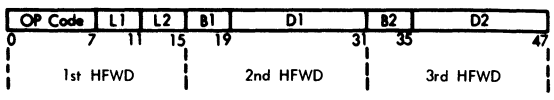
FIGURE 605.2ND LEVEL INSTRUCTION FETCH, SS LOGICAL

ENTRY DATA





SS



OBJECTIVES

TO FORM THE EFFECTIVE ADDRESS OF OPND 1 AND OPND 2, AND PERFORM A FUNCTION BRANCH ON THE LOW ORDER 4 BITS OF THE OP CODE.

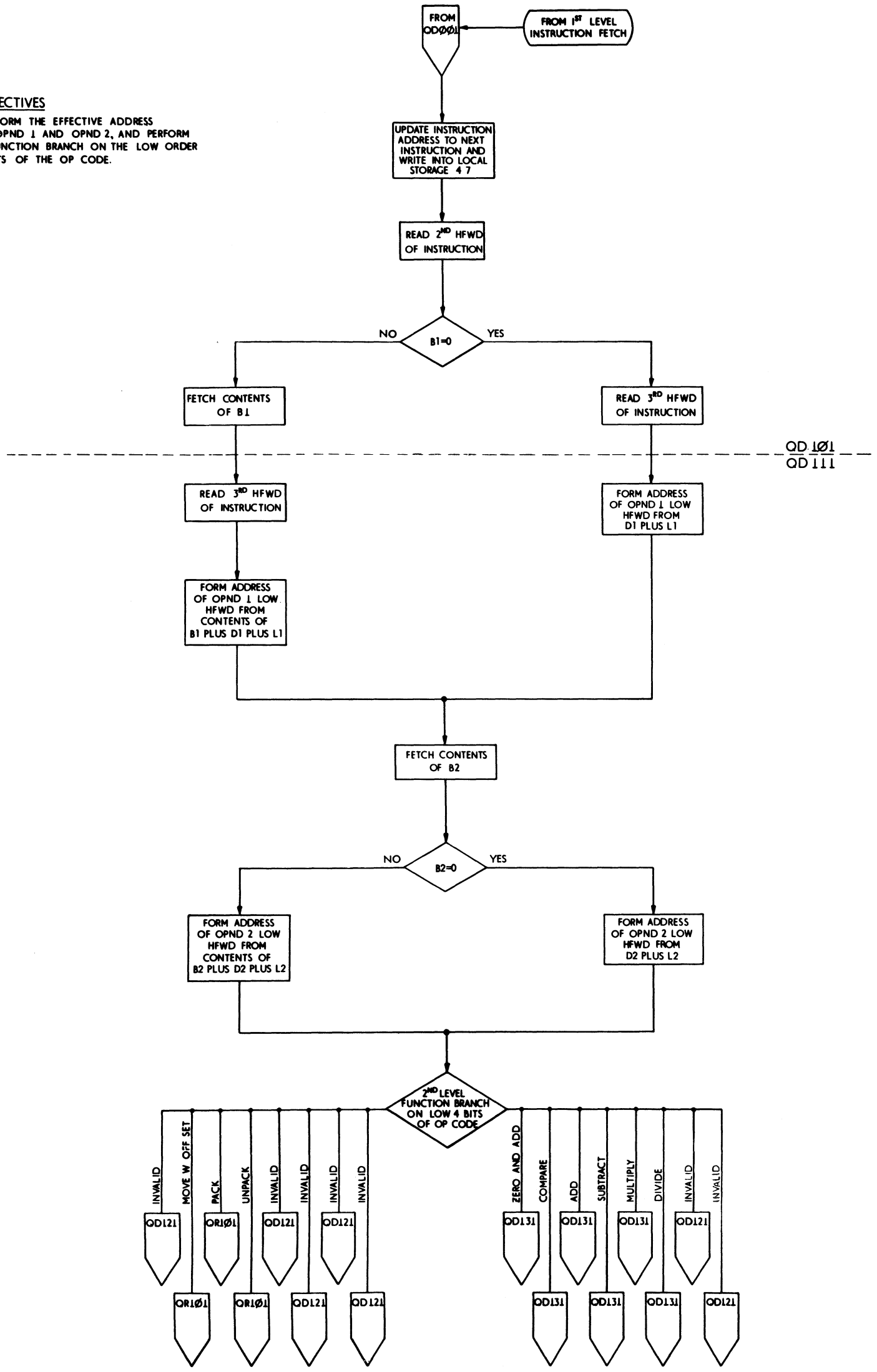


FIGURE 606 2ND LEVEL INSTRUCTION FETCH, SS DECIMAL

NO EXAMPLE SHOWN.

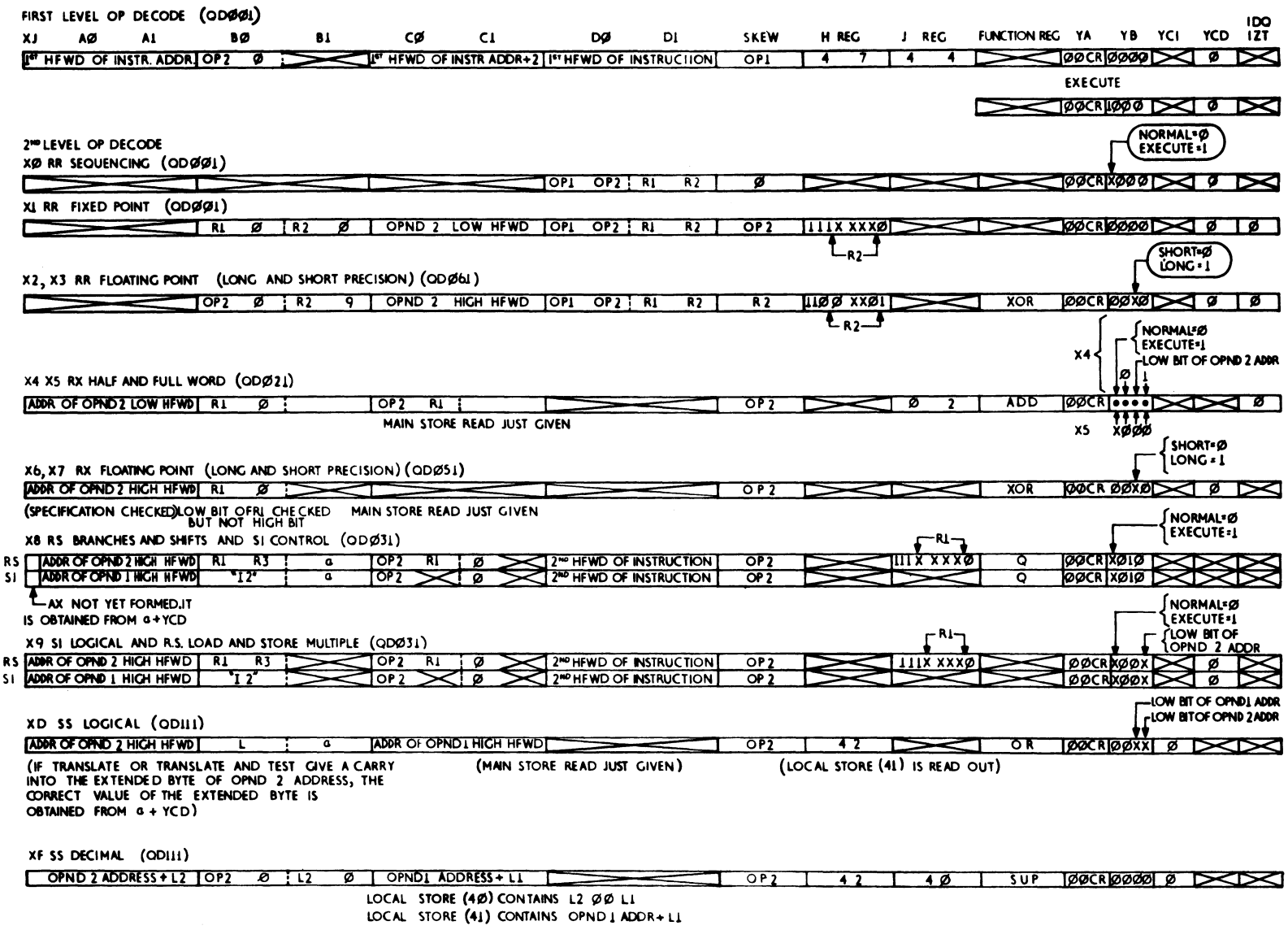
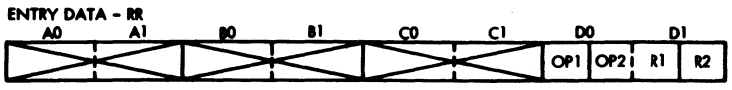
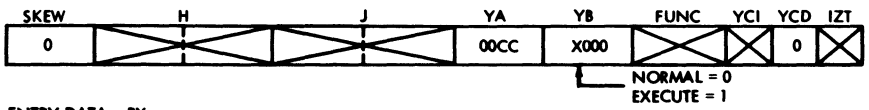


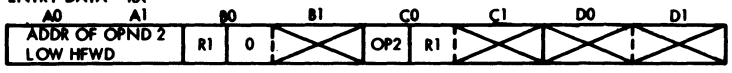
FIGURE 607 MACHINE STATUS AT 1ST AND 2ND LEVEL FUNCTION BRANCHES

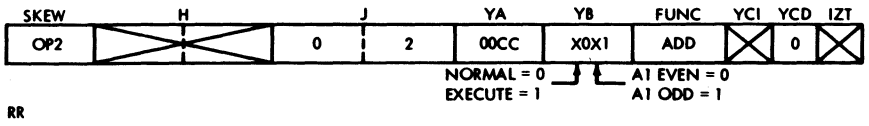
ENTRY DATA - RR



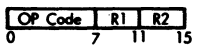


ENTRY DATA - RX

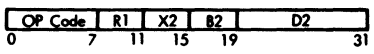




RR



RX



BRANCH AND LINK BALR 01, 02 DMC SECTION 121, ROUTINE 11

	ROS	REG 1	REG 2	A REG	B REG	C REG	D REG
①	38A	4F006262	00006268	6260	5060	6262	0512
②	3A8	4F006262	00006268	6268	2000	6262	0512
③	3E2	4F006262	00006268	6268	4000	6262	4F00

OBJECTIVE

THE RIGHTMOST 32 BITS OF THE PSW ARE STORED IN THE REGISTER SPECIFIED BY R1.
THE BRANCH ADDRESS IS FORMED TO REPLACE THE INSTRUCTION ADDRESS IN THE RR FORMAT IF R2=Ø
THIS INDICATES NO BRANCH

OP CODE

Ø5—RR
45—RX HALF

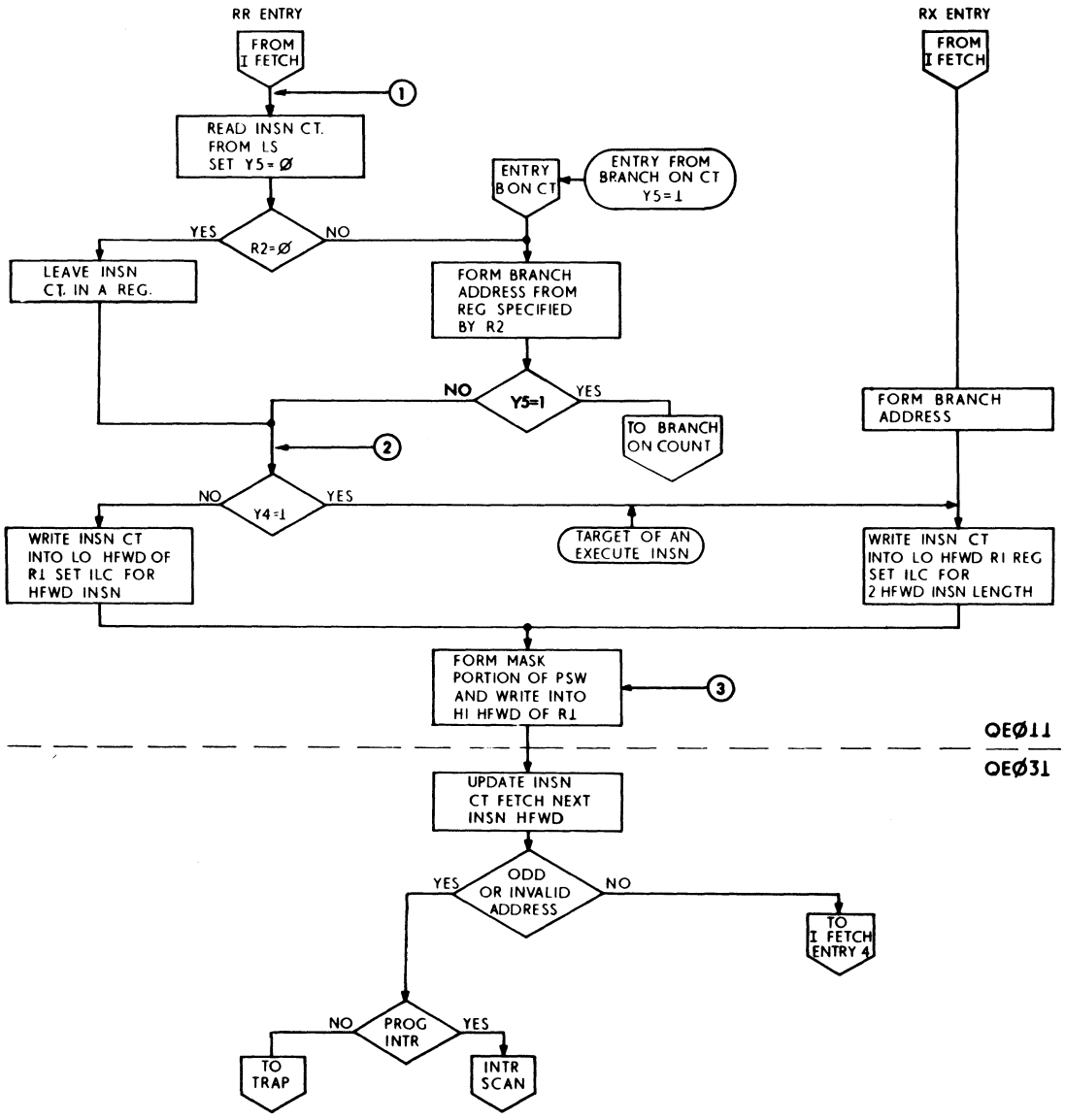
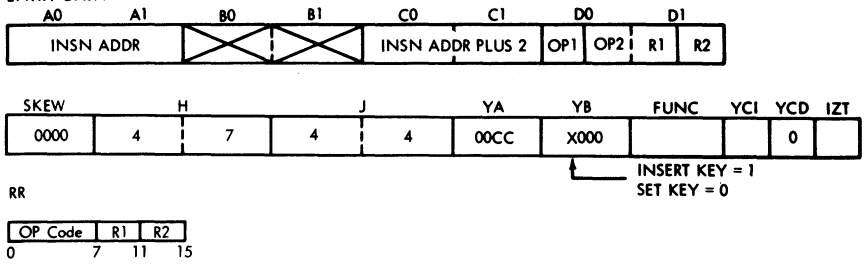


FIGURE 608. BRANCH AND LINK

ENTRY DATA



RR

NO EXAMPLE SHOWN.

OBJECTIVES.

SET STORE KEY.

THE KEY OF THE STORAGE BLOCK ADDRESSED BY THE REGISTER DESIGNATED BY R2 IS SET ACCORDING TO THE REGISTER DESIGNATED BY R1.

INSERT STORAGE KEY.

THE KEY OF THE STORAGE BLOCK ADDRESSED BY THE REGISTER DESIGNATED BY R2 IS INSERTED IN THE REGISTER DESIGNATED BY R1.

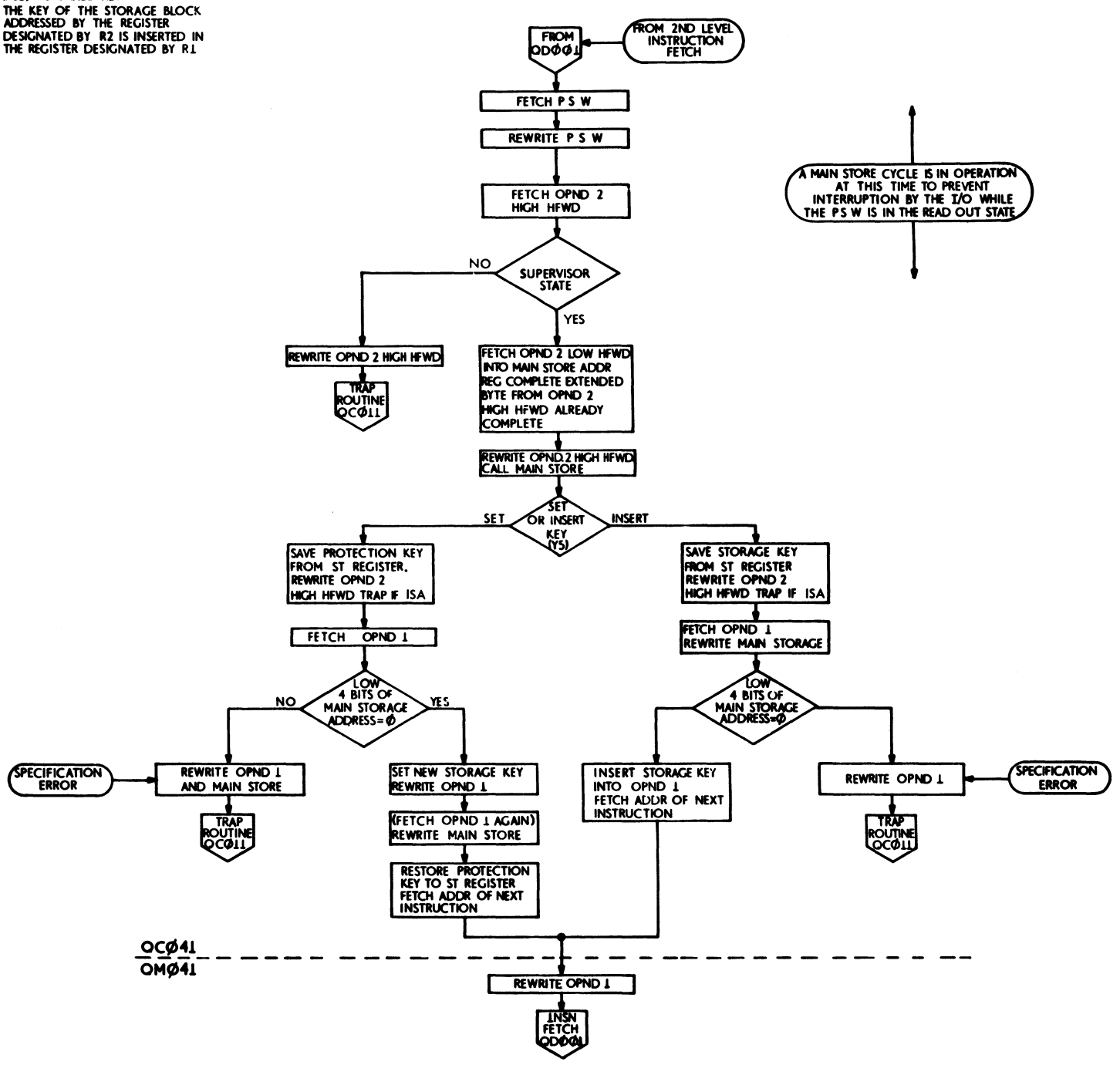


FIGURE 609. SET AND INSERT STORAGE KEY

ENTRY DATA

A0 A1	B0	B1	C0	C1	D0 D1
ADDR OF OPND 2 LOW HFWD	R1	0	OP2	R1	OPND 2 LOW HFWD

SKEW	H	J	YA	YB	FUNC	YCI	YCD	IZT
OP2		0	2		ADD			

RX

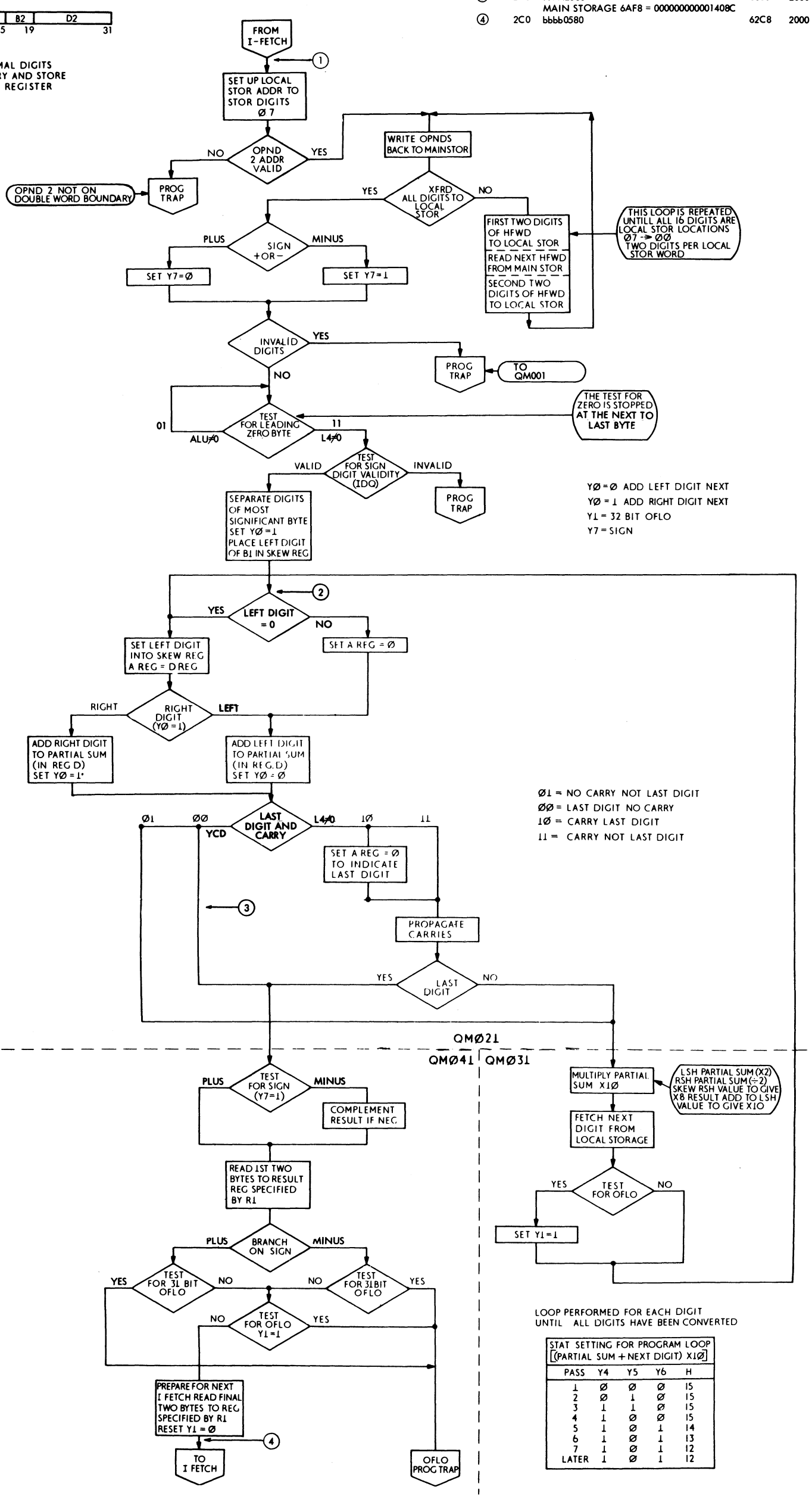
OP Code	R1	X2	B2	D2
0 7	11	15	19	31

OBJECTIVE

TO CONVERT 15 DECIMAL DIGITS PLUS SIGN TO BINARY AND STORE THEM IN A GENERAL REGISTER

CONVERT TO BINARY CVB DMC SECTION 167, ROUTINE 05

	ROS	REG 5	REG 6	A REG	B REG	C REG	D REG
①	31F	000068C4	000068C0	6AF8	2000	0000	0000
		MAIN STORAGE 68C4 = 0000000000001408C					
		MAIN STORAGE 68C0 = 0000000000000580					
②	2AF	FFA42580		2040	2001	0000	0000
③	2A0	FFA42580		0000	2000	0000	0000
		MAIN STORAGE 6AF8 = 00000000001408C					
④	2C0	bbbb0580		62C8	2000	0000	0580



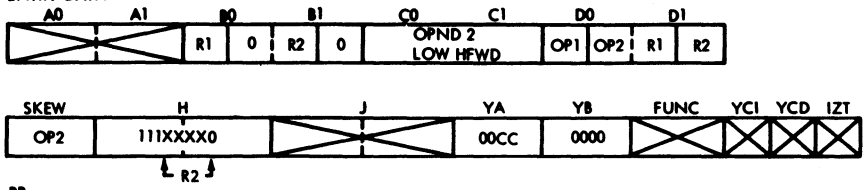
LOOP PERFORMED FOR EACH DIGIT UNTIL ALL DIGITS HAVE BEEN CONVERTED

STAT SETTING FOR PROGRAM LOOP [(PARTIAL SUM + NEXT DIGIT) X1Ø]

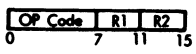
PASS	Y4	Y5	Y6	H
1	Ø	Ø	Ø	15
2	Ø	1	Ø	15
3	1	1	Ø	15
4	1	Ø	Ø	15
5	1	Ø	1	14
6	1	Ø	1	13
7	1	Ø	1	12
LATER	1	Ø	1	12

FIGURE 610 CONVERT DECIMAL TO BINARY

ENTRY DATA



RR



LOAD POSITIVE REGISTER DMC SECTION 16D, ROUTINE 01, STOP ON MS 6100

	ROS	GP REG 2	A REG	B REG	C REG	D REG
①	381	00000001	6100	2020	0001	1022
②	3BB	0000bbbb	61FF	0000	0001	0022
③	1E8	bbbb0001	3FFFF	0000	0001	0022
		CONTINUED ON FLOW CHART NUMBER 615				
④	1C9	00000001	6101	7030	0000	0000
⑤	1EE	00000001	6108	0000	0001	0022

OBJECTIVES

TO PERFORM THE RR FIXED POINT SIGN OPS. LOAD AND TEST, LOAD COMPLEMENT, LOAD POSITIVE, LOAD NEGATIVE. (FOR DESCRIPTION OF OBJECTIVES OF EACH INSTRUCTION SEE PRINCIPLES OF OPERATION MANUAL)

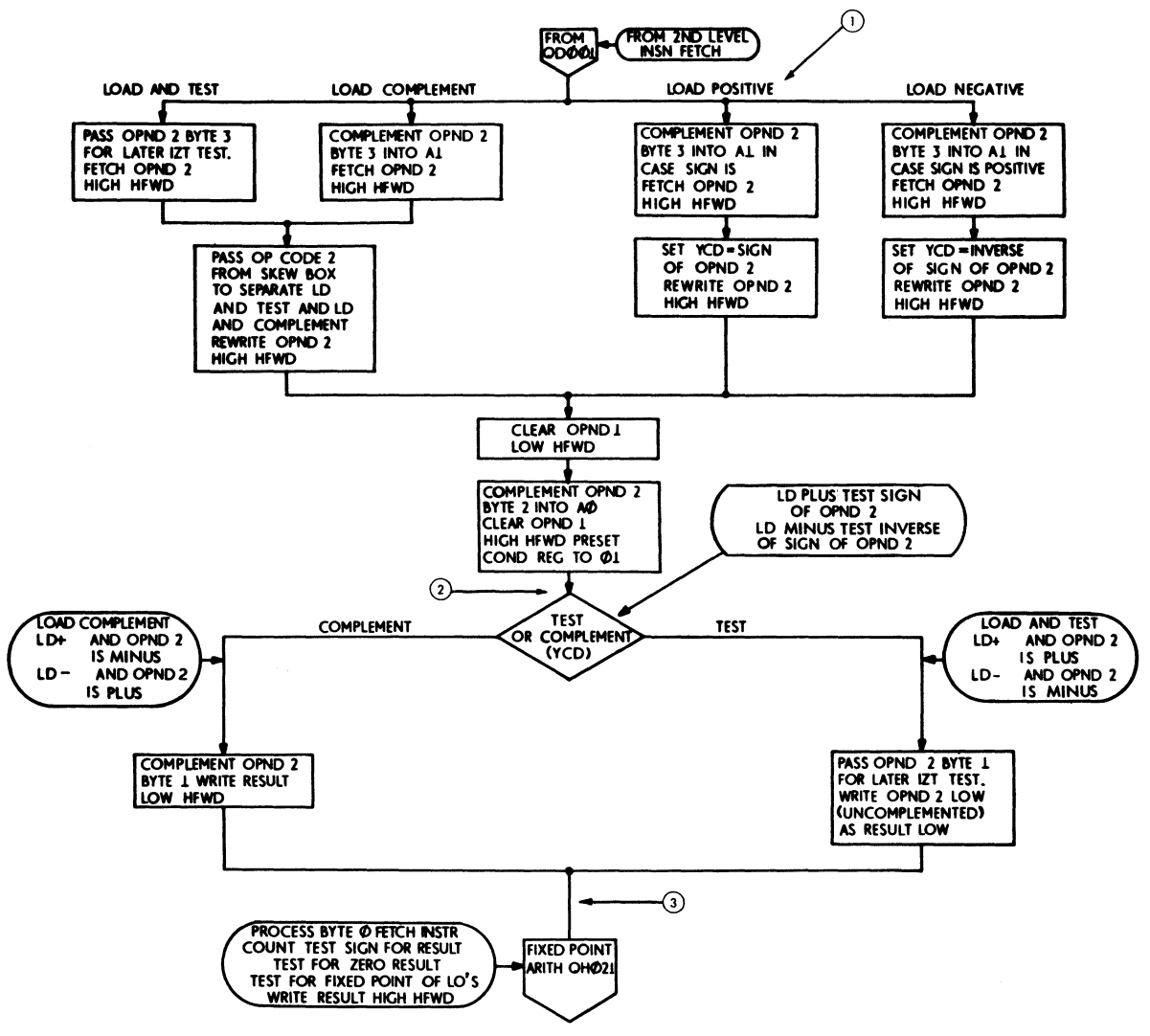
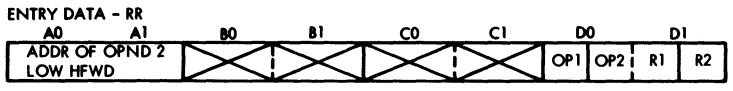
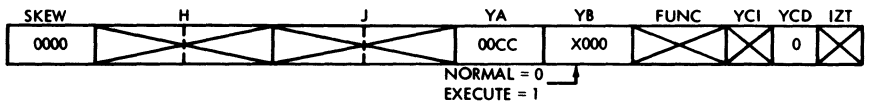


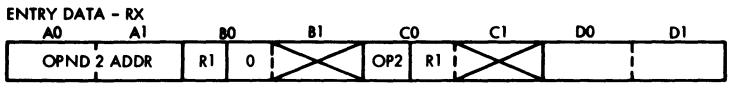
FIGURE 611 FIXED POINT SIGN OPERATIONS (QH071)

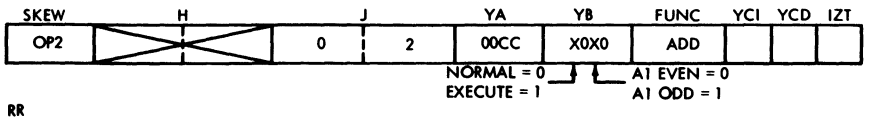
ENTRY DATA - RR





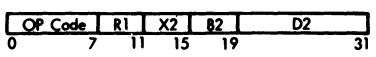
ENTRY DATA - RX





RR

RX



OBJECTIVE

THE CONTENTS OF REGISTER R1 ARE ALGEBRAICALLY REDUCED BY ONE. IF RESULT IS ZERO, NORMAL INSTRUCTION SEQUENCING TAKES PLACE. IF NOT ZERO INSTRUCTION ADDRESS IS REPLACED BY BRANCH ADDRESS

BRANCH ON COUNT BCT 02, 05 DMC SECTION 122, ROUTINE 25

	ROS	REG 2	REG 4	REG 5	A REG	B REG	C REG	D REG
①	38C	00000002	00000001	00006542	6546	6040	6548	0625
②	3E8	0000bbbb	00000001	00006542	6542	6002	6500	0625
③	3C9	bbbb0000	00000001	00006542	6542	0000	6500	0000

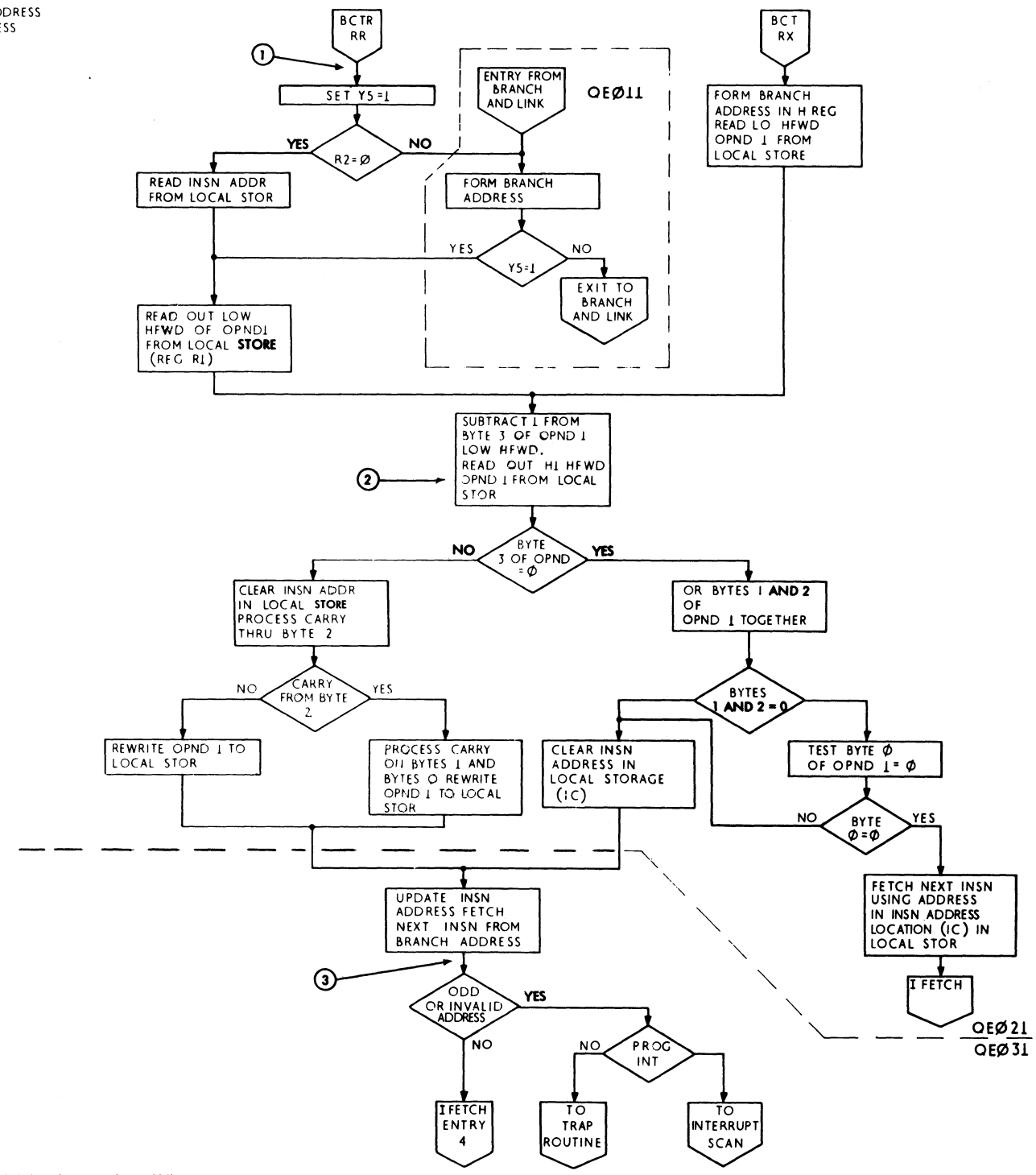
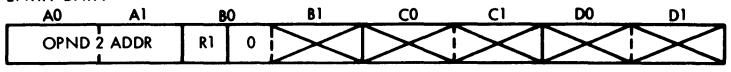
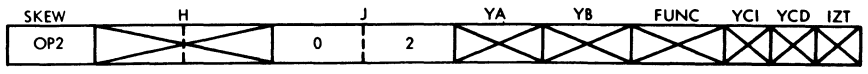


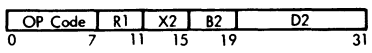
FIGURE 612. BRANCH ON COUNT (QE011, QE021, QE013)

ENTRY DATA





RX



CONVERT TO DECIMAL CVD DMC SECTION 167, ROUTINE 12

	ROS	REG 0	REG 8	A REG	B REG	C REG	D REG
①	31D	FFA42580	00000010	6AF8	4000	0000	0000
②	2CD	FFA42580	00000010	6AFD	4000	1000	0010
③	2FE	FFA42580	00000010	6AFE	4000	0000	016C

OBJECTIVES

TO CONVERT A 31 BIT BINARY
WORD PLUS SIGN TO 15 4 BIT DECIMAL
DIGITS PLUS 4 BIT SIGN

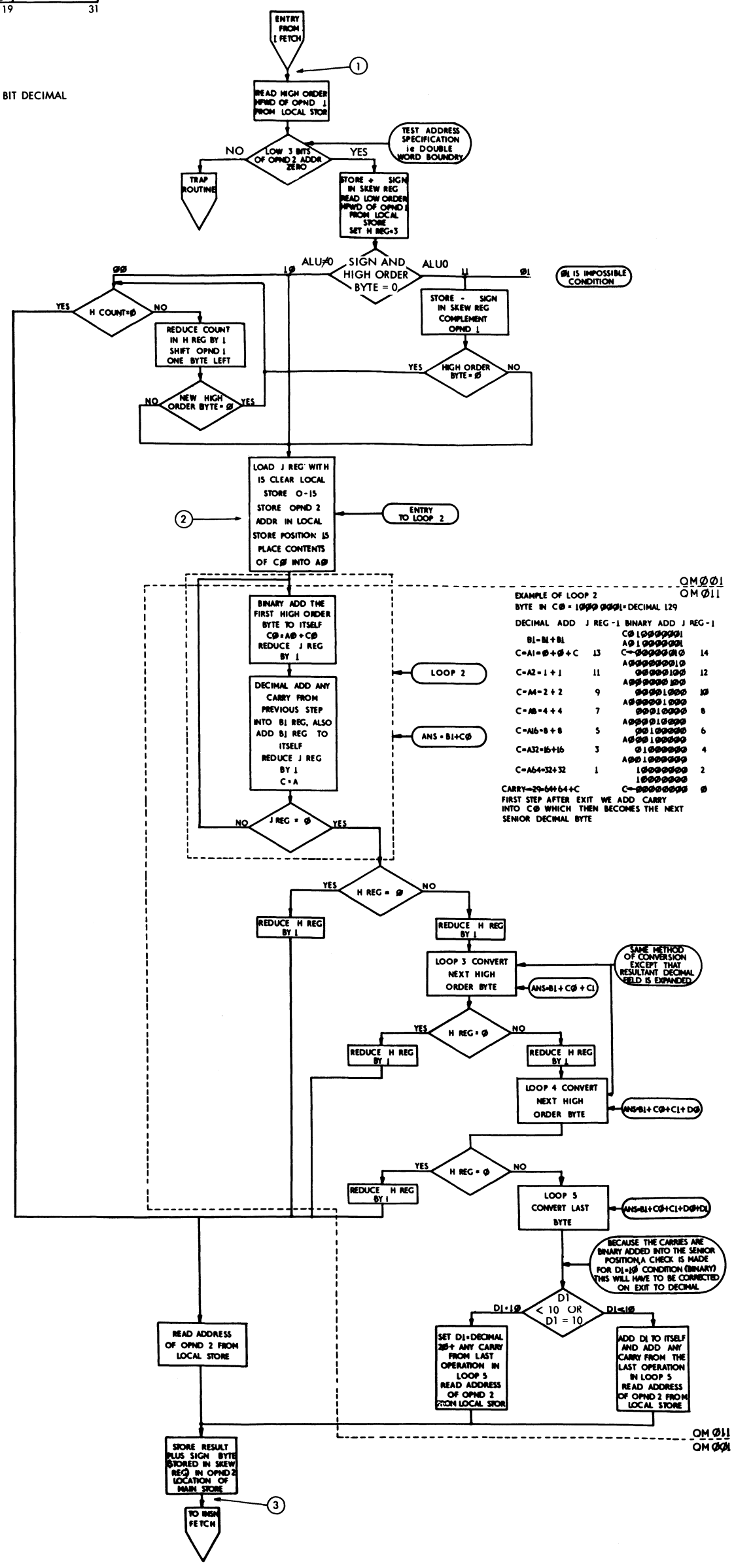


FIGURE 613. CONVERT BINARY TO DECIMAL

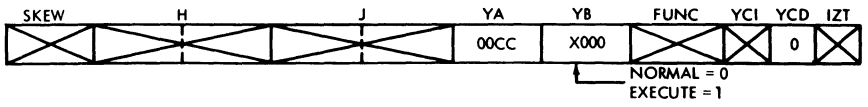
ENTRY DATA

SET PROGRAM MASK

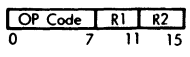
	ROS	REG 1
①	388	3F000000
②	3E4	3F000000

DMC SECTION 124, ROUTINE 24 STOP ON MS 6B00

A REG	B REG	C REG	D REG
6B00	4010	6B02	0410
6B02	4010	6B02	3F00



RR



OBJECTIVES

BITS 2 - 7 OF THE GENERAL REGISTER SPECIFIED BY R1 REPLACE THE PROGRAM MASK AND CONDITION REGISTER. BITS 0 - 1 AND 8 - 31 OF REG R1 ARE IGNORED.

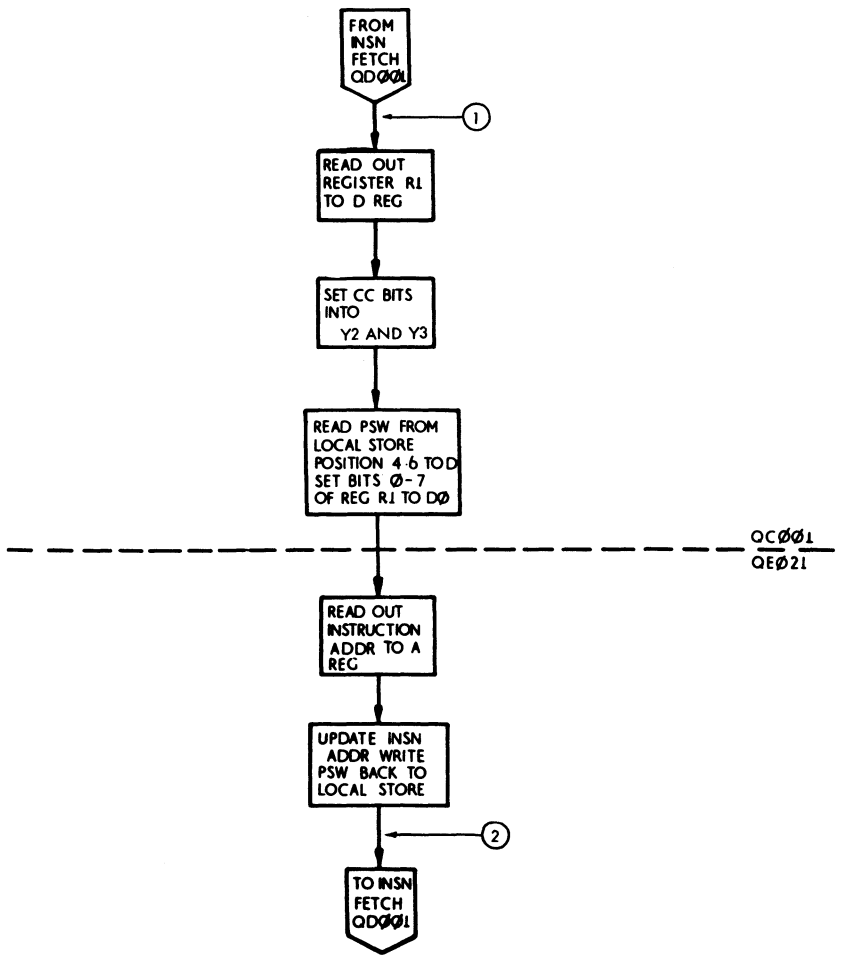
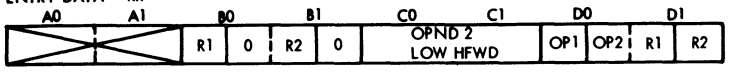
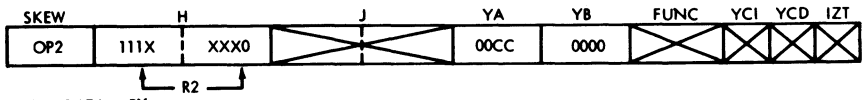


FIGURE 614. SET PROGRAM MASK

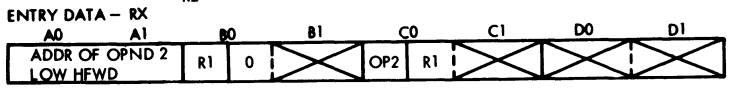
EXAMPLE CONTINUED FROM FIGURE 611

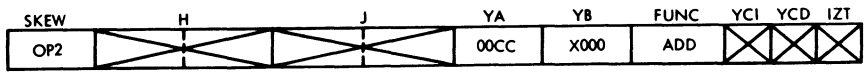
ENTRY DATA – RR



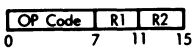


ENTRY DATA – RX

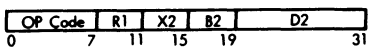




RR



RX



OBJECTIVES

TO PERFORM THE INSTRUCTIONS COVERED BY RR AND RX (FULLWORD) FIXED POINT ARITHMETIC AND LOGIC OPERATIONS

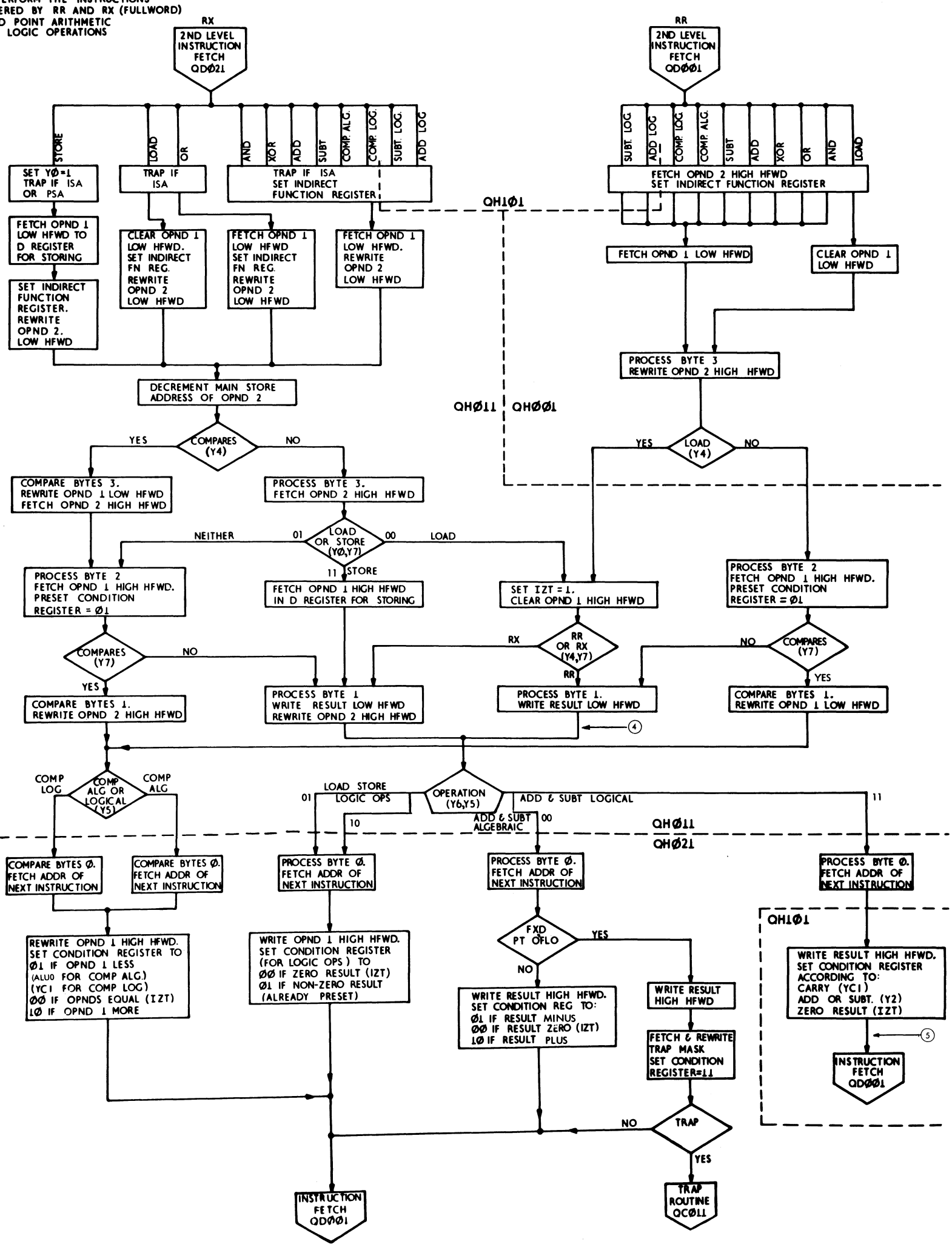
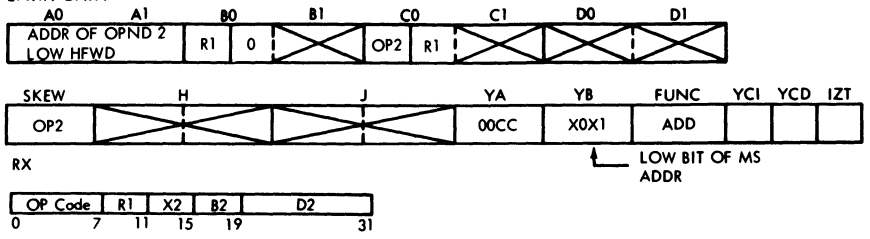


FIGURE 615. RR AND RX FIXED POINT ARITHMETIC AND LOGIC

ENTRY DATA



ADD REGISTER AR 08, 07 DMC SECTION 114, ROUTINE 20

	ROS	REG 7	REG 8	A REG	B REG	C REG	D REG
①	395	66666666	22222222	641E	7080	2222	1A78
②	1E2	6666bbbb	22222222	6666	700E	2288	2222
③	1E4	bbbb8888	22222222	6444	8888	0088	8888
④	1EC	88888888	22222222	6424	8888	0026	4780

OBJECTIVES

THE HALFWORD SECOND OPERAND IS ADDED TO OR SUBTRACTED FROM THE FIRST OPERAND AND THE SUM IS PLACED IN THE FIRST OPERAND LOCATION. THE HFWD. 2ND OPND IS EXPANDED TO A FULL WORD BEFORE THE ADDITION OR SUBTRACTION BY PROPAGATING THE SIGN-BIT VALUE THROUGH THE 16 HIGH ORDER BITS

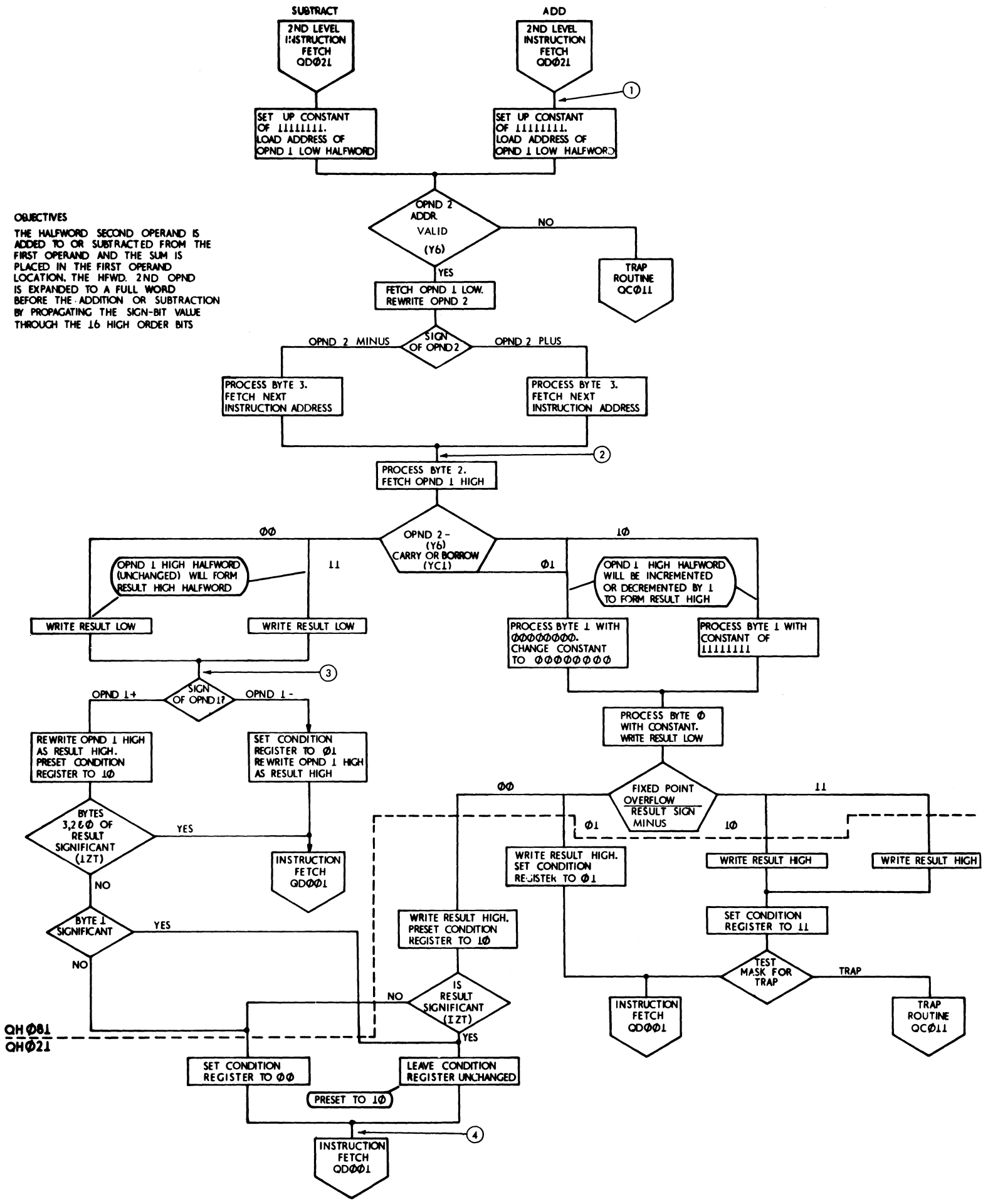
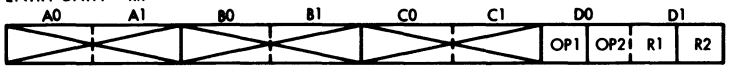
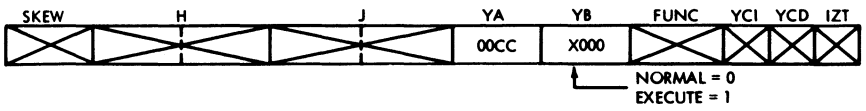


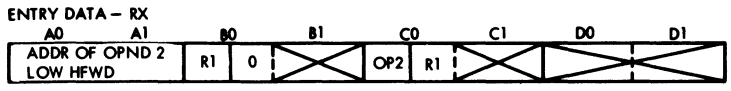
FIGURE 616.RX FIXED POINT ADD AND SUBTRACT

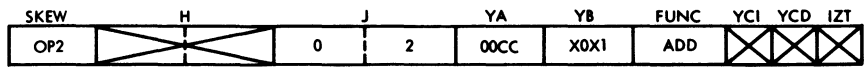
ENTRY DATA – RR



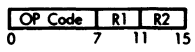


ENTRY DATA – RX

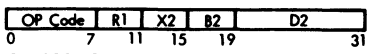




RR



RX



OBJECTIVE

TO COMPARE THE FIRST OPERAND TO THE SECOND OPERAND AND SET THE CONDITION CODE ACCORDING TO THE RESULT.

COMPARE DMC SECTION 105, ROUTINE 46 STOP ON MS 618C

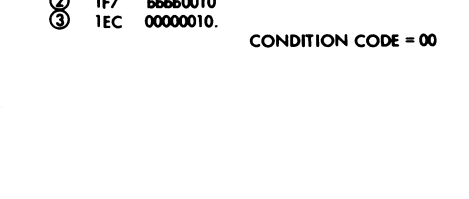
	ROS	REG 2	A REG	B REG	C REG	D REG
①	312	00000010	682E	2000	0000	0010
②	1F7	bbbb0010	682C	0000	0010	0000
③	1EC	00000010.	6820	0000	0022	4780

CONDITION CODE = 00

METHOD

COMPARISON IS FROM LEFT TO RIGHT, AS FOLLOWS:-

1 COMPARE SIGNS ;
IF DIFFERENT, SET CONDITION REGISTER ACCORDING TO SIGN OF OPND 1 (IF PLUS SET 10; IF MINUS SET 01)(TIME 3 CYCLES);
IF SAME, THEN
2 TEST HIGH 16 BITS OF OPND 1 FOR SIGNIFICANCE (i.e. FOR NON-ZERO IF PLUS AND FOR PRESENCE OF ANY ZERO BIT IF MINUS);
IF SIGNIFICANT, SET CONDITION REGISTER TO SIGN OF OPND 1 AND EXIT (TIME 4 CYCLES);
IF NOT SIGNIFICANT, THEN
3 COMPARE BYTES 2 (SEE DIAGRAM);
IF NOT EQUAL, SET CONDITION REGISTER ACCORDING TO THE INEQUALITY, AND EXIT (TIME 5 CYCLES);
IF EQUAL, THEN
4 COMPARE BYTES 3 AND SET THE CONDITION REGISTER ACCORDING TO RESULT (TIME 6 CYCLES).



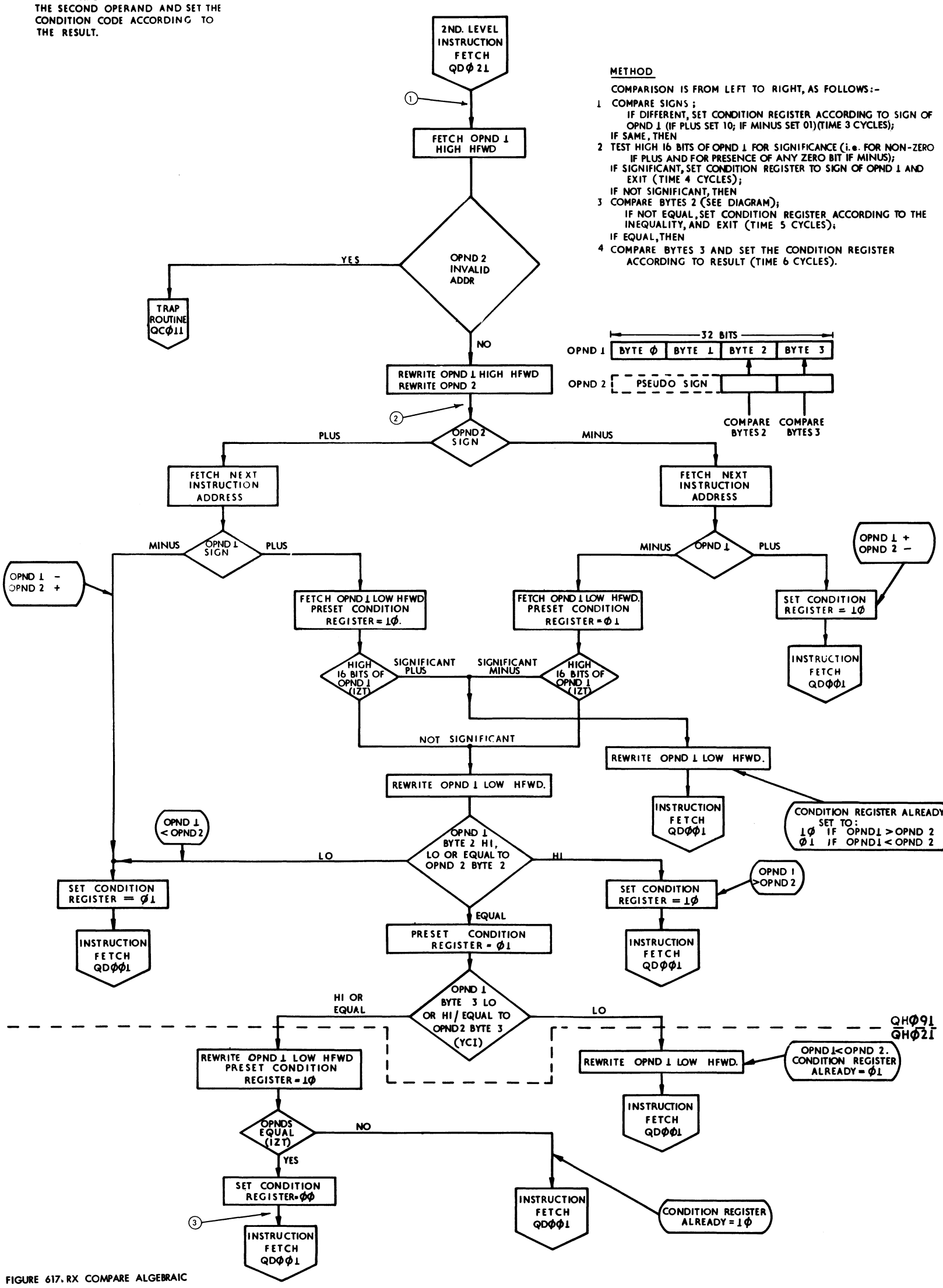
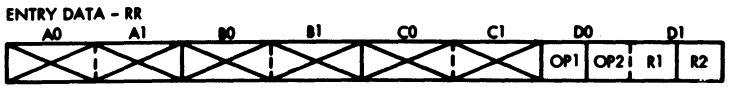
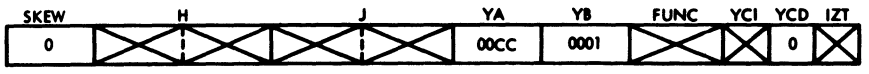


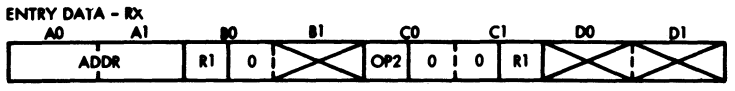
FIGURE 617. RX COMPARE ALGEBRAIC

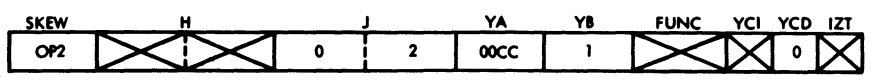
ENTRY DATA - RR



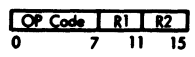


ENTRY DATA - RX

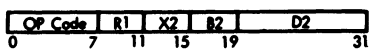




RR



RX



OBJECTIVE

THE UPDATED INSTRUCTION ADDRESS IN THE PSW IS REPLACED BY THE BRANCH ADDRESS IF THE STATE OF THE CONDITION CODE IS AS SPECIFIED BY THE 4 BIT MASK CARRIED IN THE R1 FIELD OF THE INSTRUCTION

BRANCH ON CONDITION BCR F, 05 CR BITS = 00 DMC SECTION 120, ROUTINE 10

	ROS	REG 0	REG 5	A REG	B REG	C REG	D REG
①	38E	00000002	00005D8C	1CD4	7037	1CD6	07F2
②	3F8	00000002	00005D8C	5228	A000	6228	07F8
③	3CE	00000002	00005D8C	1CD6	9000	1CD6	07F2
④	3D0	00000002	0000bbbb	6324	0000	631E	07D5

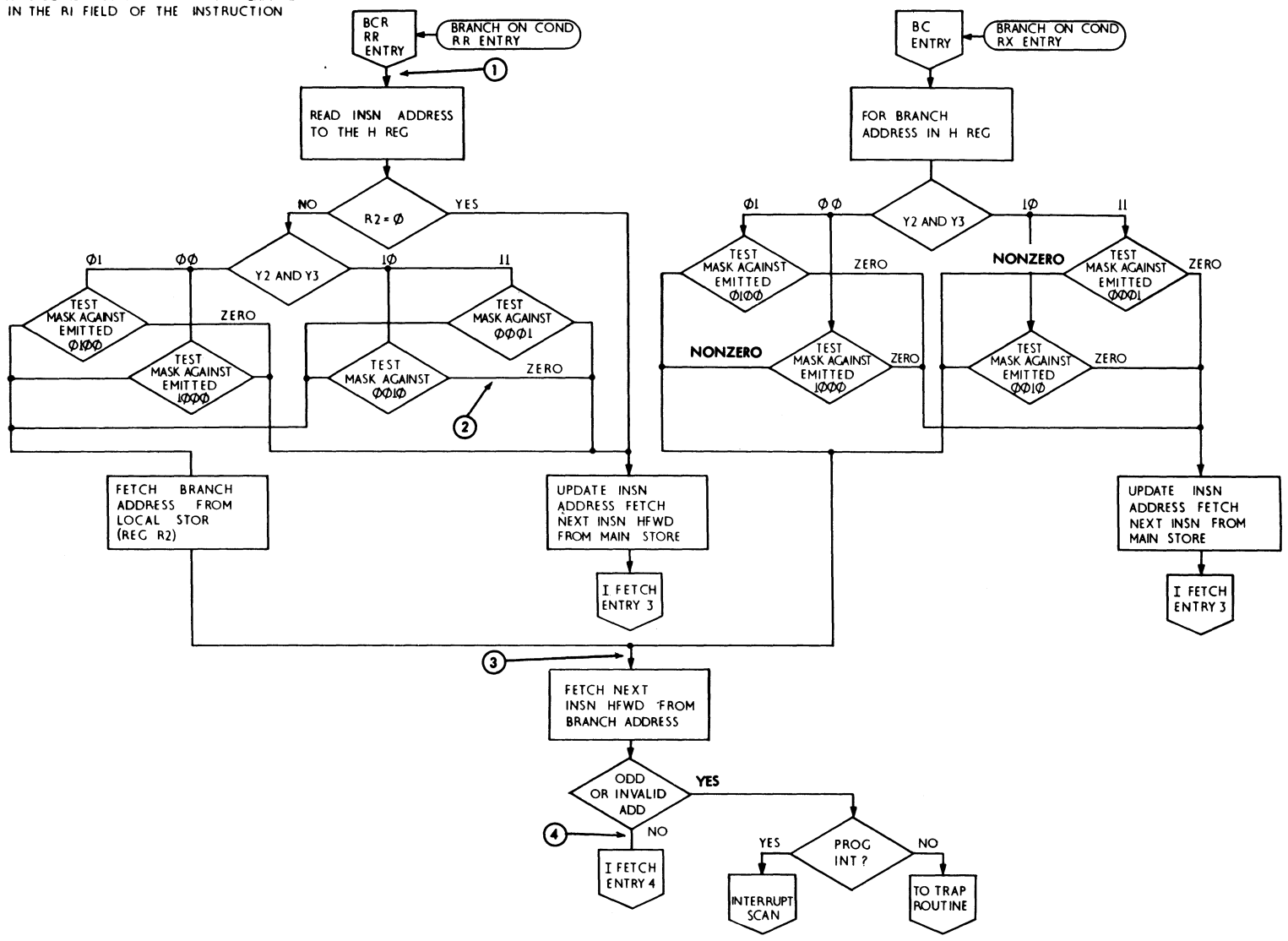


FIGURE 618. BRANCH ON CONDITION RR AND RX (QE031)

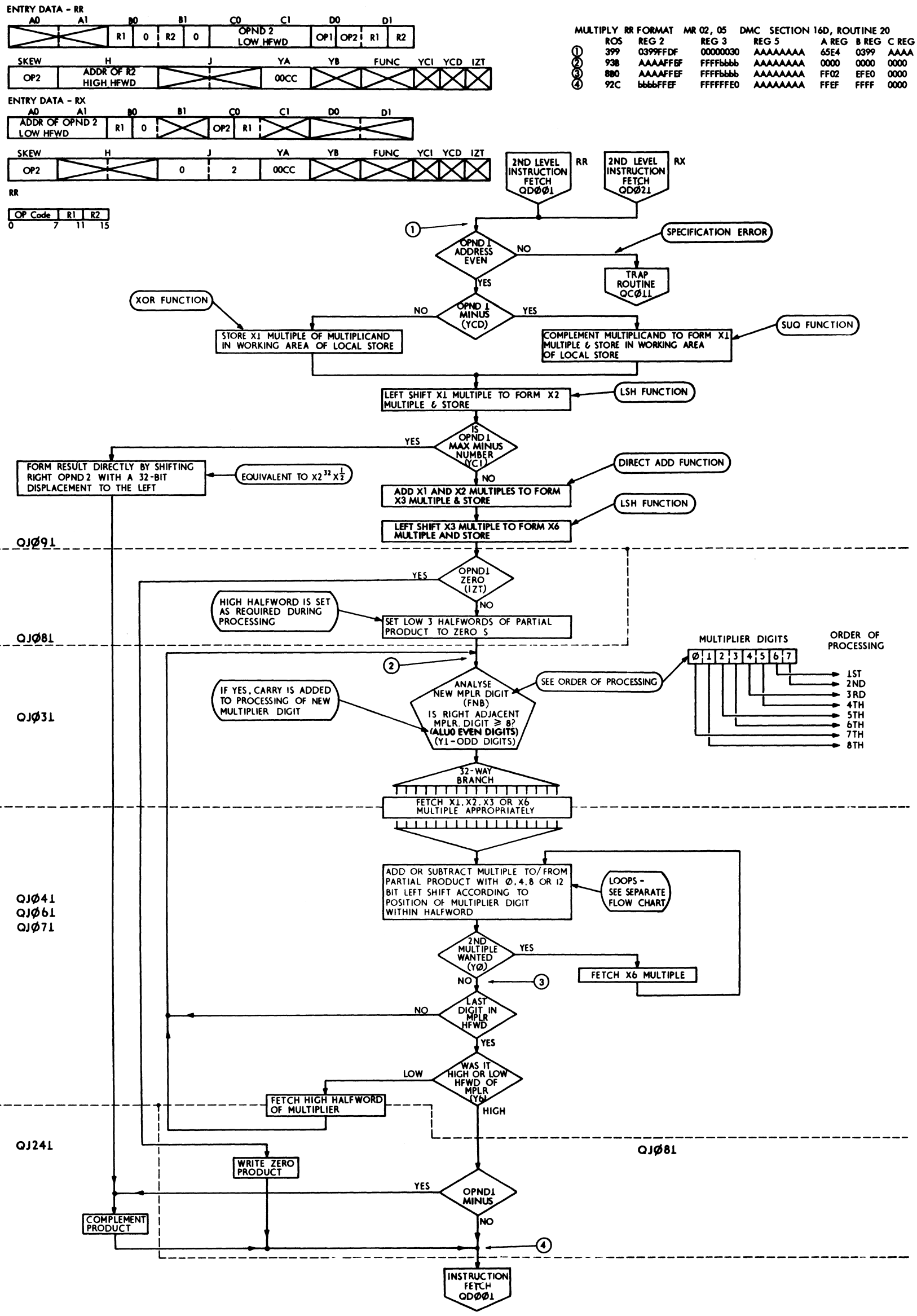
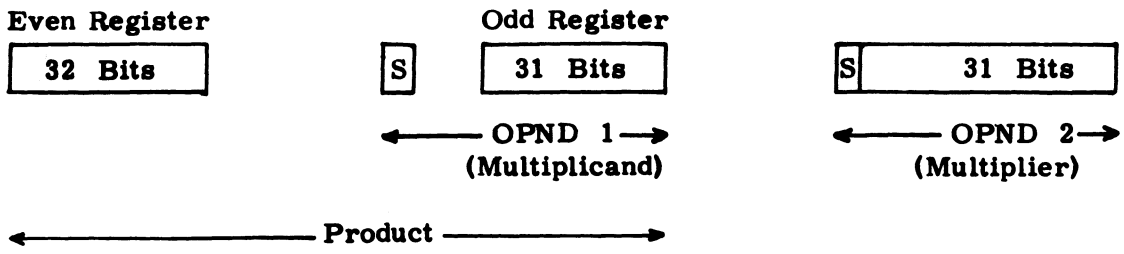


FIGURE 620. RR AND RX FIXED POINT MULTIPLY



The Multiplicand is located in the Odd Register of the Even/Odd pair specified by R1. The product is stored in the Register pair, overwriting the Multiplicand.

Method

1. The Multiplicand is complemented if it is negative.

2. X1, X2, X3 and X6 multiples of the Multiplicand are formed and held in Local Storage. Each multiple occupies 2 half-words and for X3 and X6 multiples the extended bits of the high order half-word are used to store the overflow bits.

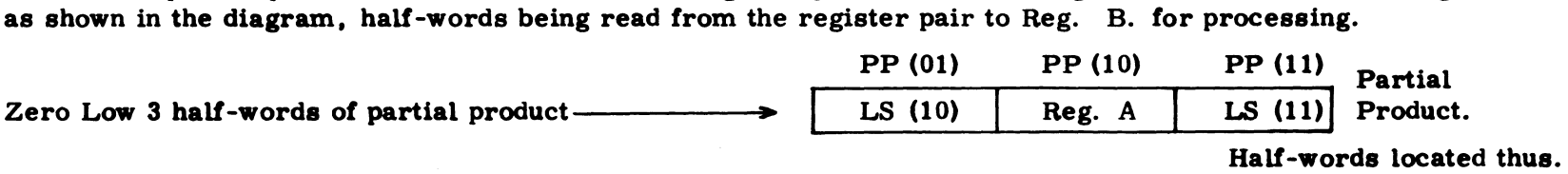
3. The low order half-word of the multiplier is read to D. Each hexadecimal digit of the multiplier is analysed using a function branch. Multiples are added to, or subtracted from a partial product according to the digit value as shown in the table.

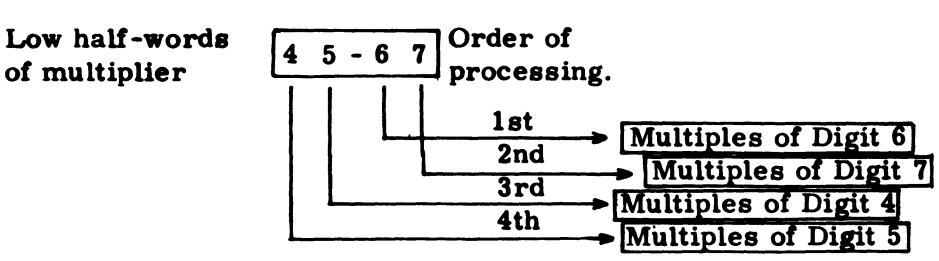
Multiplier Digit	Multiples Used: No Carry In	Multiples Used: Carry In	Multiplier Digit	Multiples Used: No Carry In	Multiples Used: Carry In
0	No Op.	+1	8	-2-6 (+16)	-1-6 (+16)
1	+1	+2	9	-1-6 (+16)	-6 (+16)
2	+2	+3	A	-6 (+16)	+1-6 (+16)
3	+3	-2+6	B	+1-6 (+16)	+2-6 (+16)
4	-2+6	-1+6	C	+2-6 (+16)	-3 (+16)
5	-1+6	+6	D	-3 (+16)	-2 (+16)
6	+6	+1+6	E	-2 (+16)	-1 (+16)
7	+1+6	+2+6	F	-1 (+16)	(+16)

The (+ 16) for multiplier digits which are greater than or equal to 8 is dealt with by forcing a carry into the next most significant digit position. Multiples are added, or subtracted form Skew and byte offset appropriate to the position of the multiplier digit within the half-word.

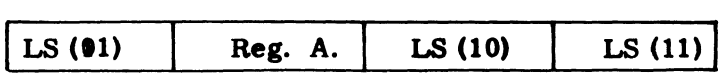
Register H addresses the multiples in Local Storage and during the Add/Subtract loop each half-word of a multiple is read to Register C. Register J addresses the Even/Odd register pair. The partial product is stored as shown in the diagram, half-words being read from the register pair to Reg. B.

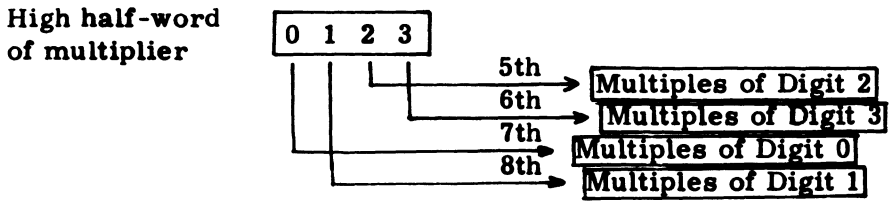
The partial product is stored in the result register pair in Local Storage and in the Data Flow Reg. A, as shown in the diagram, half-words being read from the register pair to Reg. B. for processing.





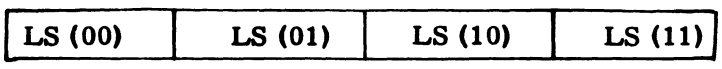
Multiples are added or subtracted with Skew and byte shift appropriate to postion of multiplier digit.





The numbers in brackets afterr PP and Local Storage e. g. (01)are binary and refer to the low order two address bits.

This forms final product correctly in the Even/odd pairs of result register in Local Storage.



4. The process is repeated for the High Order multiplier half-word. The sign bit of the multiplier is treated as a Data bit, so that the correct two's complement form of a negative product is produced automatically.

5. The result is complemented if the original multiplicand was negative.

FIGURE 621. RR AND RX FIXED POINT MULTIPLY, NOTES.

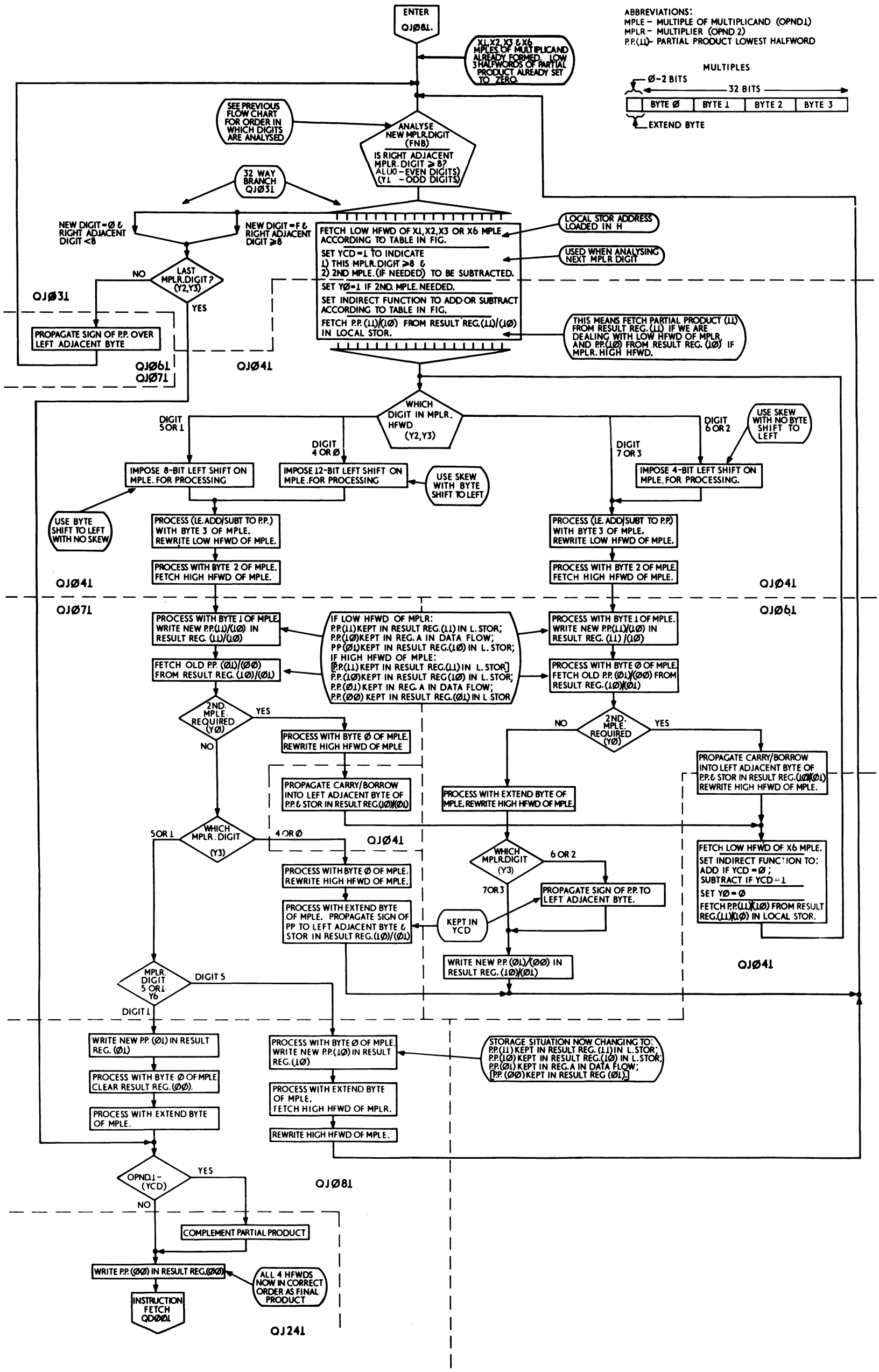
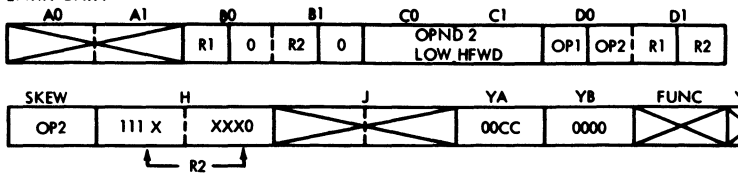
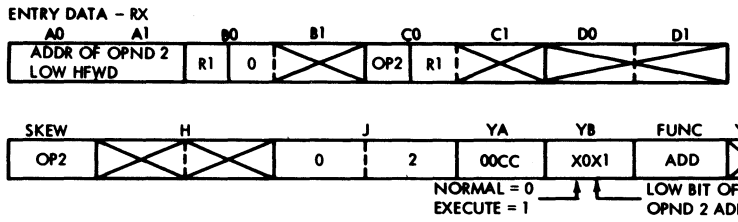


FIGURE 622. RR AND RX FIXED POINT MULTIPLY, DETAIL OF LOOPS

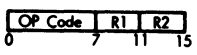
ENTRY DATA – RR



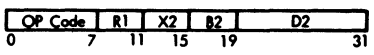
ENTRY DATA – RX



RR



RX



FIGURES 623 AND 624 DR 8,0 DMC Section 170 ROUTINE 20

	ROS	REG 0	REG 8	REG 9	A REG	B REG	C REG	D REG	CAS
①	39B	80000000	C0000000	80000000	657C	8000	0000	1D80	QJ241
②	973	80000000	3FFFbbbb	80000000	0000	0000	8000	FFFF	QJ271
③	9CD	80000000	bbbbbbbb	00000000	0000	0000	FFFF	FFF8	QJ221
④	9F0	80000000	bbbbbbbb	7FFFbbbb	FFFF	5555	0000	0000	QJ211
⑤	8DA	80000000	bbbb0000	7FFFFFFF	0000	0000	0000	0000	QJ211

OBJECTIVES
FOR FIXED POINT DIVISION THE DOUBLE LENGTH DIVIDEND (OPERAND 1) IS DIVIDED BY THE DIVISOR (OPERAND 2), THE QUOTIENT AND REMAINDER REPLACING OPERAND 1.
IN THE INITIALISATION MICROPROGRAM BOTH OPERANDS ARE MADE POSITIVE. X1, X2, X4 AND X8 MULTIPLES OF THE DIVISOR ARE FORMED IN LOCAL STORAGE. THE DIVIDE LOOP IS ENTERED AFTER A TRIAL SUBTRACTION FOR DIVIDE CHECK.

RR DIVIDE RX DIVIDE

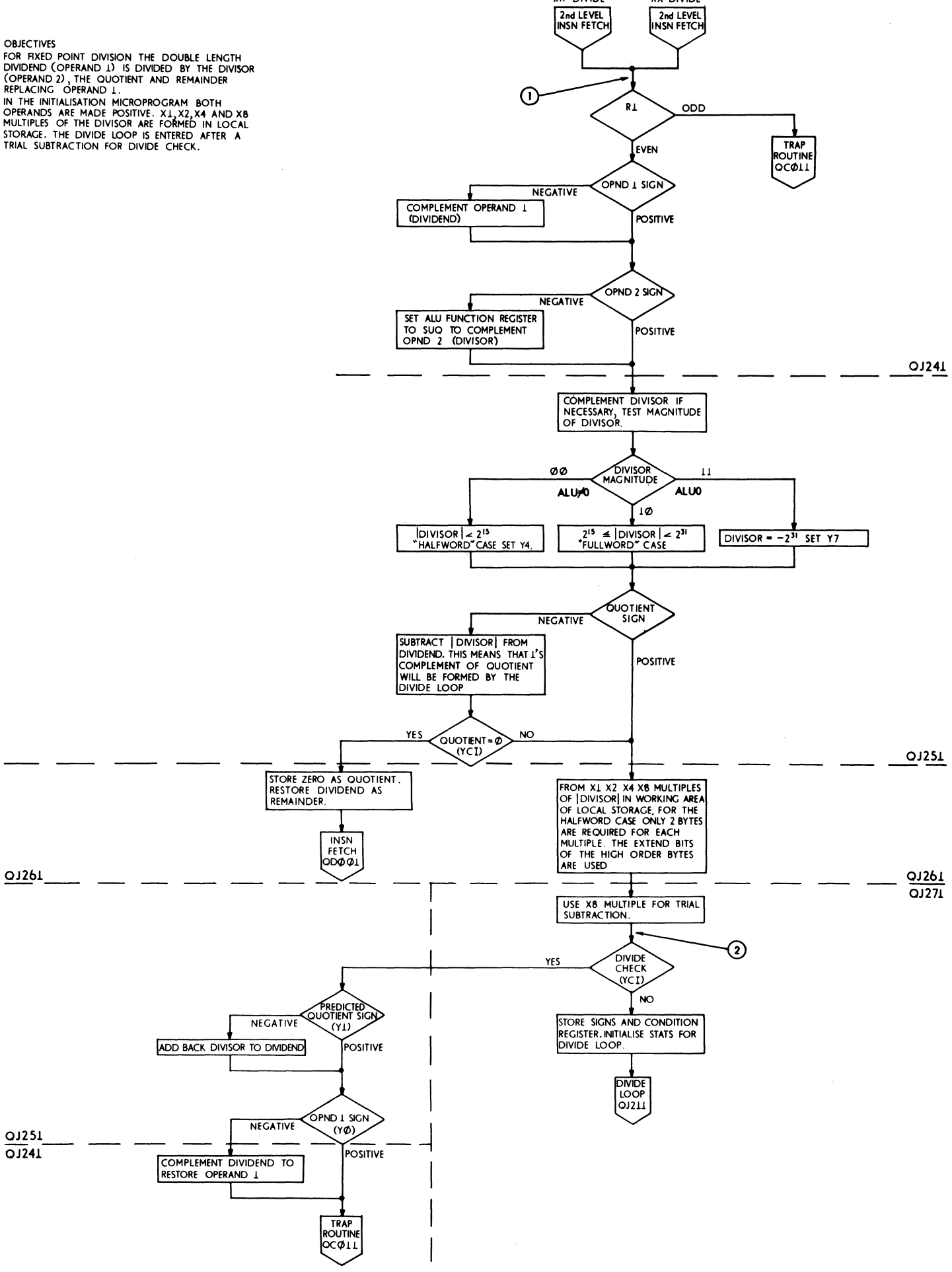


FIGURE 623 RR, RX FIXED POINT DIVIDE INITIALIZATION

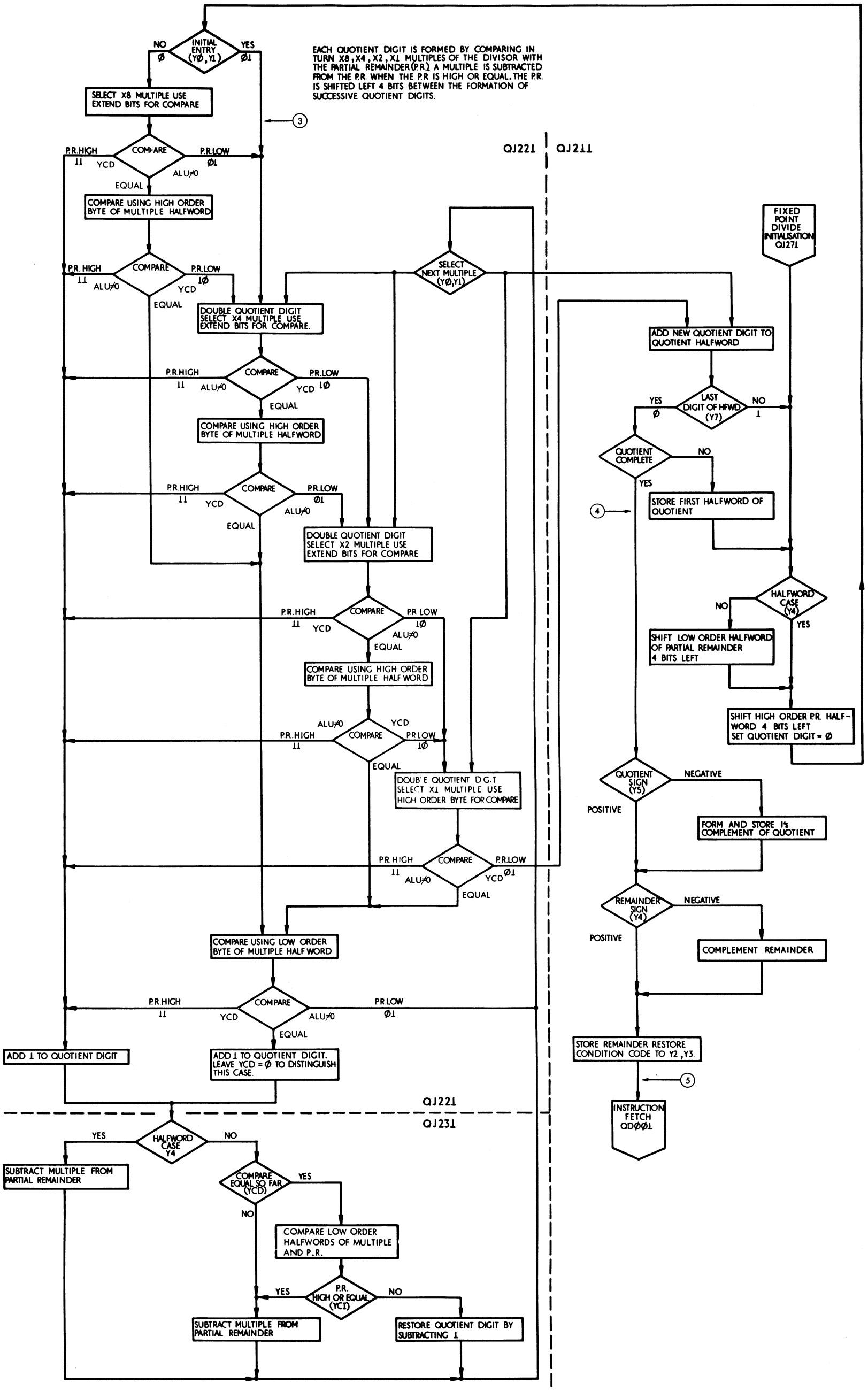
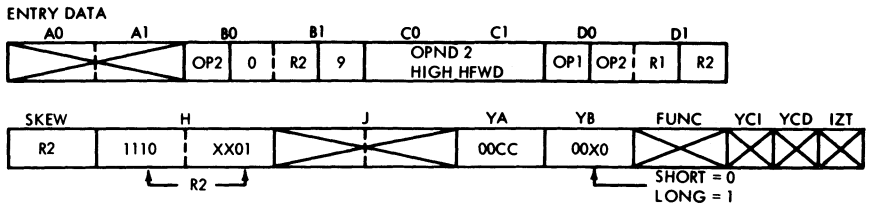
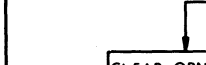


FIGURE 624. FIXED POINT DIVIDE LOOP

ENTRY DATA



RR



LOAD POSITIVE, FLOATING POINT LPER DMC SECTION 19C, ROUTINE 01, STOP ON MS 60FA.

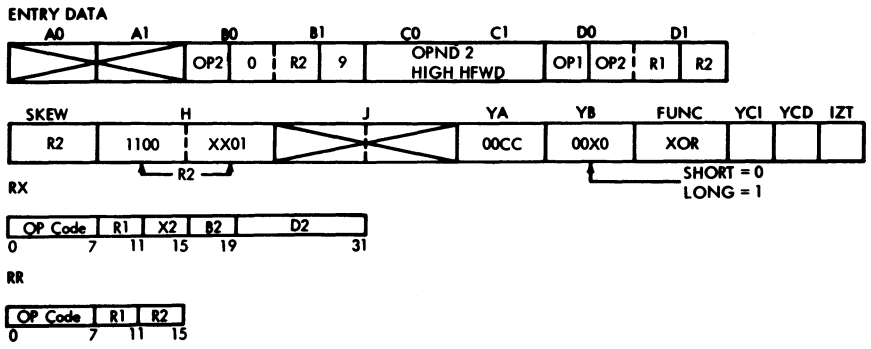
	ROS	FP REG 00	FP REG 06	A REG	B REG	C REG	D REG
①	A45	F0F0F0F0	0F1E2D3C	60FA	0069	60FC	3006
②	A2C	F0F0F0F0	0F1Ebbbb	8FFA	2D3C	0F1E	3006
③	BF6	bbbbF0F0	0F1E2D3C	00FA	2D3C	0F1E	3006
④	BEF	0F1E2D3C	0F1E2D3C	60FC	2D3C	0FFE	47C0

OBJECTIVE

TO PERFORM THE RR FLOATING POINT INSTRUCTIONS LOAD, HALVE LOAD COMPLEMENT, LOAD AND TEST, LOAD NEGATIVE AND LOAD POSITIVE (FOR DESCRIPTI ↓ OF OBJECTIVES OF EACH INSTRUTION SEE PRINCIPLES OF OPERATIONS MANUAL)

FIGURE 625. RR FLOATING POINT SIGN OPERATIONS

ENTRY DATA



NO EXAMPLE SHOWN.

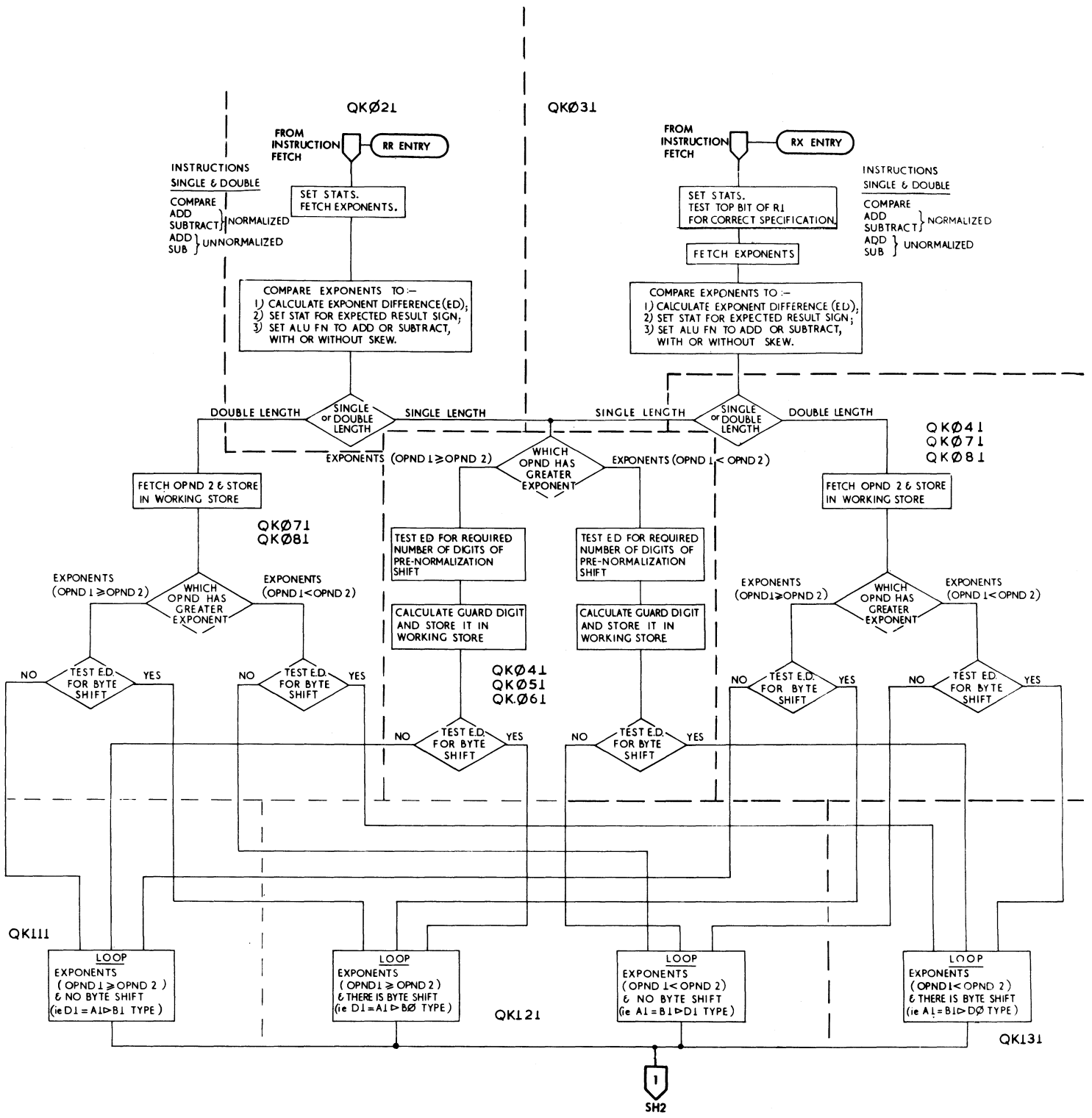


FIGURE 626. FLOATING POINT RR AND RX (QK021, QK171) (SHEET 1 OF 2)

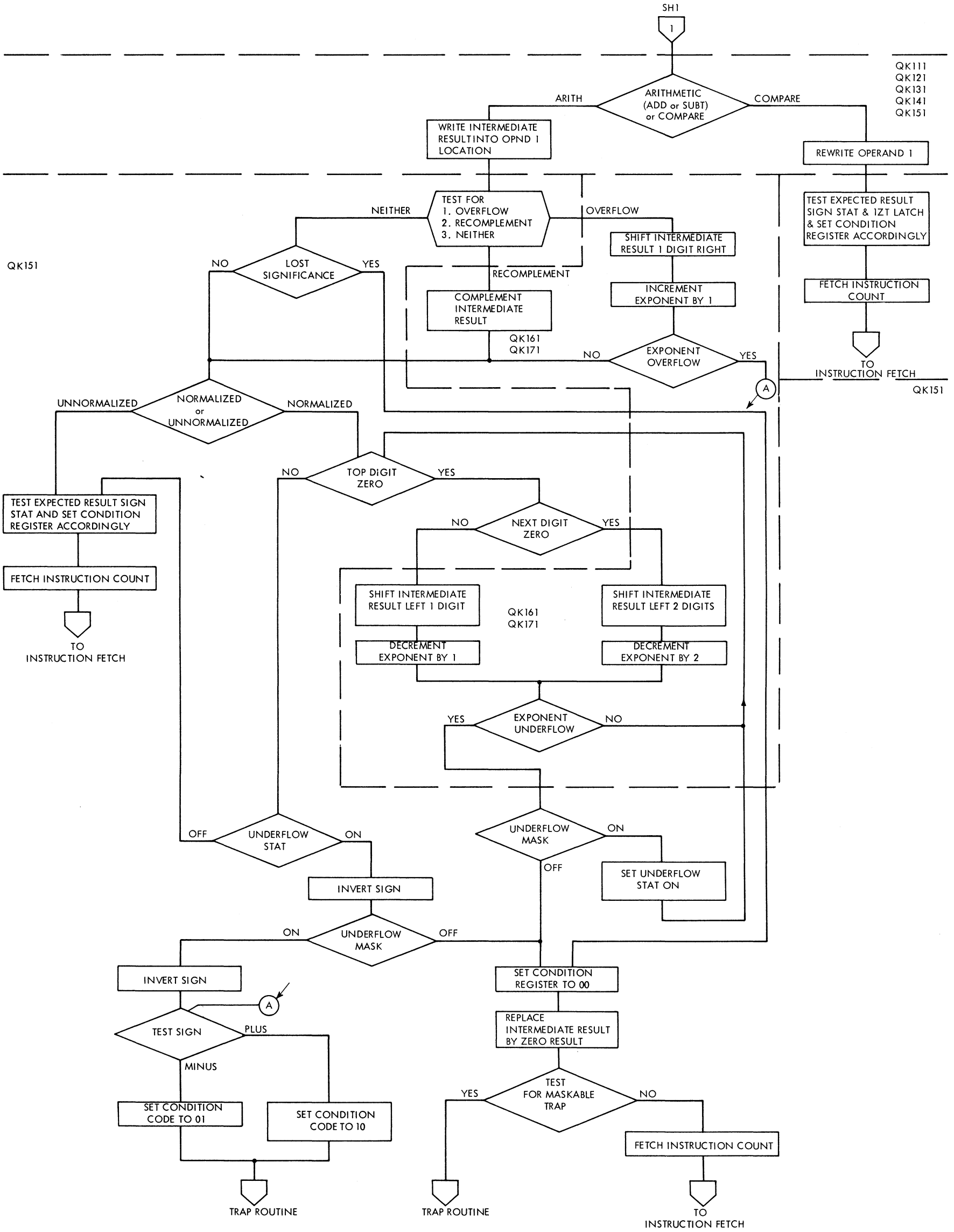
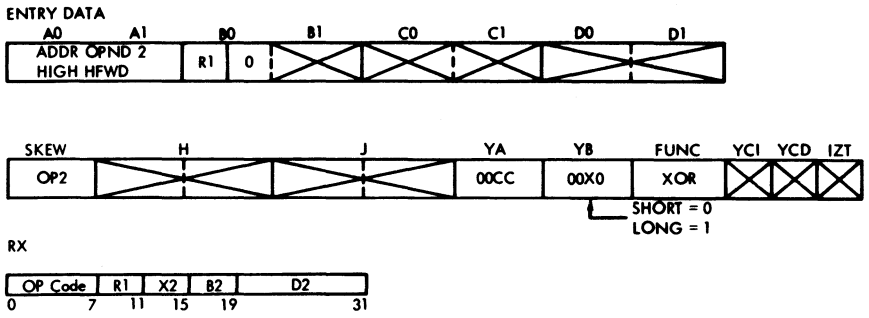


FIGURE 626. FLOATING POINT RR AND RX (QK021, QK171((SHEET 2 OF 2)

ENTRY DATA



LOAD FLOATING POINT SHORT DMC SECTION 195, ROUTINE 03, STOP ON MS 6158

	ROS	FP REG 00	A REG	B REG	C REG	D REG
①	A11	AB687B1B4321DCBA	6E58	2000	0082	5555
②	BF9	AB687B1B4321DCBA	6E58	0000	5555	5555
③	174	AB687B1B4321DCBA	6E58	0000	5555	5555
④	1F5	AB687B1B4321DCBA	0939	00F0	0000	5AF0

OBJECTIVES

STORE FLOATING POINT
THE FIRST OPND LOCATED IN THE FLTG POINT REG SPECIFIED BY R1 IS STORED AT THE EFFECTIVE ADDRS GENERATED BY X2, D2 AND B2. THE STORED OPND AND THE CONDITION CODE REMAIN UNCHANGED
LOAD FLOATING POINT
THE SECOND OPND WHICH IS LOCATED IN MAIN STORE AT THE EFFECTIVE ADDRESS GENERATED BY X2, B2 AND D2 IS PLACED IN THE FIRST OPND POSITION WHICH IS THE FLTG. POINT REG. SPECIFIED BY R1. THE LOADED OPND AND THE CONDITION CODE REMAIN UNCHANGED

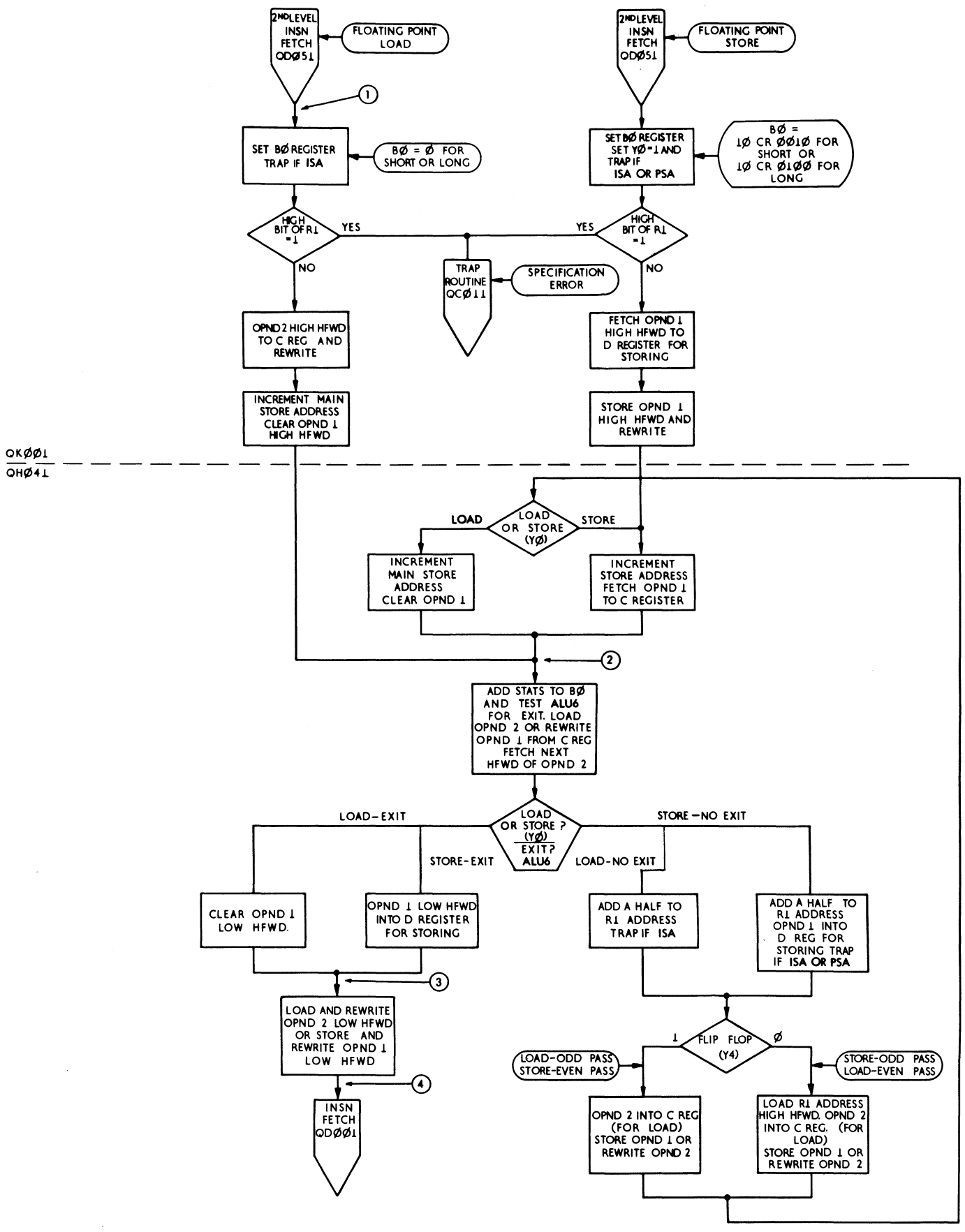
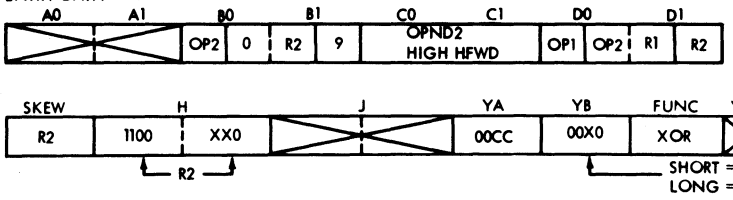
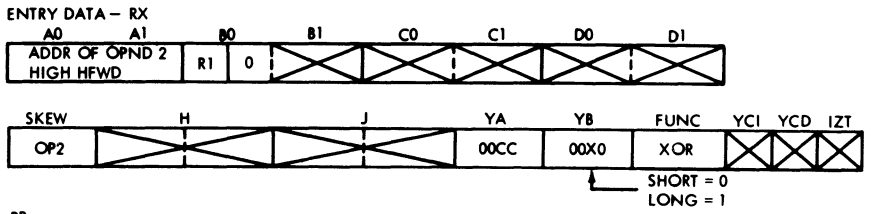


FIGURE 627. FLOATING POINT LOAD AND STORE (SHORT AND LONG)

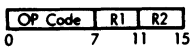
ENTRY DATA – RR



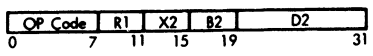
ENTRY DATA – RX



RR



RX



FLOATING POINT MULTIPLY, SHORT NMER DMC SECTION 1CC, ROUTINE 01, STOP ON MS 60F2. THIS OPERATION WILL USE FIGURES 628 AND 629.

	ROS	FP REG 00	FP REG 02	A REG	B REG	C REG	D REG
①	A18	45100000	45100000	60F2	C009	4510	3C20
②	AA6	45100000	45100000	60F4	2040	4510	0000
③	A87	45100000	45100000	60F4	2045	4510	4510
④	AD0	45100000	45100000	4AF4	0010	0020	200C
⑤	AF5	45100000	45100000	4AF4	0010	0020	200C
FIGURE 629							
⑥	8B7	45100000	45100000	0000	0060	4A10	0000
⑦	8F7	45100000	4510bbbb	0000	0000	4510	0000
⑧	92F	45100000	4510bbbb	0000	0000	4510	0000
⑨	930	45100000	4510bbbb	0000	0000	0000	0000
⑩	93E	45100000	bbbbbbbb	0000	0000	0000	4500
⑪	84C	45100000	bbbbbbbb	0000	0000	0010	4500
⑫	8CE	45100000	bbbb0000	0100	0000	0010	4500
⑬	9AE	45100000	bbbb0000	60F4	4910	0100	0000

OBJECTIVE
TO SHOW THE INITIALIZATION FOR FLOATING POINT DIVIDE/MULTIPLY

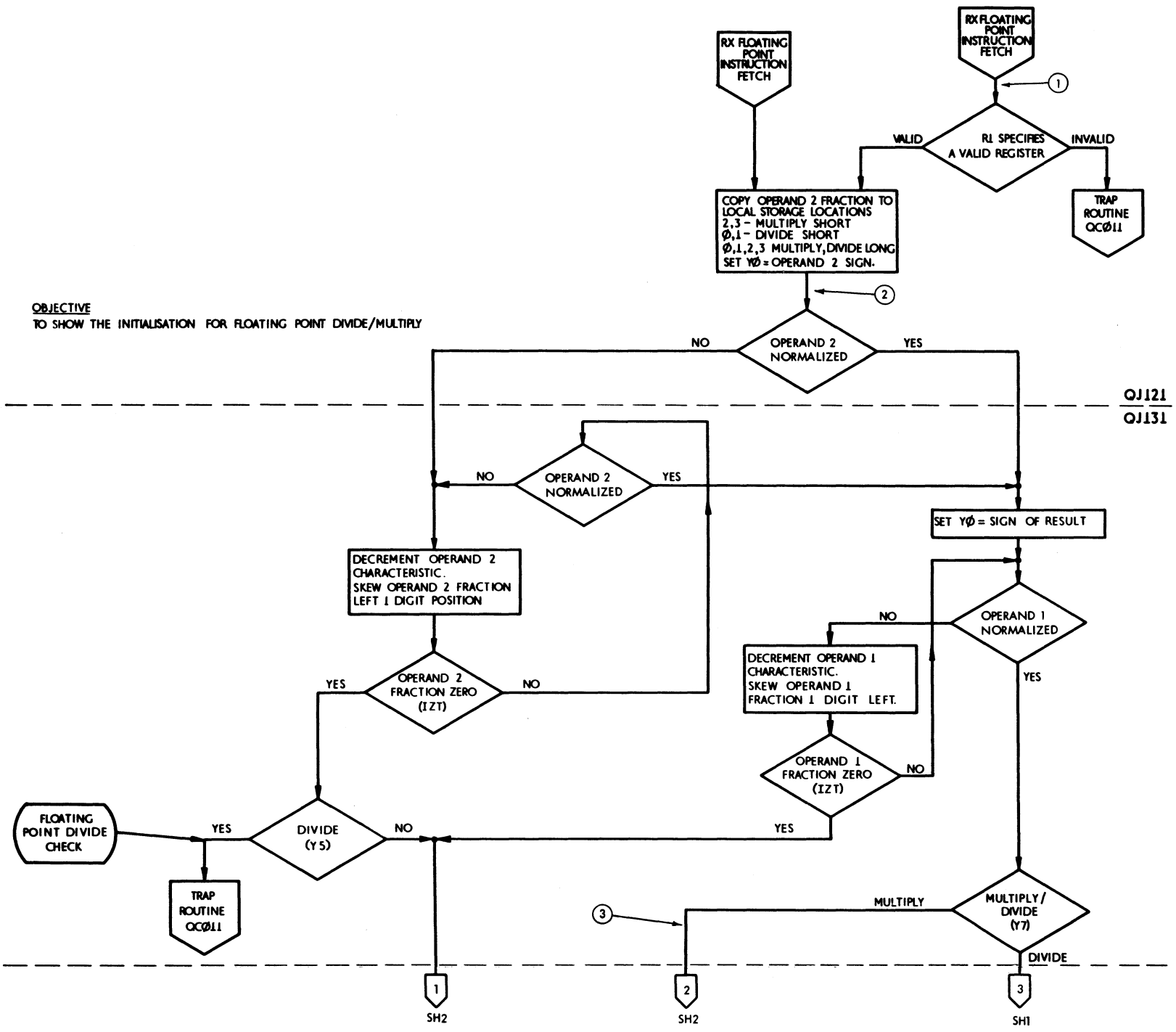


FIGURE 628. FLOATING POINT MULTIPLY/DIVIDE INITIALIZATION (SHEET 1 OF 2)

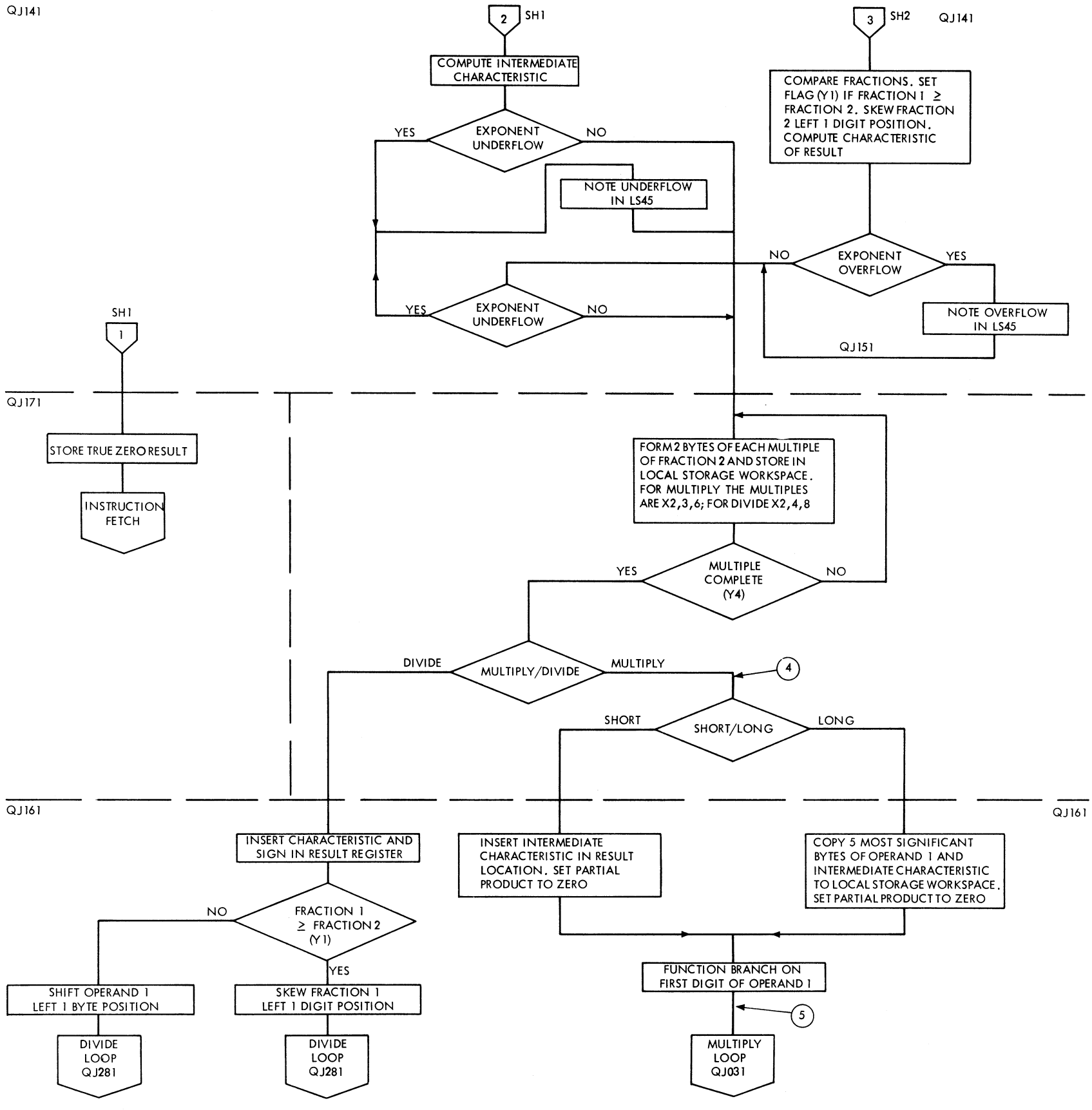


FIGURE 628. FLOATING POINT MULTIPLY/DIVIDE INITIALIZATION (SHEET 2 OF 2)

THE FRACTION OF THE FLOATING-POINT OPERANDS ARE MULTIPLIED USING THE SAME ALGORITHM AS FOR FIXED POINT MULTIPLY. THE MICROPROGRAM IS WIDELY SHARED; STAT Y4 IS NORMALLY SET FOR FLOATING-POINT.

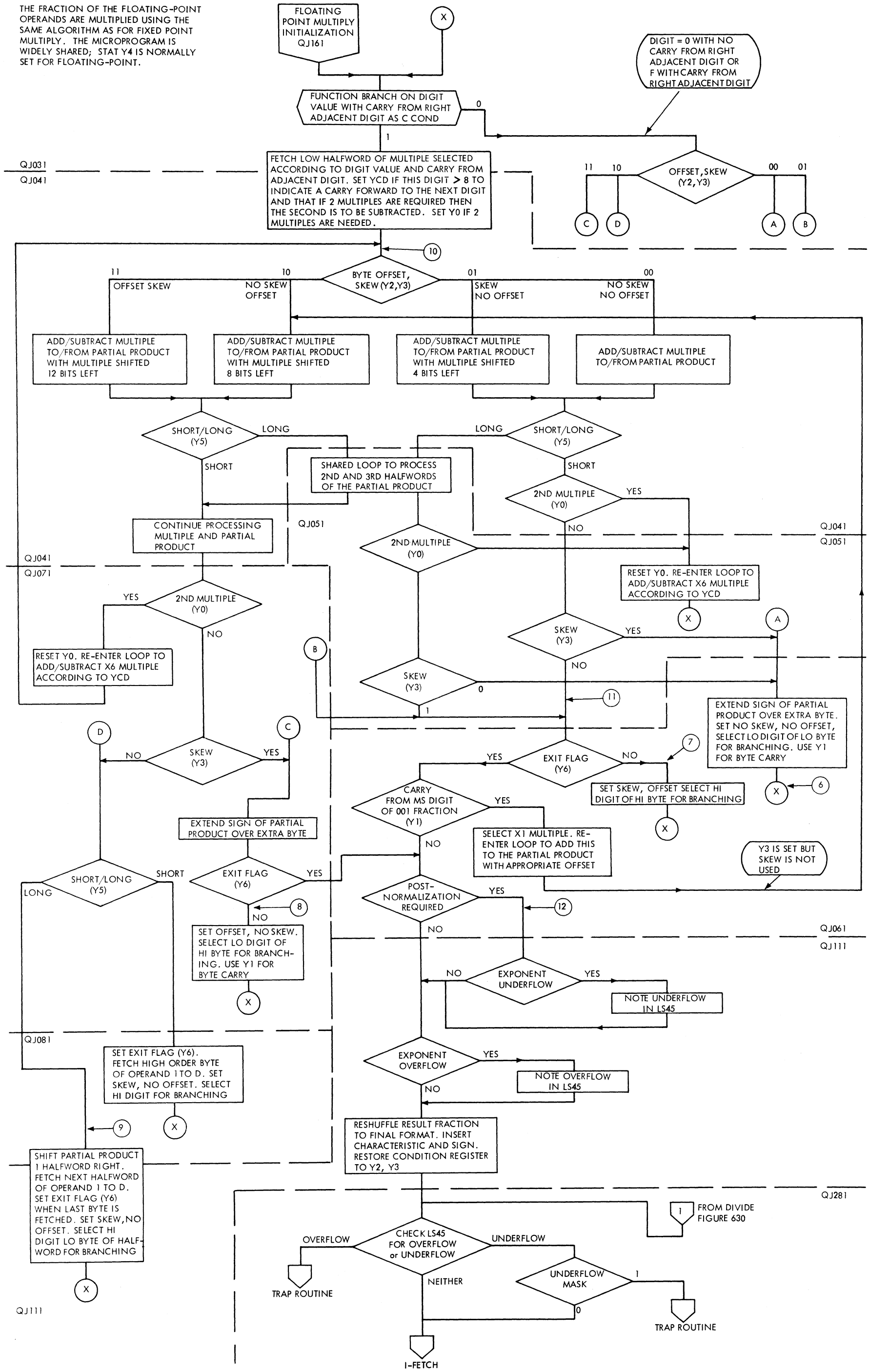


FIGURE 629. FLOATING POINT MULTIPLY LOOP

FLOATING POINT DIVIDE, SHORT DER DMC SECTION 1C6, ROUTINE 09, STOP ON MS 6238

	ROS	FP REG 02	FP REG 06	A REG	B REG	C REG	D REG
(1)	A56	45150F75	42345000	1A28	4700	150F	7500
(2)	9D6	45150F75	42345000	1A28	4700	150F	7500
(3)	986	45150F75	42345000	0000	4701	150F	7500
(4)	9C9	45150F75	42345000	0D14	4701	15FB	7500
(5)	9E3	45150F75	42345000	068A	4702	07FB	7500
(6)	9D1	45150F75	42345000	068A	4703	0171	7500
(7)	9ED	bbbb0F75	42345000	0645	4515	0171	7500
(8)	9EF	bbbb0F75	42345000	6745	4515	0034	5000
(9)	A6F	45671000	42345000	0045	1000	0000	0000

EACH DIGIT OF THE QUOTIENT FRACTION IS FORMED BY COMPARING PARTIAL REMAINDER IN TURN WITH THE X8, X4, X2, and X1 MULTIPLES OF THE DIVIDED. WHEN A MULTIPLE IS LESS THAN OR EQUALS THE PARTIAL REMAINDER IT IS SUBTRACTED BETWEEN THE FORMATION OF SUCCESSIVE QUOTIENT DIGITS. THE PARTIAL REMAINDER IS SKEWED LEFT ONE DIGIT POSITION.

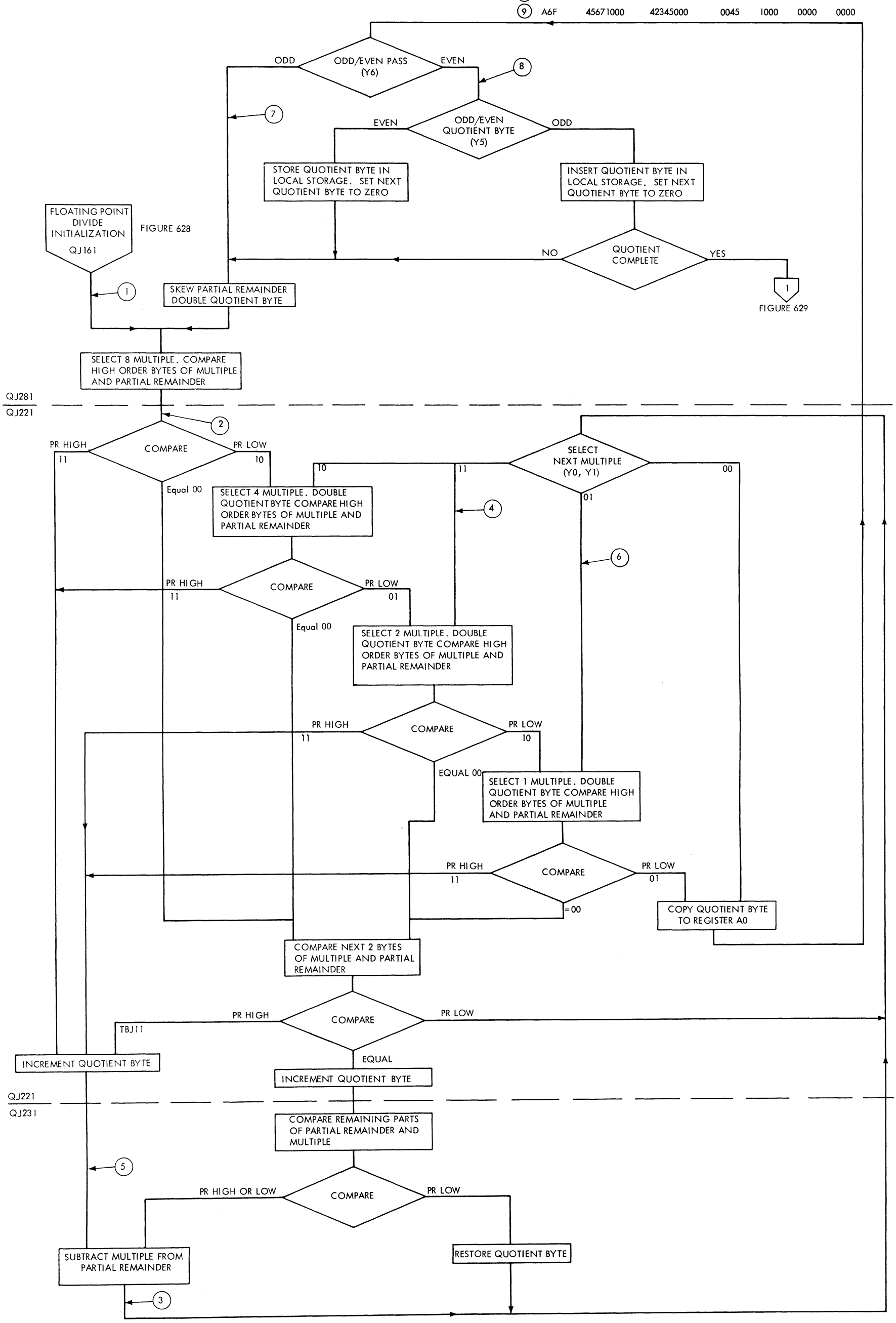
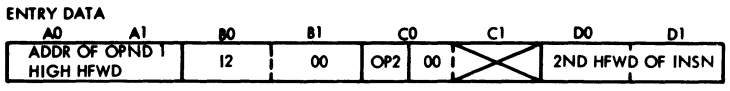
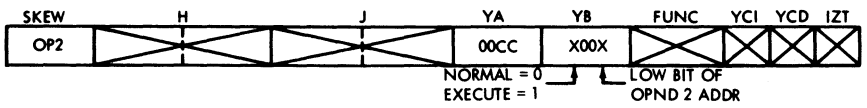


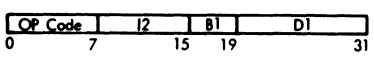
FIGURE 630. FLOATING POINT DIVIDE LOOP

ENTRY DATA





SI



TEST UNDER MASK

	ROS	REG 6	REG 7
①	343	00FFFFFF	FFFFFFFF
②	36D	00FFFFFF	00005D8C
③	374	00FFFFFF	00006000

DMC SECTION 134, ROUTINE 06

A REG	B REG	C REG	D REG
01A2	0100	1001	0000
01CB	0400	0401	0000
4E24	0202	0200	F2FF

OBJECTIVES

THE STATE OF THE FIRST OPERAND BITS SELECTED BY A MASK (IMMED. DATA I2) IS USED TO SELECT A CONDITION CODE.

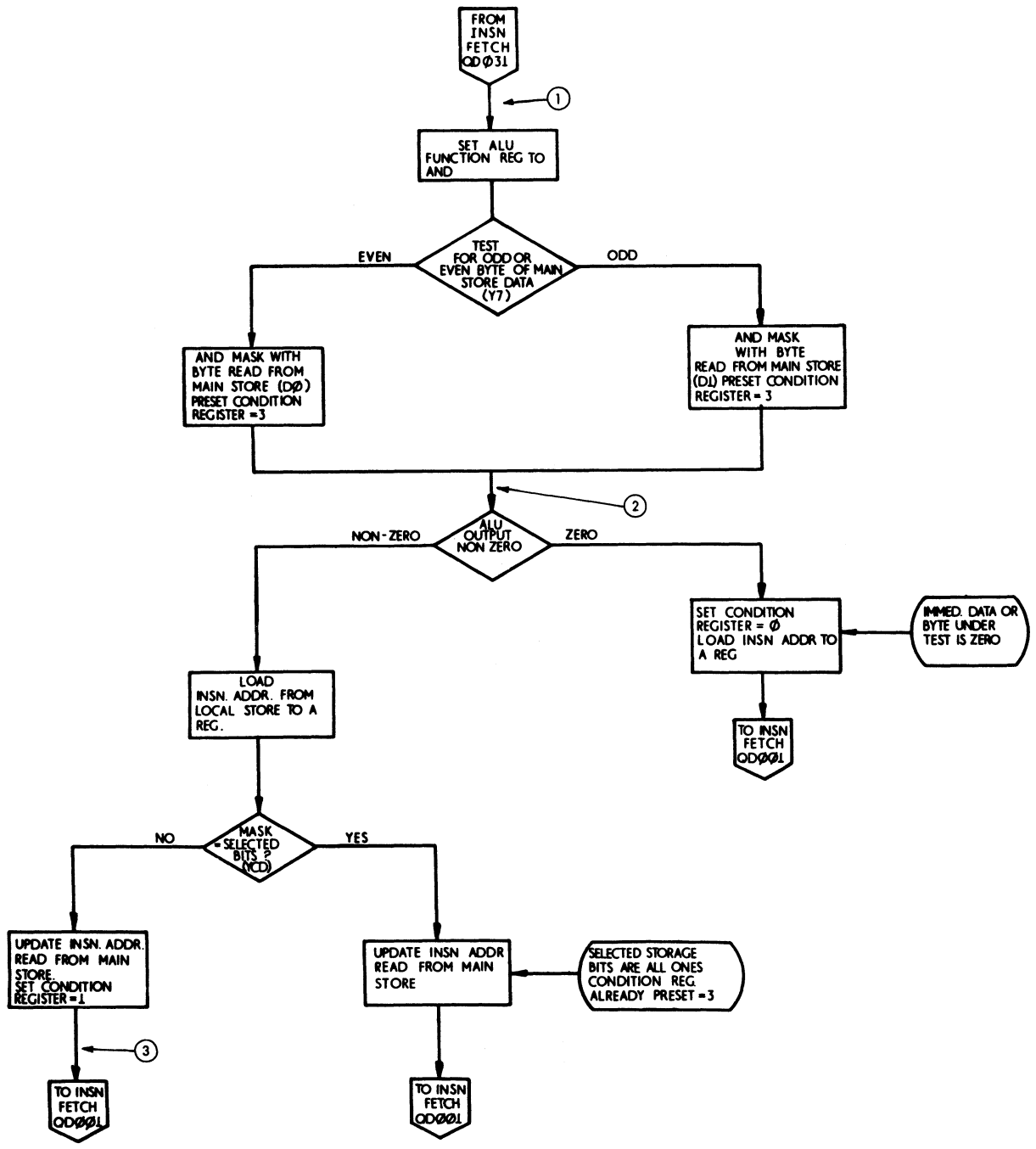
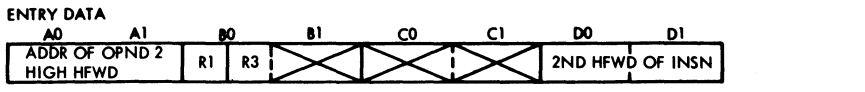
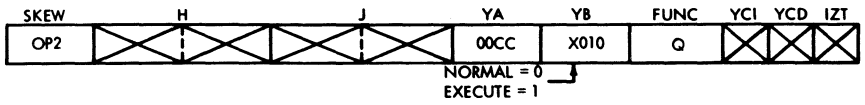


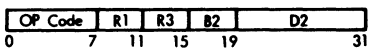
FIGURE 631 .TEST UNDER MASK (QP001)

ENTRY DATA





RS



BRANCH ON INDEX HIGH BXH 06, 08 DMC SECTION 172, ROUTINE 08

	ROS	REG 6	REG 8	A REG	B REG	C REG	D REG
①	34C	7FFFFFF0	00000001	62CE	6800	6602	F2CE
②	3E5	7FFFbbbb	00000001	62CE	0F20	0600	FFF1
③	3CB	7FFFFFF1	00000001	62CE	0000	6000	7FFF
④	3EC	80000000	800049AC	4E24	3000	3026	4780

OBJECTIVES

TO ADD THE 1ST OPND (R1) TO THE 2ND OPND (R3) AND COMPARE THE RESULT WITH THE 3RD OPND. (R3 OR R3 + 1). DEPENDENT ON THE OPERATION (BRANCH ON INDEX HIGH OR BRANCH ON INDEX LOW OR EQUAL) THE RESULT OF THE COMPARE WILL CAUSE EITHER THE BRANCH ADDRESS (B2 + D2) OR THE INSTRUCTION COUNT TO CALL THE NEXT INSN.

1ST OPND = R1 (CONTENTS OF)
2ND OPND = R3 (CONTENTS OF)
3RD OPND = R3 IF R3 IS ODD (CONTENTS)
OR R3 + 1 IF R3 IS EVEN
BRANCH ADD = B2 + D2

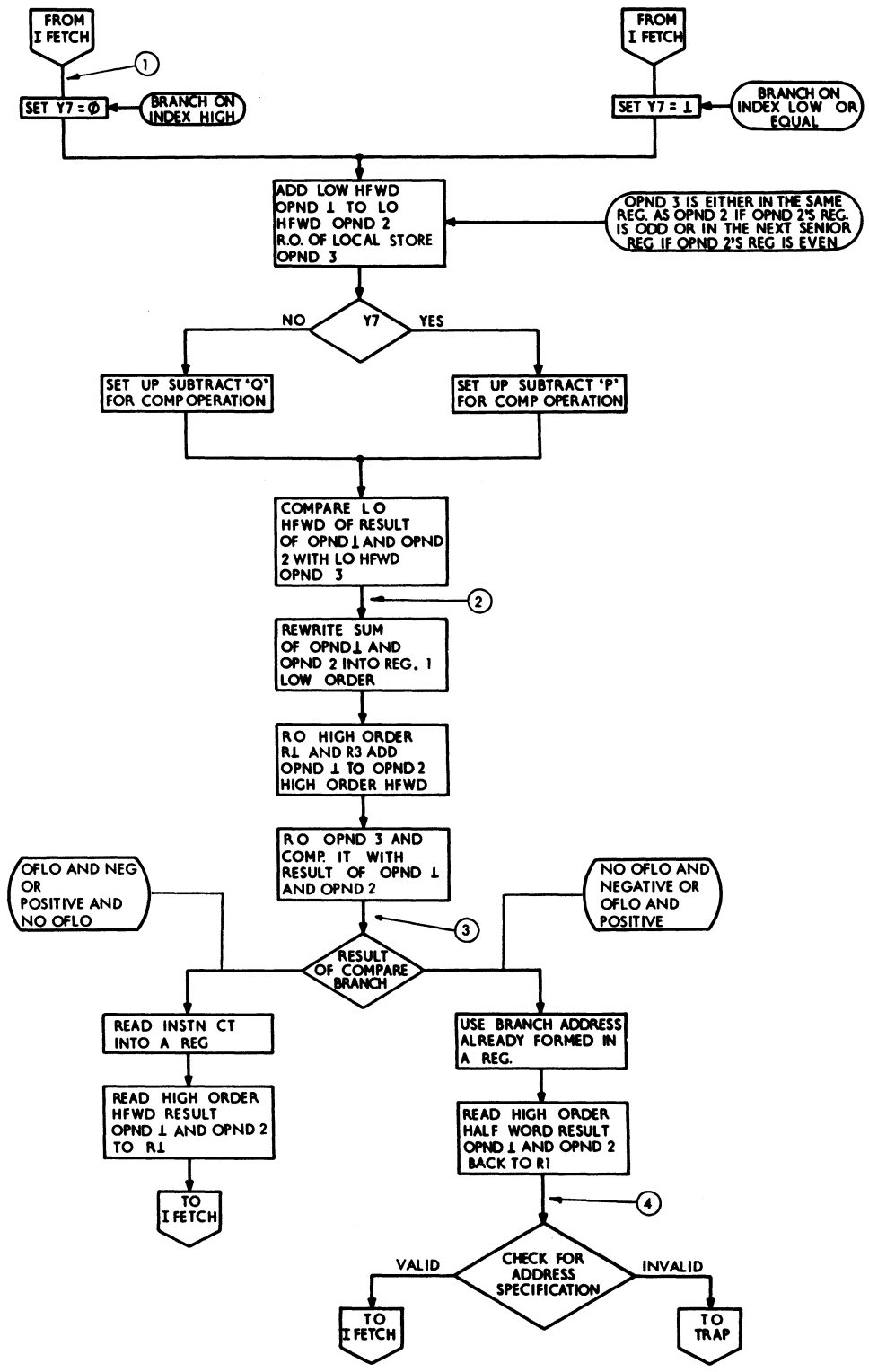
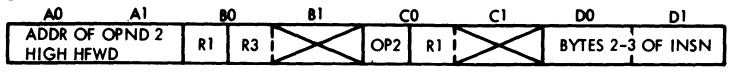
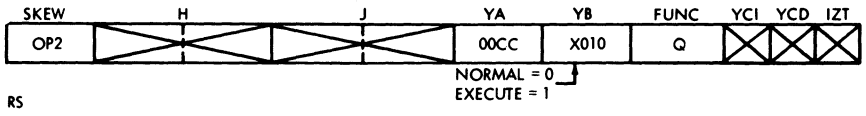


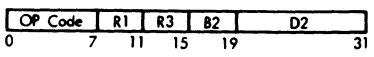
FIGURE 632. BRANCH ON INDEX (QE041)

ENTRY DATA





RS



NO EXAMPLE SHOWN.

Y5 = SET SYSTEM MASK
Y8 = ID = INHIBIT DUMP
Y9 = MI = MASKABLE INTERRUPT

OBJECTIVE
TO ALTER SYSTEM MASK PSW BITS Ø→7
1 READ SECOND OPERAND FROM MAIN STORAGE
2 PLACE CONTENTS OF 2ND OPND INTO PSW BITS Ø→7

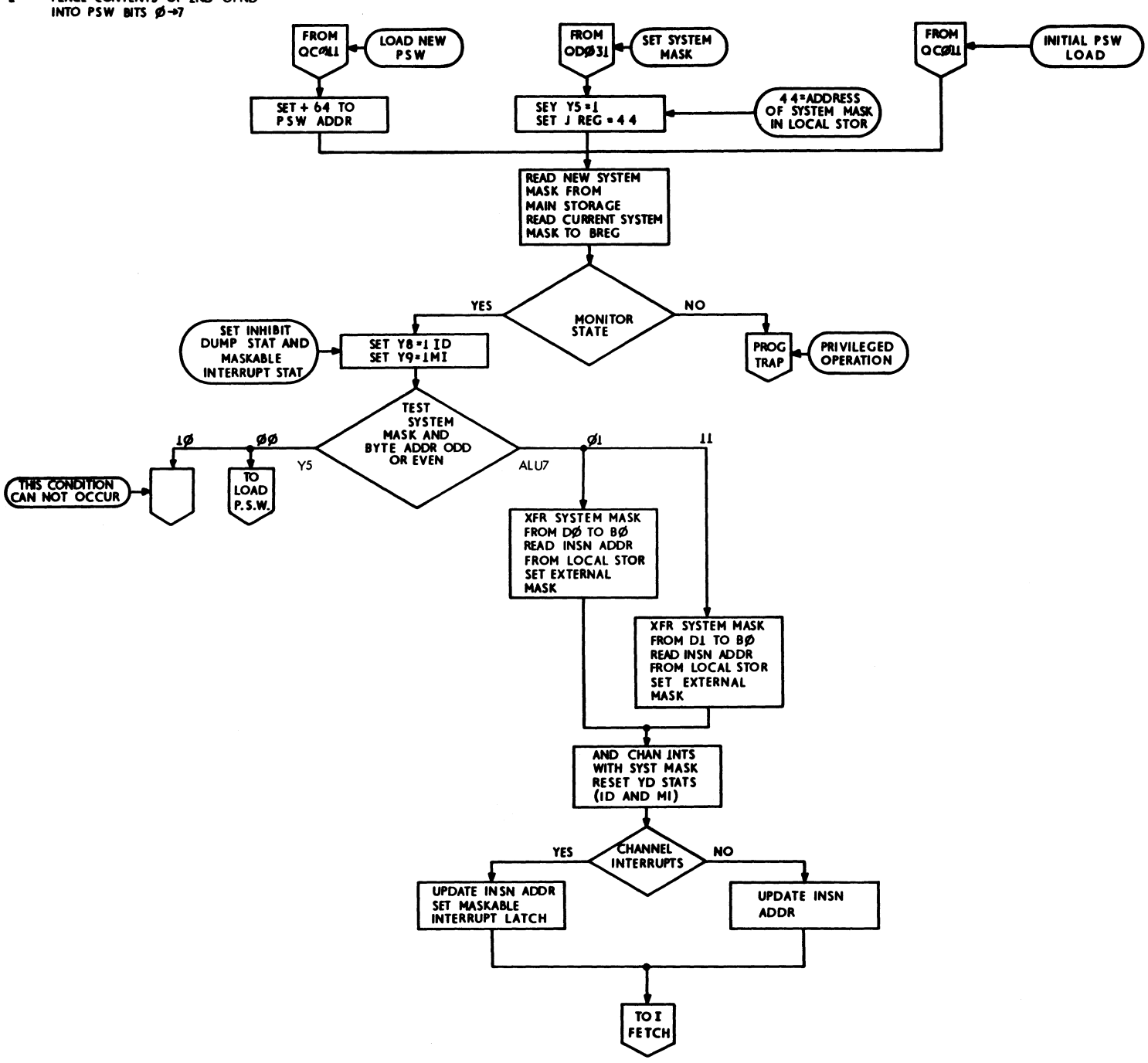
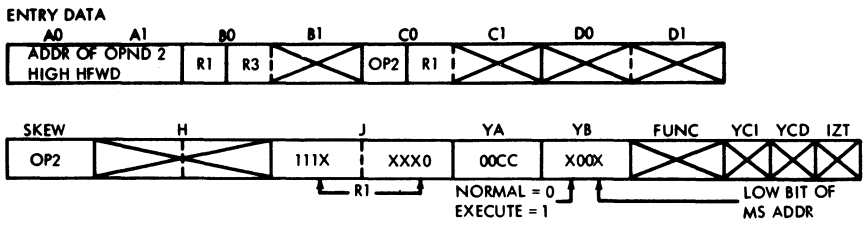
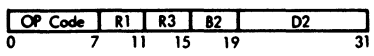


FIGURE 633. SET SYSTEM MASK (QC021)

ENTRY DATA



RS



LOAD MULTIPLE LM DMC SECTION 12C, ROUTINE 07, STOP ON MS 63C0

	ROS	A REG	B REG	C REG	D REG
①	351	6080	0F00	8000	1111
②	174	60BE	0FF0	0000	6000
③	1F5	60BE	0FF0	0000	6000

OBJECTIVES

FOR LOAD MULTIPLE THE REGISTERS STARTING WITH THE ONE SPECIFIED BY R1 AND ENDING WITH R3 ARE LOADED WITH THE 2ND OPND. STORE MULTIPLE THE CONTENTS REGISTERS R1 THRU R3 ARE LOADED INTO MAIN STORE STARTING AT THE ADDRESS SPECIFIED BY OPND 2

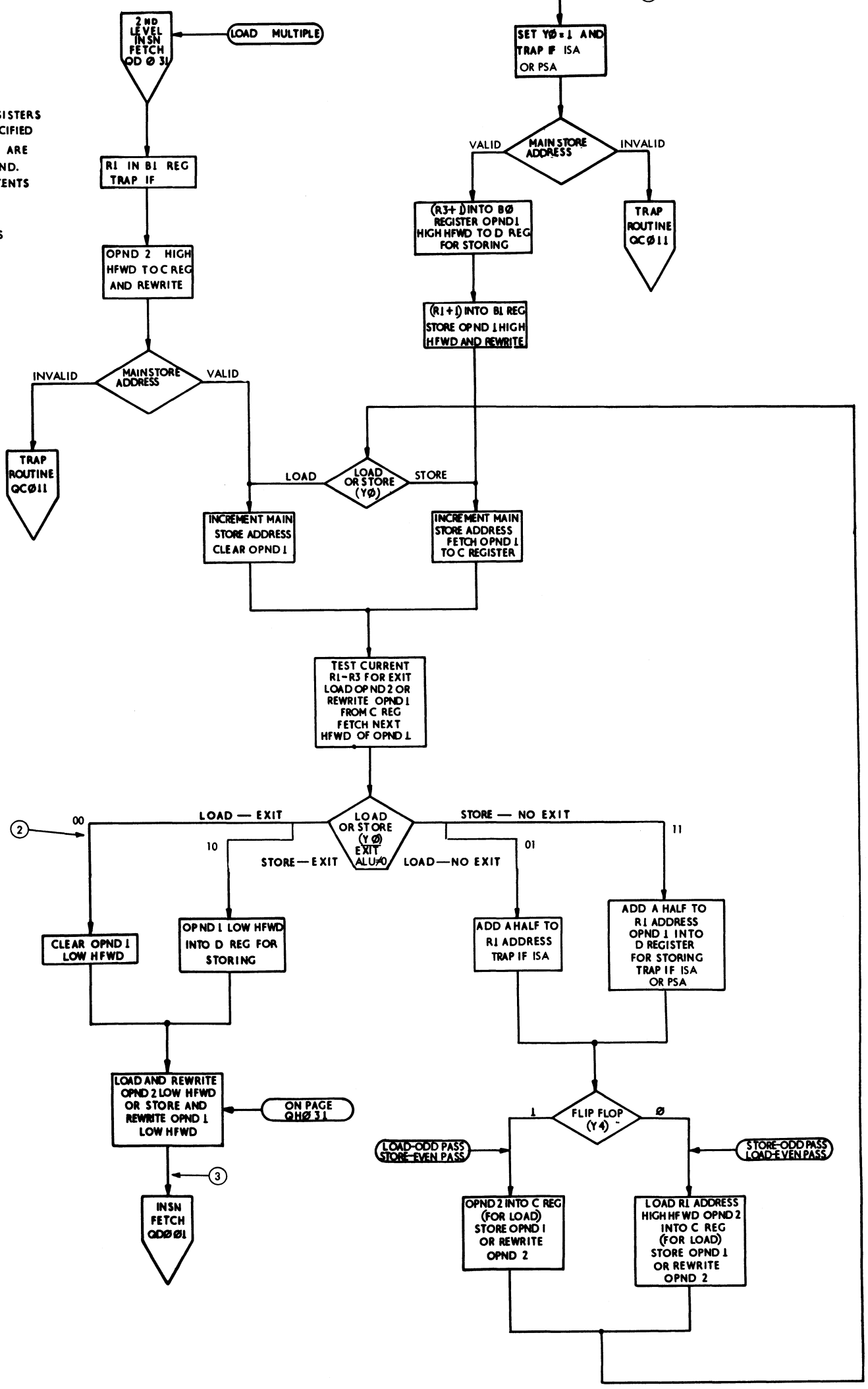
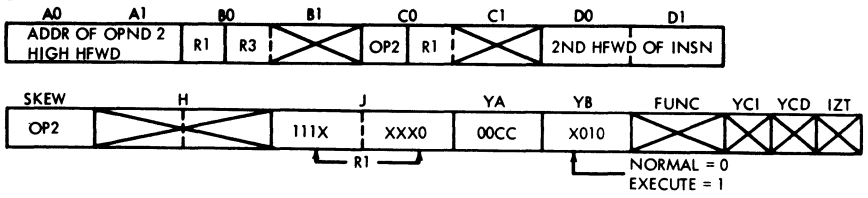
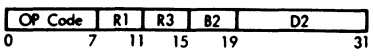


FIGURE 634.RS LOAD AND STORE MULTIPLE (QH041)

ENTRY DATA



RS



NO EXAMPLE SHOWN.

OBJECTIVES

TO PERFORM THE RS LOGICAL SHIFT OPERATIONS
(FOR DESCRIPTION OF OBJECTIVES OF EACH INSTRUCTION
SEE THE PRINCIPLES OF OPERATIONS MANUAL)

LEGEND		
SYMBOL	MEANING	ACHIEVED IN μ PROG BY
1L	SHIFT LEFT 1	LEFT SHIFT FUNCTION
4L	SHIFT LEFT 4	PASS Q FUNCTION WITH SKEW
1L.4L	SHIFT LEFT 5	LEFT SHIFT FUNCTION WITH SKEW
....8L	SHIFT LEFT +8	ADD A BYTE SHIFT TO LEFT
16L	SHIFT LEFT 16	HALFWORD SHIFT TO LEFT
1R	SHIFT RIGHT 1	RIGHT SHIFT FUNCTION
....8R	SHIFT RIGHT +8	ADD A BYTE SHIFT TO RIGHT
....16R	SHIFT RIGHT +16	ADD A HALFWORD SHIFT TO RIGHT
EXAMPLE		
(4L.16R)+(1R)	SHIFT RIGHT 13	PASS Q FUNCTION WITH SKEW TOGETHER WITH HALFWORD SHIFT TO RIGHT (LOOP 2), ALL FOLLOWED BY RIGHT SHIFT FUNCTION (LOOP 6)

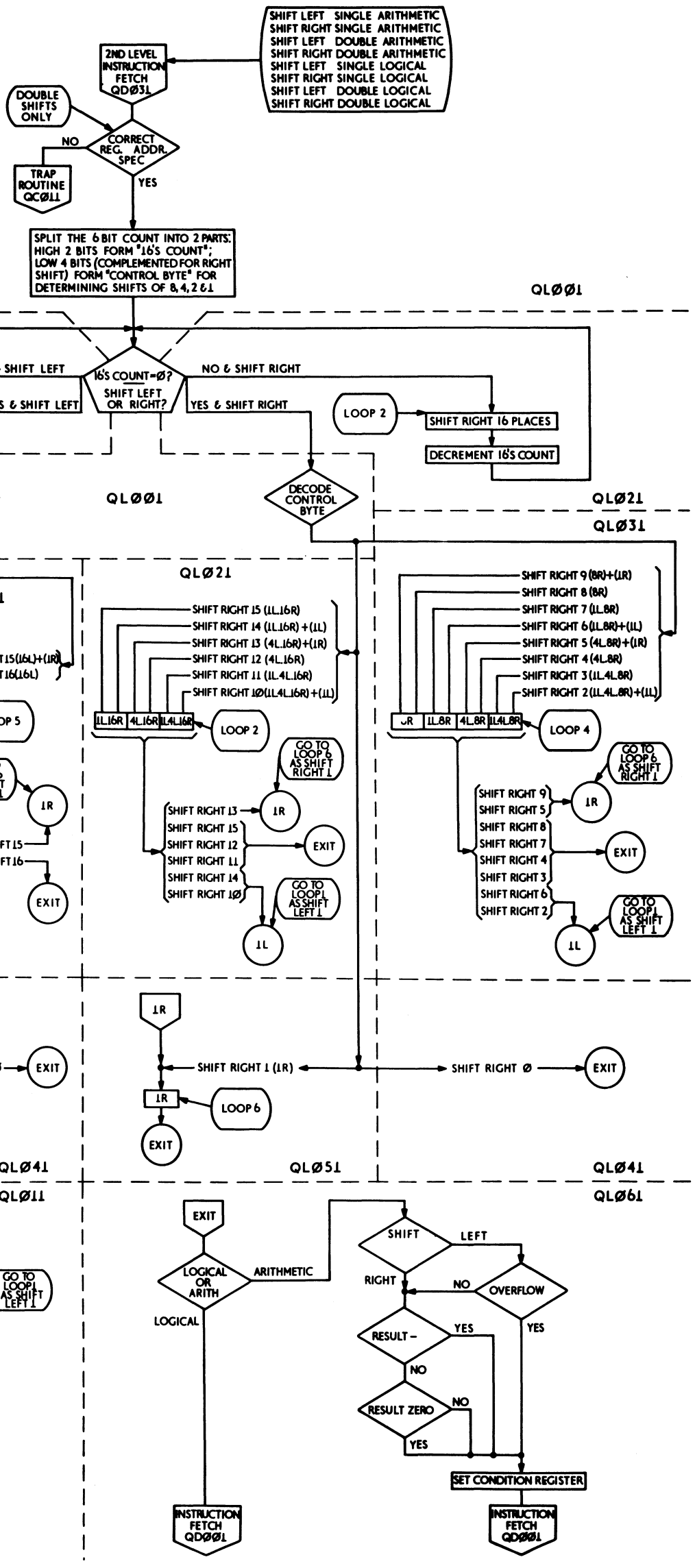
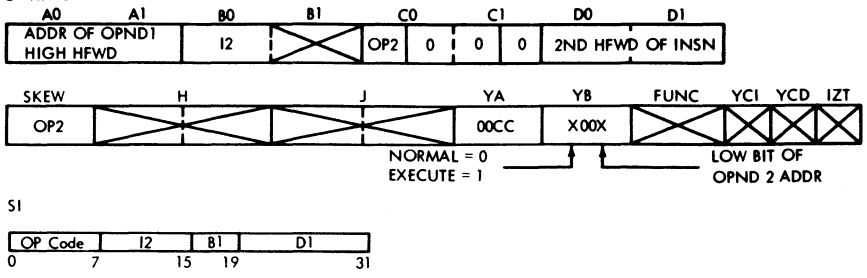


FIGURE 635. SHIFTS

	ROS	A REG	B REG	C REG	D REG
①	349	032C	0000	4003	0000
②	368	032C	00FF	4003	0000
③	368	01C0	DFFF	4D01	4400

ENTRY DATA



SI

OBJECTIVES

TO AND, OR, XOR OR MOVE THE IMMEDIATE DATA WITH THE BYTE SPECIFIED BY B1, D1.
THE CONDITION CODE IS SET ACCORDING TO RESULT FOR THE AND, OR, XOR OPS. IT IS UNCHANGED FOR THE MOVE OP

TABLE 1

STAT SETTING					
OP	MOVE	AND	OR	XOR	
INDIRECT FUNCTION	PASS P	AND	OR	XOR	
B1 REG AT POINT A	ØØCRXØ1Ø	11111111	ØØØØØØØØ	ØØØØØØØØ	
YCD AT POINT A	Ø	1	Ø	Ø	
Y STATS AT POINT B	ØØCRXØ1Ø	ØØØØØØØØ	ØØØØØØØØ	ØØØØØØØØ	Ø RESULT CR = ØØ
Y STATS AT POINT C	ØØCRXØ1Ø	ØØØ1ØØØØ	ØØØ1ØØØØ	ØØØ1ØØØØ	Ø̄ RESULT CR = Ø1

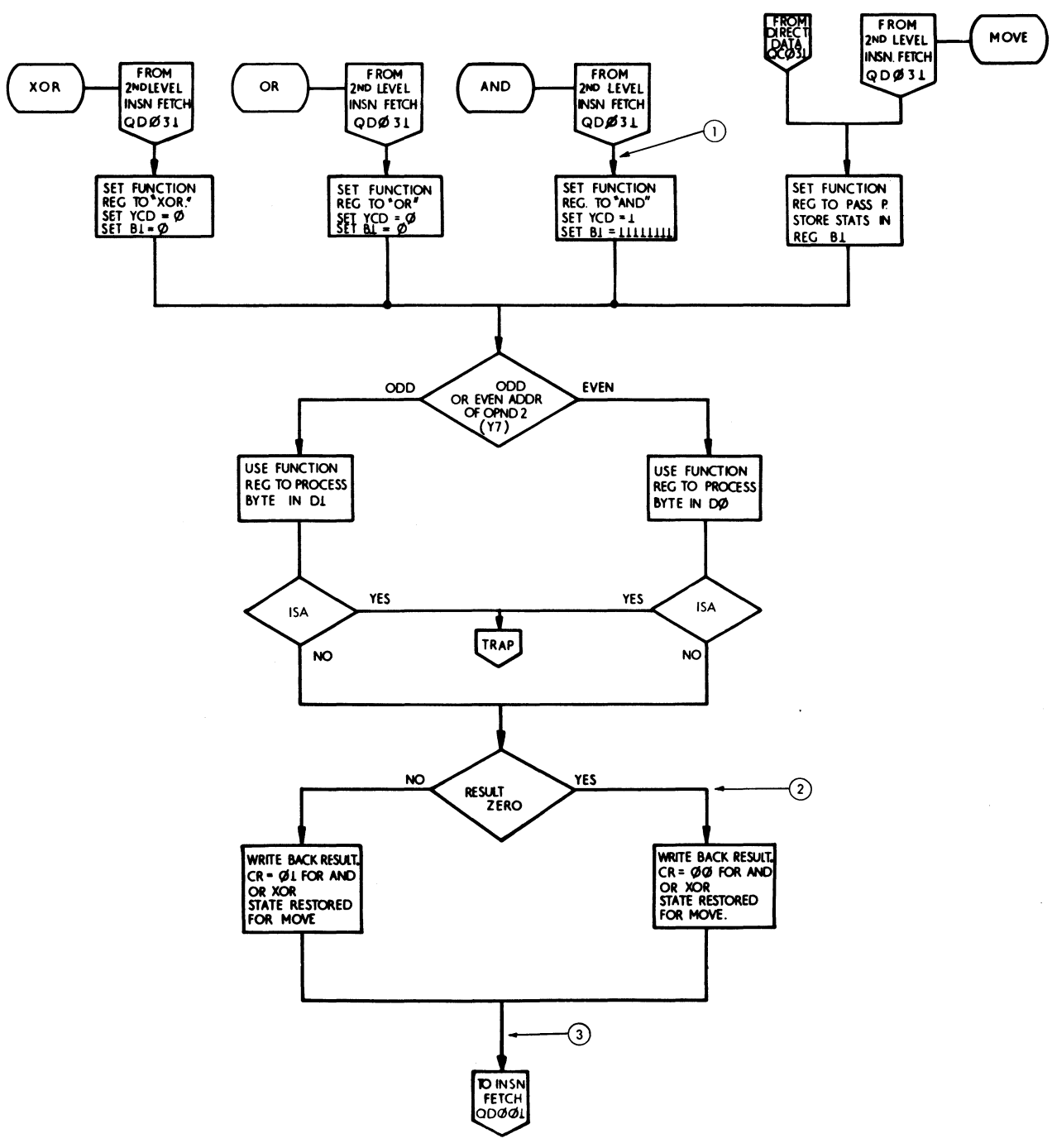
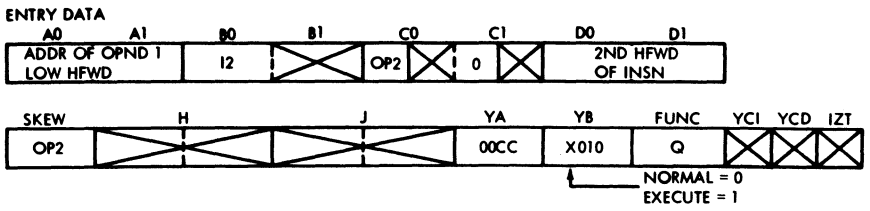
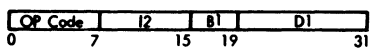


FIGURE 636 SI OPERATIONS, AND, OR, EXOR, MOVE (QP001)

ENTRY DATA



SI



NO EXAMPLE SHOWN.

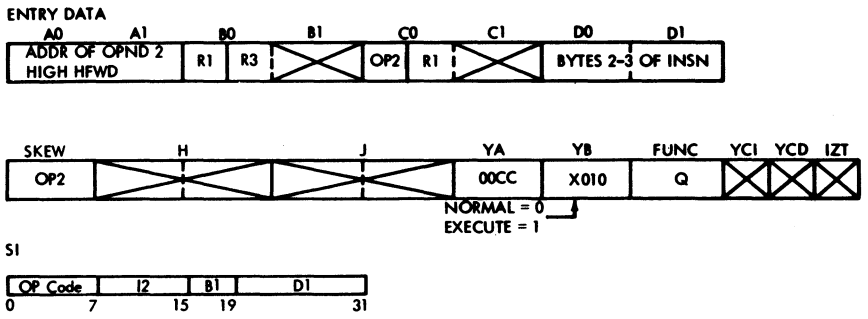
OBJECTIVES

WRITE THE BYTE DESIGNATED BY THE OPERAND ADDRESS IS MADE AVAILABLE AS A SET OF DIRECT-OUT STATIC SIGNALS. THE I2 FIELD IS MADE AVAILABLE SIMULTANEOUSLY AS A SET OF TIMING SIGNALS.

READ A DIRECT-IN DATA BYTE IS ACCEPTED FROM AN EXTERNAL DEVICE AND PLACED IN THE LOCATION DESIGNATED BY THE OPERAND ADDRESS.

FIGURE 637. READ DIRECT AND WRITE DIRECT (QC031)

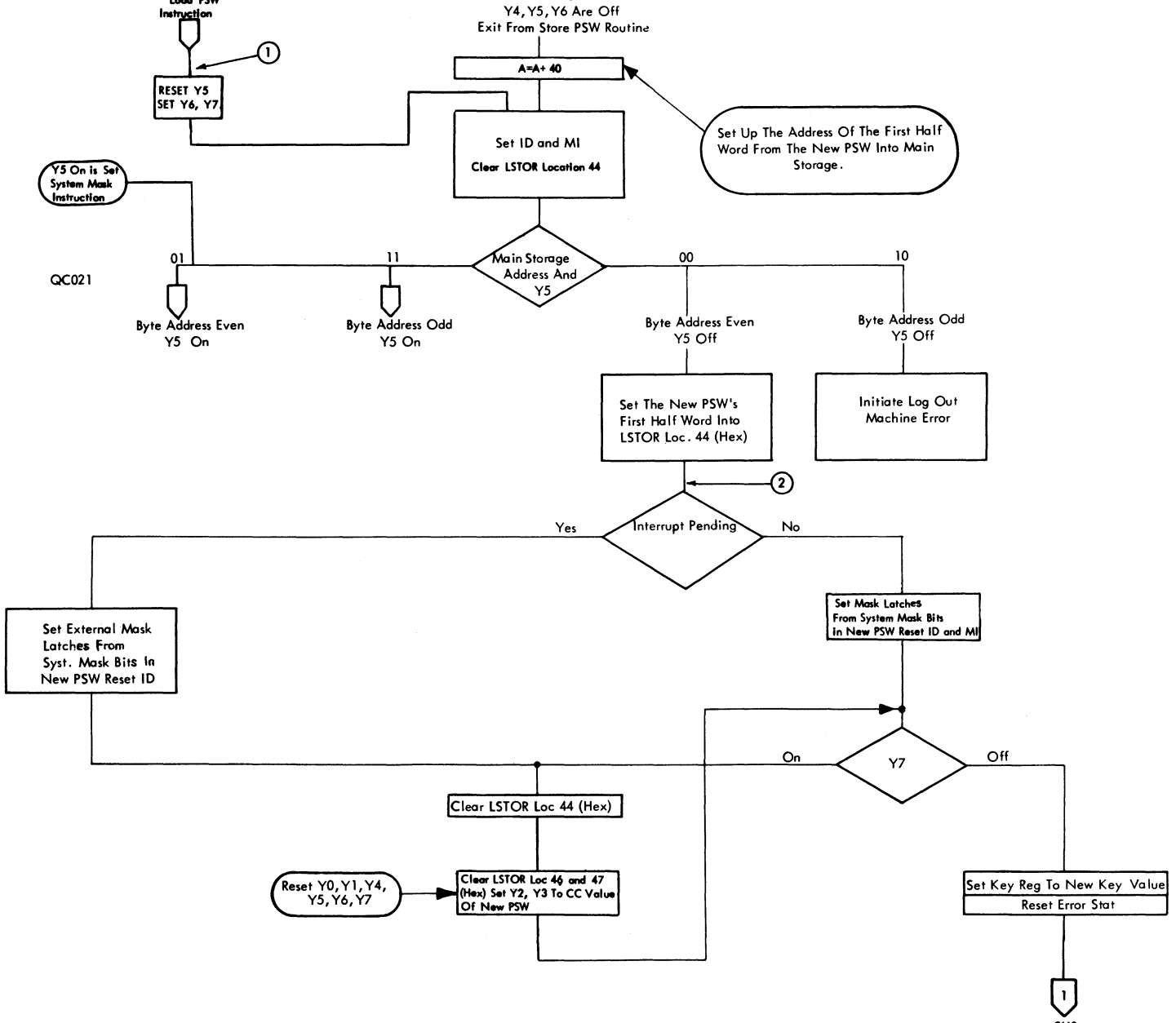
ENTRY DATA



SI

LOAD PSW LPSW DMC SECTION 124, STOP AT MS 641E

	ROS	A REG	B REG	C REG	D REG
①	344	6050	0000	2000	F050
②	4CC	65E0	0004	0000	49D8
③	106	0834	0004	0000	0834
④	12C	0CAC	3000	3004	0CAC



SH2

FIGURE 638. LOAD PSW (SHEET 1 OF 2)

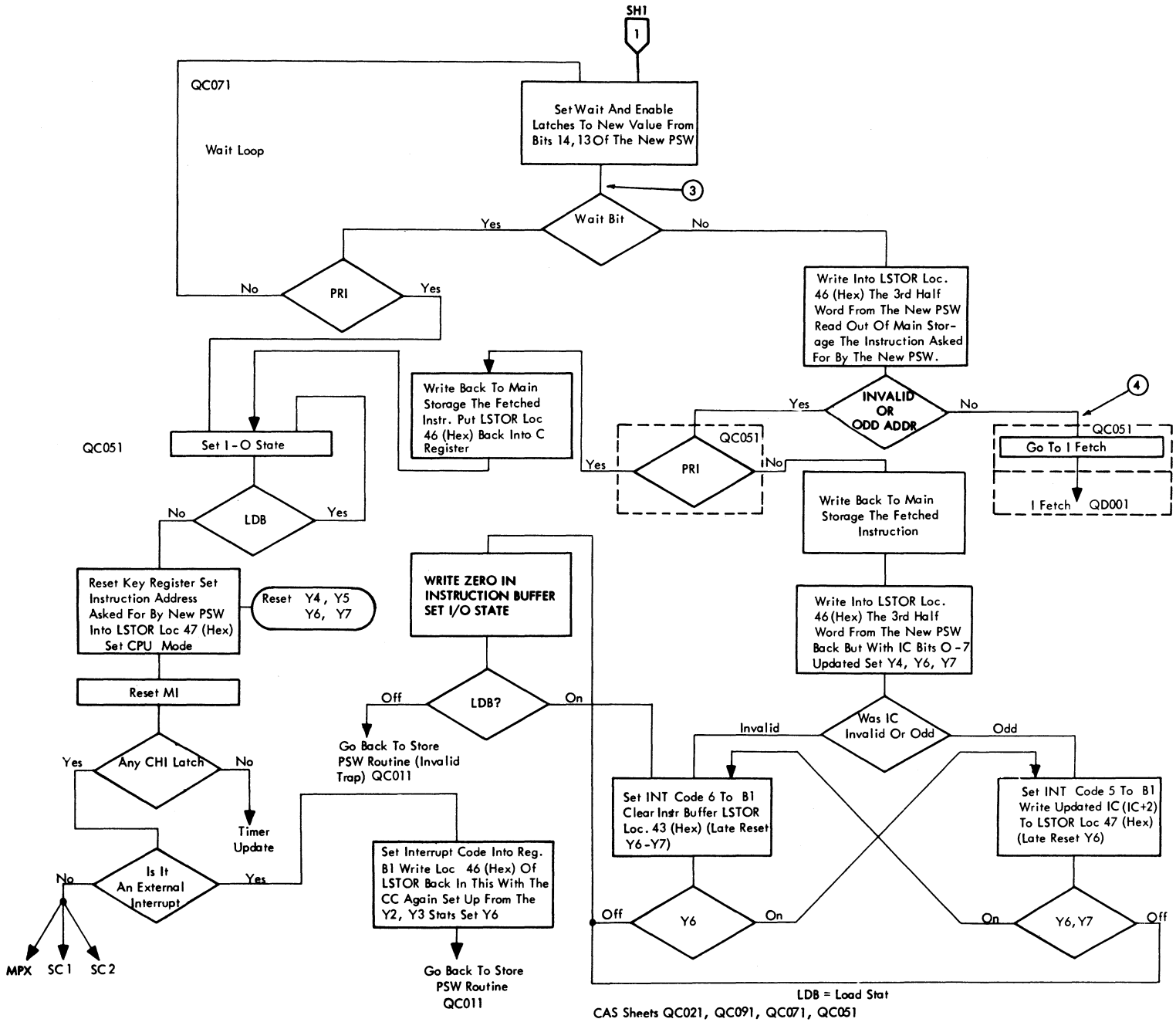


FIGURE 638 LOAD PSW (SHEET 2 OF 2)

NO EXAMPLE SHOWN.

ENTRY DATA

A0	A1	B0	B1	C0	C1	D0	D1
ADDR OF OPND 2 HIGH HFWD	R1	R3		OP2	R1	0	2ND HFWD OF INSN

SKEW	H	J	YA	YB	FUNC	YCI	YCD	IZT
		111X (R1) XXX0	00CC	X010 (NORMAL = 0, EXECUTE = 1)				

RS

OP Code	R1	R3	B2	D2
0 7	11	15	19	31

EFFECTIVE MICRO PROGRAM ADDRESS
CONTENTS OF B2 + D2
R1 R3 = STATS SETTINGS

OBJECTIVE

TO PERFORM THE DIAGNOSE INSTRUCTION
DETERMINED BY THE SETTING OF THE YA AND YB
STATS
(R1) (R3)

YA	YB	FUNCTION
ØØØØ	ØØØ1	XFER LOCAL STORE TO LOCAL STORE TO ONE EXTENDED WORD
ØØØØ	ØØ11	XFER LOCAL STORE TO LOCAL STORE FROM ONE EXTENDED WORD
ØØØØ	1ØØ1	CLEAR A LOCAL STORAGE WORD
Ø1ØØ	1ØØØ	XFER A LOCAL STORAGE WORD TO BUMP
Ø1ØØ	ØØØØ	XFER BUMP TO LOCAL STORE
Ø1ØØ	ØØ1Ø	CYCLE A BUMP WORD
ØØØØ	Ø11Ø	CYCLE SPLS WORD CPU
Ø1ØØ	Ø11Ø	CYCLE SPLS WORD MPX

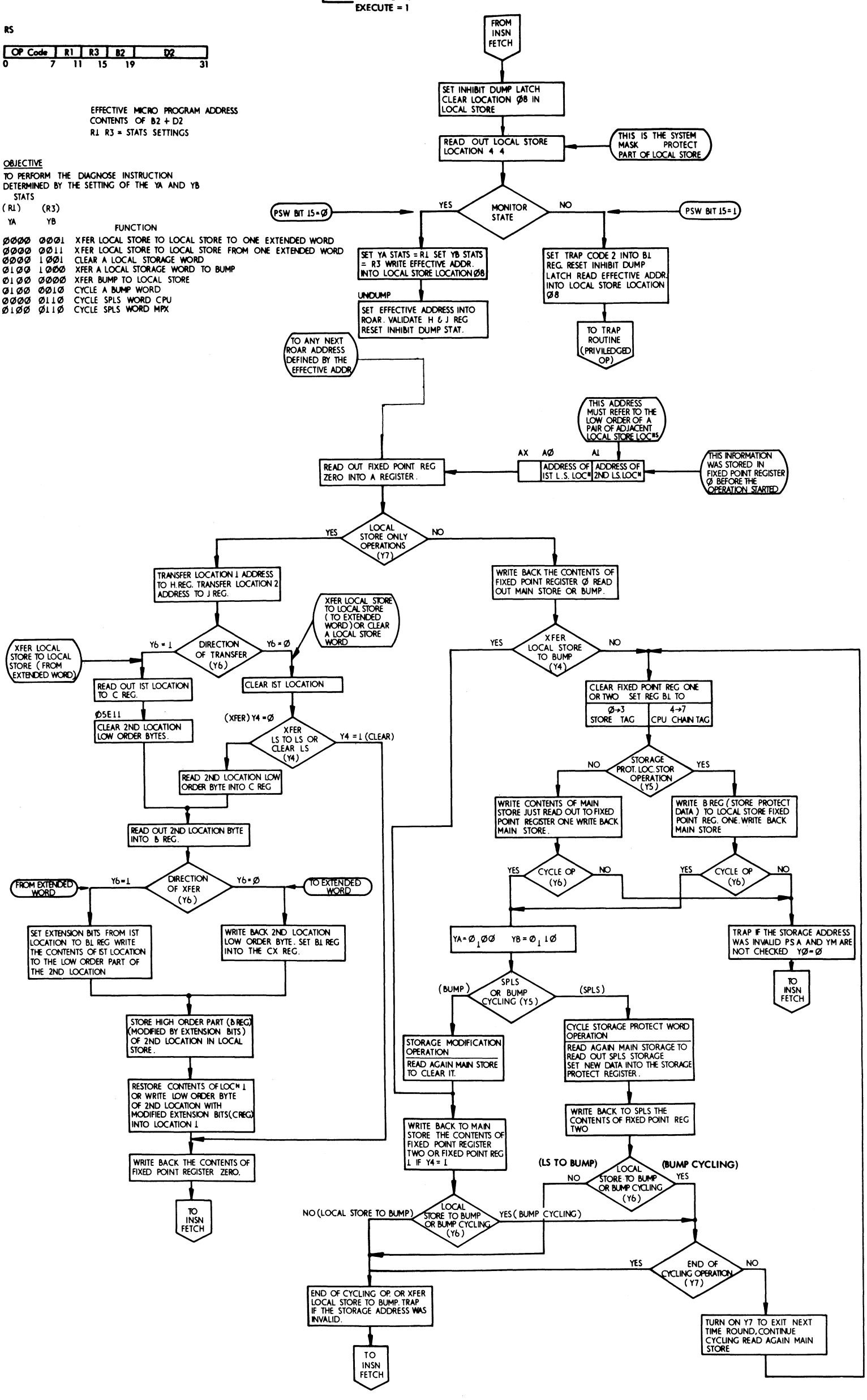
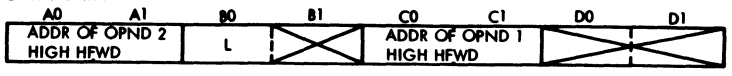
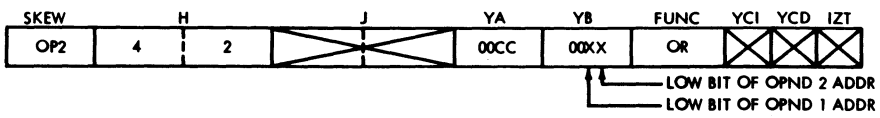


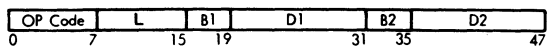
FIGURE 639, DIAGNOSE INSTRUCTION (QS 101)

ENTRY DATA





SS



TRANSLATE TR DMC SECTION 183, ROUTINE 02 STOP ON MS 617A

THEN STOP ON MS 617A

	ROS	A REG	B REG	C REG	D REG
①	C18	6A68	0000	6868	FFFF
②	C70	6868	0068	6768	33FE
③	CD4	679B	009B	3768	33FE
④	C7C	6868	FF68	67CC	CCFE

OBJECTIVE

BYTES OF THE FIRST OPERAND, ARE USED AS ARGUMENTS TO REFRENCE THE LIST DESIGNATED BY THE SECOND OPERAND ADDRESS. EACH RESULTING FUNCTION BYTE REPLACES THE CORRESPONDING ARGUMENT BYTE IN THE FIRST OPERAND.

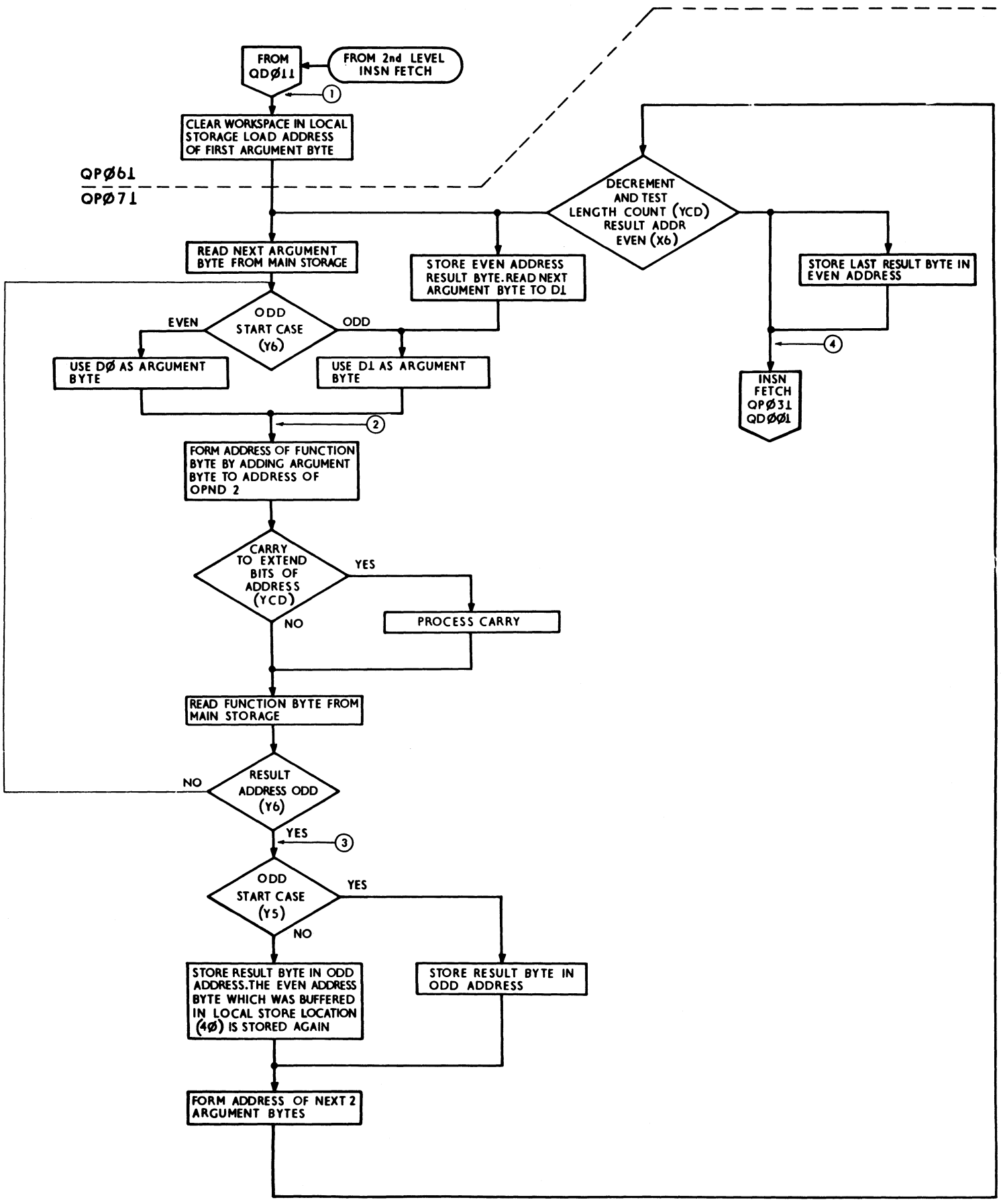
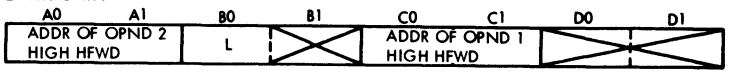
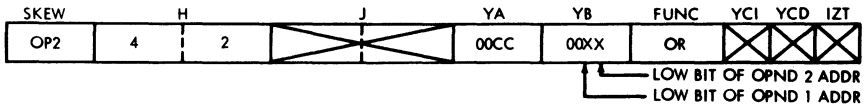


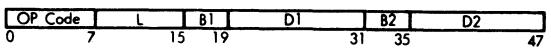
FIGURE 641.SS TRANSLATE

ENTRY DATA





SS



TRANSLATE AND TEST TRT DMC SECTION 185,

ROUTINE 02. STOP ON MS 6182

	ROS	A REG	B REG	C REG	D REG
①	C1A	6750	0000	6850	FFFE
②	CD4	6794	0094	6750	4409
③	CE6	6794	10BB	6850	0000

OBJECTIVE

BYTES OF THE FIRST OPERAND ARE USED AS ARGUMENTS TO OBTAIN FUNCTION BYTES FROM THE LIST DESIGNATED BY THE SECOND OPERAND ADDRESS. THE OPERATION PROCEEDS UNTIL THE FIRST OPERAND FIELD IS EXHAUSTED OR UNTIL A NON-ZERO FUNCTION BYTE IS REACHED. IN THE LATTER CASE THE FUNCTION BYTE AND CORRESPONDING ARGUMENT ADDRESS ARE INSERTED IN GENERAL REGISTERS. THE CONDITION CODE IS SET BY THE OPERATION. BOTH OPERANDS REMAIN UNCHANGED.

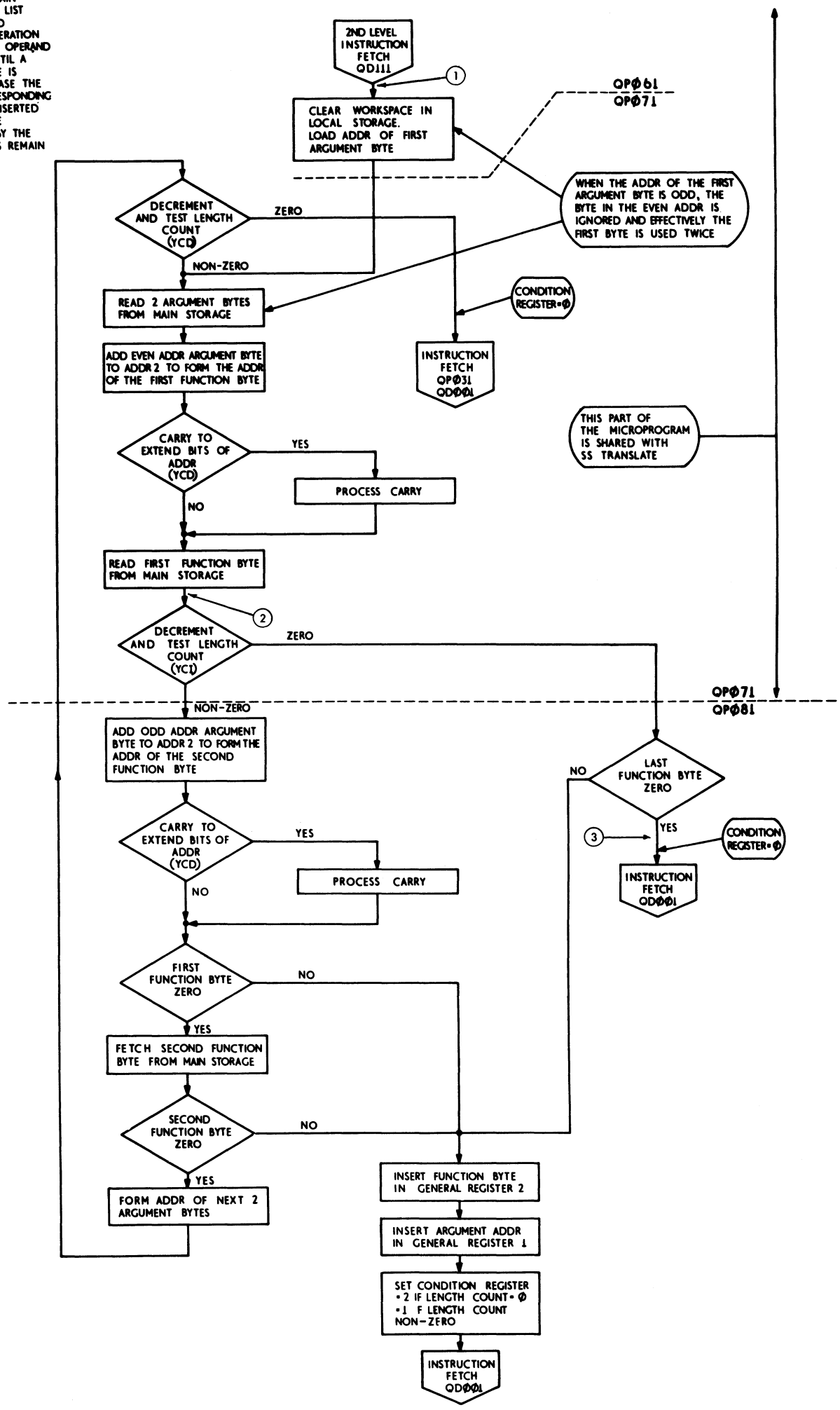
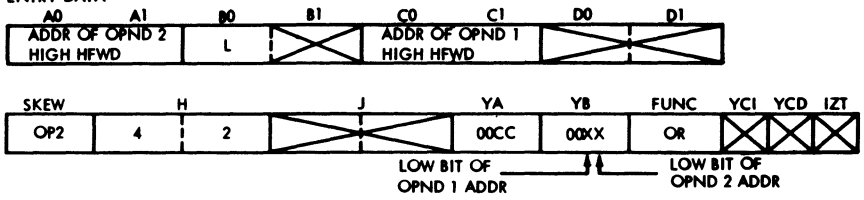
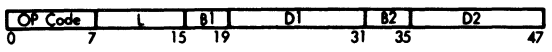


FIGURE 642 SS TRANSLATE AND TEST

ENTRY DATA



SS



OBJECTIVES

THE FORMAT OF THE SECOND OPND (SOURCE) IS TO BE CHANGED FROM PACKED TO ZONED AND EDITED UNDER CONTROL OF THE FIRST OPND (PATTERN)
FOR FULLER DESCRIPTION OF OBJECTIVES REFER TO PRINCIPLES OF OPERATION MANUAL

EDIT DMC SECTION 1EE, ROUTINE 01, STOP ON MS 60F8

	ROS	A REG	B REG	C REG	D REG
①	C1C	66E0	0300	66C0	666C
②	DAD	66C0	03BF	F040	40BF
③	DC4	66C0	40BF	F040	BFBF
④	DB4	66C0	BFBF	4040	40BF
⑤	D94	66C4	FE00	4040	4040

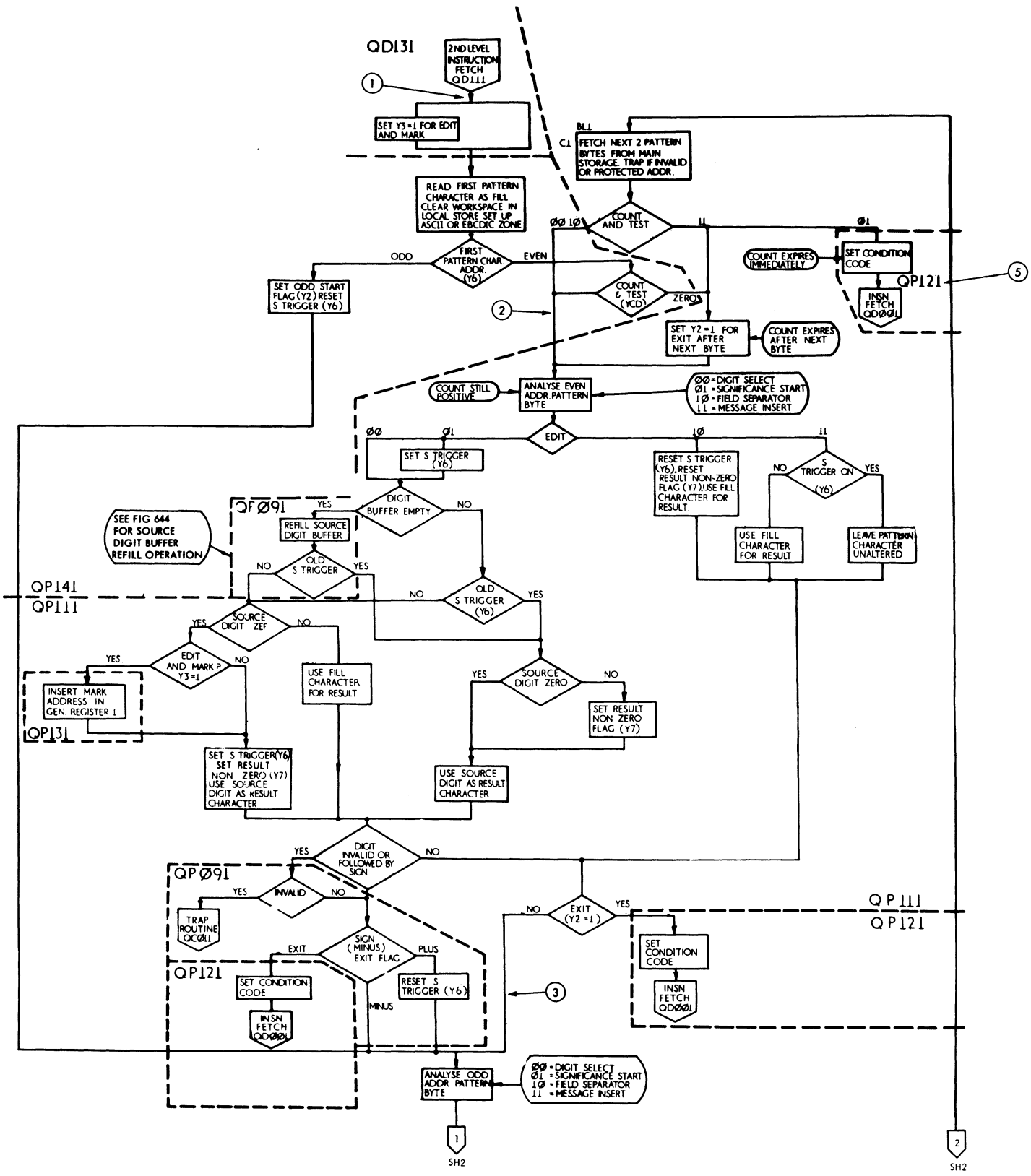


FIGURE 643 EDIT, EDIT AND MARK (QP091, QP101, QP121, QP131, QP141) (SHEET 1 OF 2)

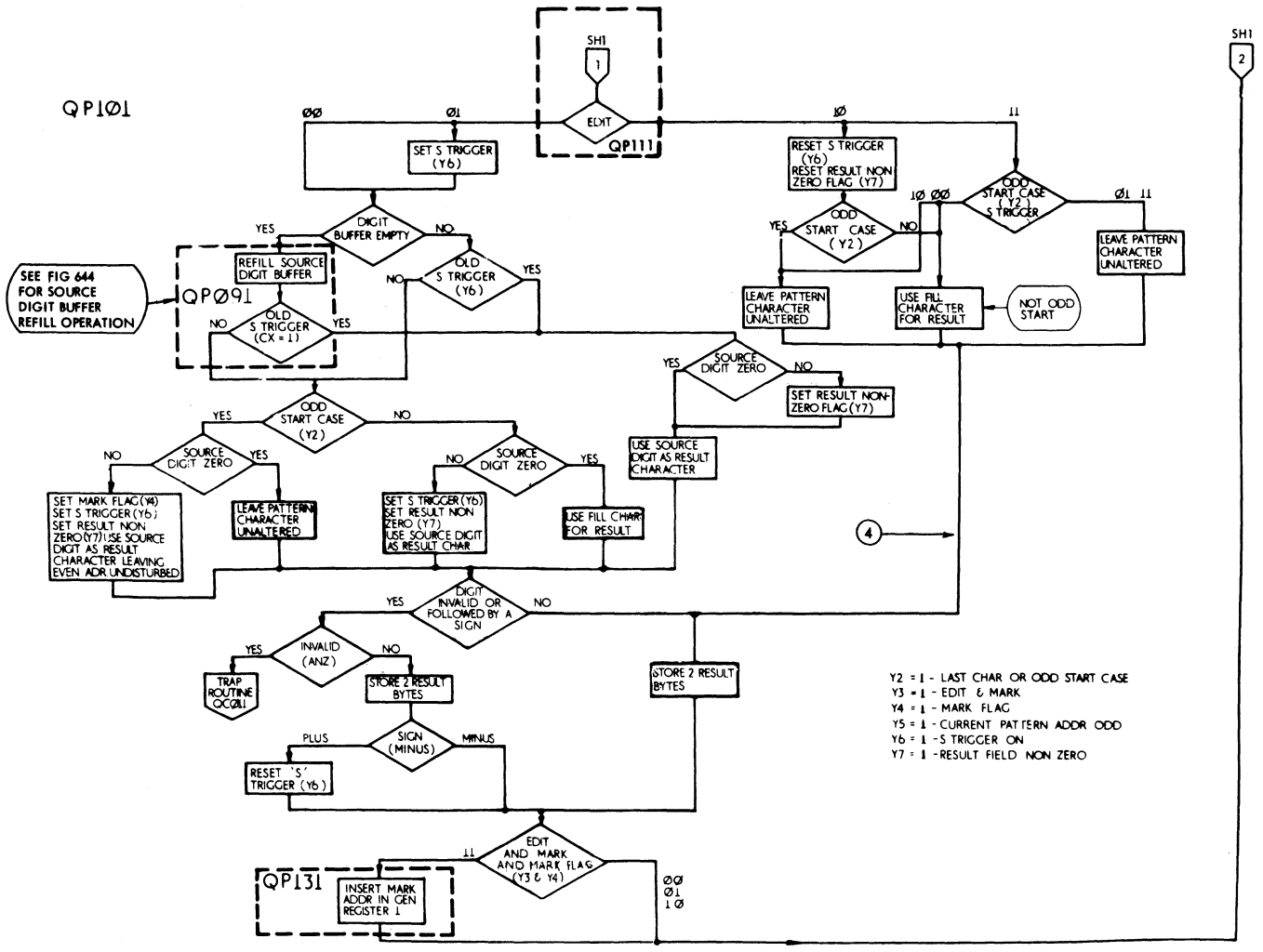


FIGURE 643 EDIT, EDIT AND MARK (QP091, QP101, QP121, QP131, QP141) (SHEET 2 OF 2)

OBJECTIVE
THIS SUBROUTINE IS USED BY THE EDIT MICROPROGRAM TO FETCH UP TO 4 DIGITS FROM THE SOURCE FIELD. THE FORMAT OF THE DIGITS IS CHANGED FROM PACKED TO ZONED. THE DIGITS ARE PUT IN A LOCAL STORAGE BUFFER FOR USE WHEN REQUIRED IN THE MAIN EDIT LOOP. SPECIAL CODES ARE SET IN THE BUFFER FOR SIGNS OR INVALID DIGITS.

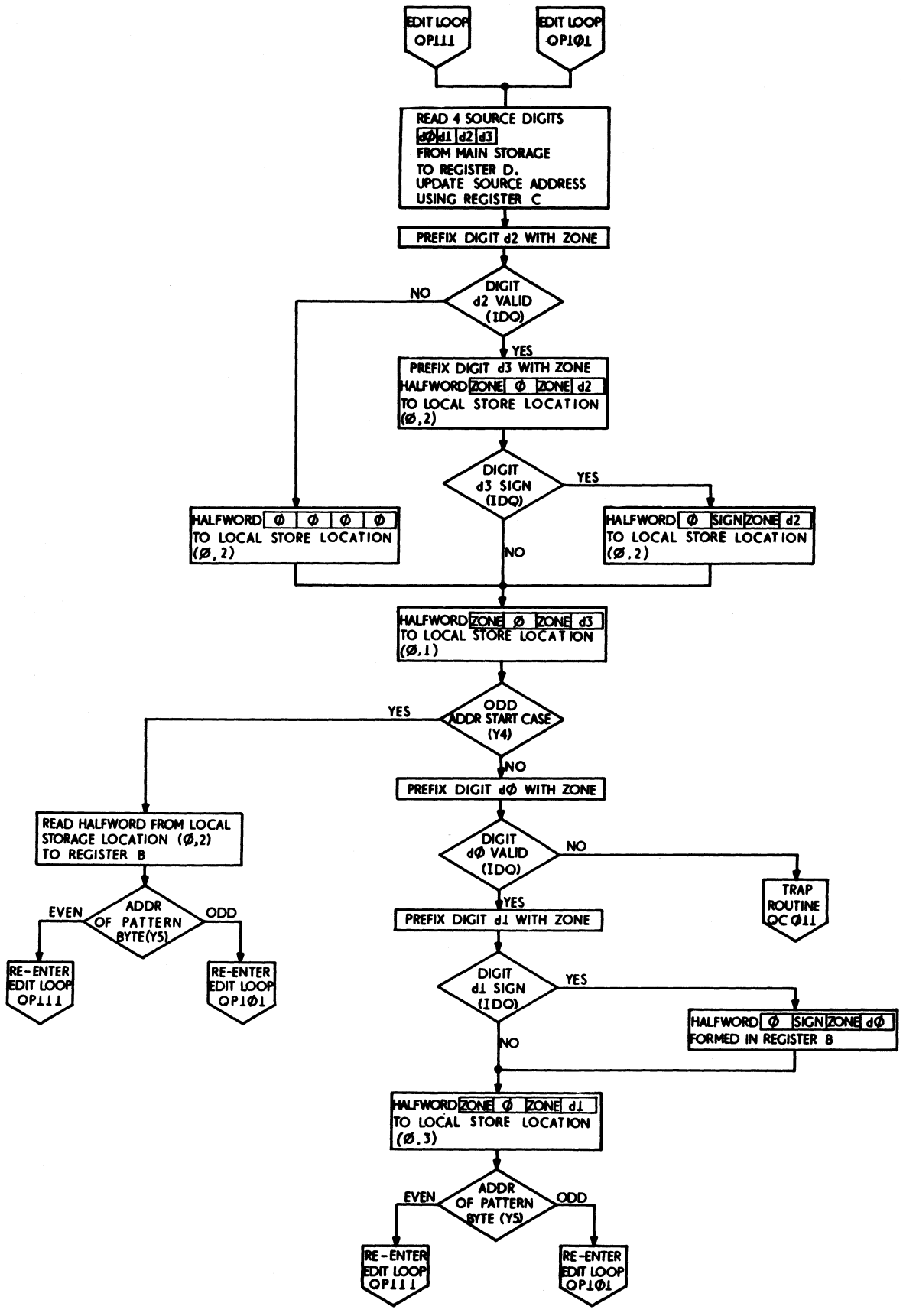
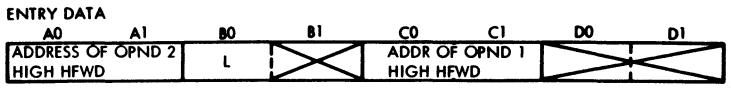
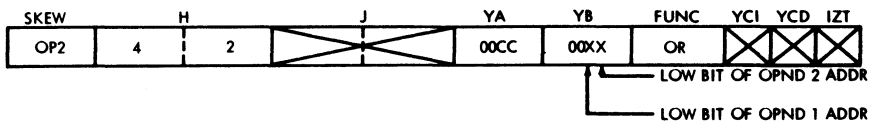


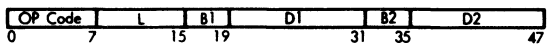
FIGURE 644 SS EDIT, REFILL (QP 091)

ENTRY DATA





SS



OPND 1 = DEST
OPND 2 = SOURCE

OBJECTIVES

FOUR BITS FROM EACH BYTE OF THE SECOND OPERAND FIELD. REPLACE THE CORRESPONDING 4 BITS IN THE FIRST OPERAND.
FOR MOVE NUMERIC THE LOW ORDER 4 BITS ARE USED, FOR MOVE ZONES THE 4 HIGH ORDER BITS. FOR OVERLAPPING FIELDS MOVEMENT IS LEFT TO RIGHT THROUGH EACH FIELD, ONE BYTE AT A TIME.

MOVE NUMERIC DMC SECTION 17D, ROUTINE 01, STOP ON MS 6108

	ROS	A REG	B REG	C REG	D REG
①	C02	6978	0000	6958	F1F1
②	CB2	6978	0000	697A	F1F1
③	C8E	6958	F100	69FF	0000
④	C7C	6958	0001	6901	0100

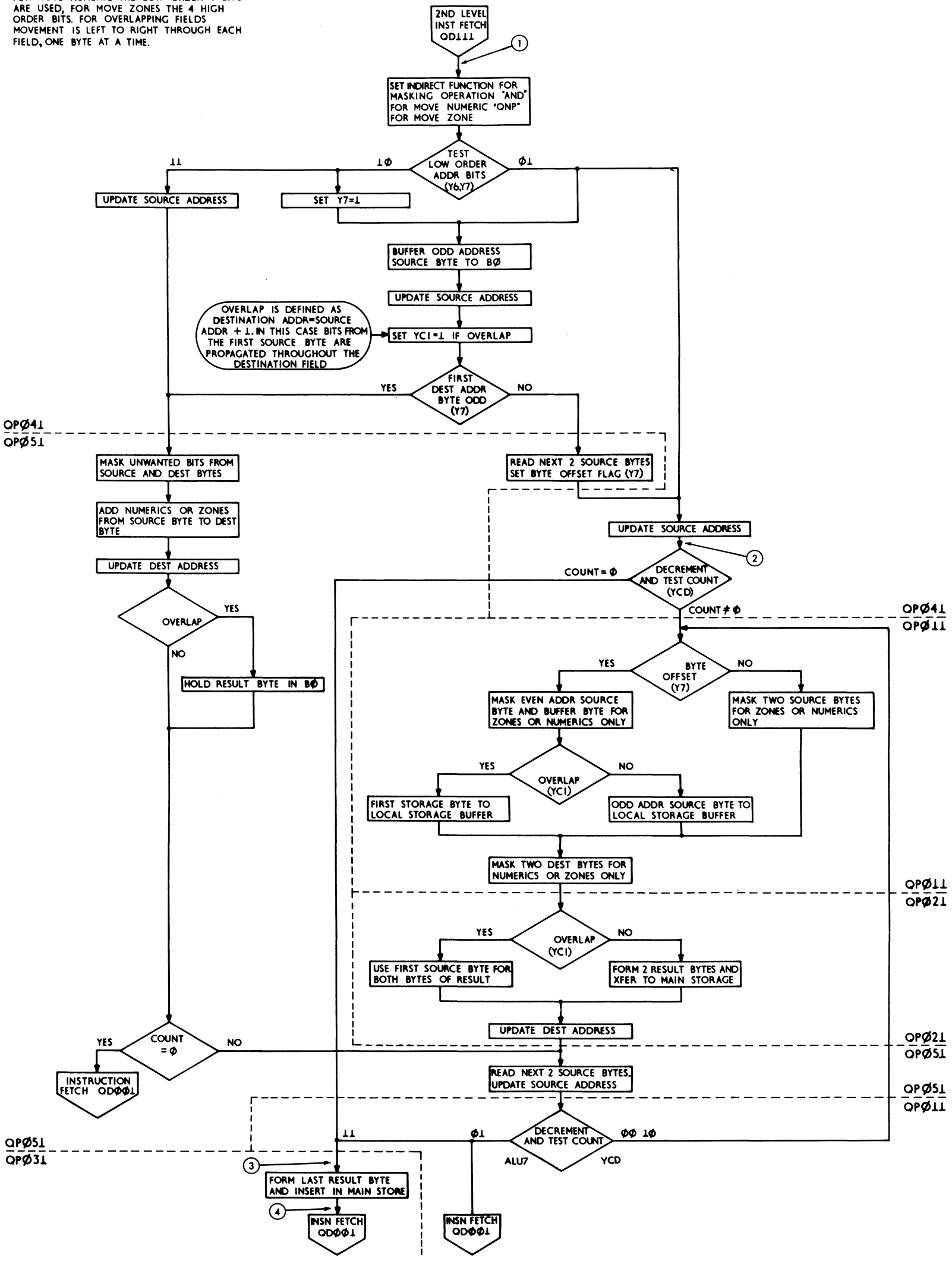
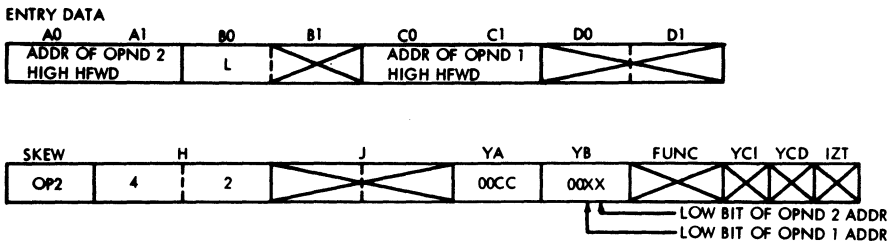
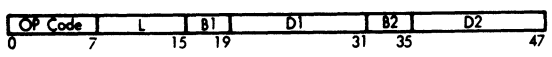


FIGURE 647 SS LOGICAL OPERATIONS - MOVE NUMERIC, MOVE ZONE

ENTRY DATA



SS



OPND 1 - DEST
OPND 2 - SOURCE

OBJECTIVE

BYTES FROM THE SECOND OPERAND FIELD ARE COPIED TO THE FIRST OPERAND LOCATION. MOVEMENT IS LEFT TO RIGHT THROUGH EACH FIELD A BYTE AT A TIME.

MOVE CHARACTER DMC SECTION 141, ROUTINE 18, STOP ON MS 676C

	ROS	A REG	B REG	C REG	D REG
①	C04	6CF8	0300	0008	F8F9
②	CB0	6CF8	0303	00FA	F8F9
③	C5A	0008	F8F9	6C02	F8F9
④	CB5	6CFC	6C00	6CFE	FCFD

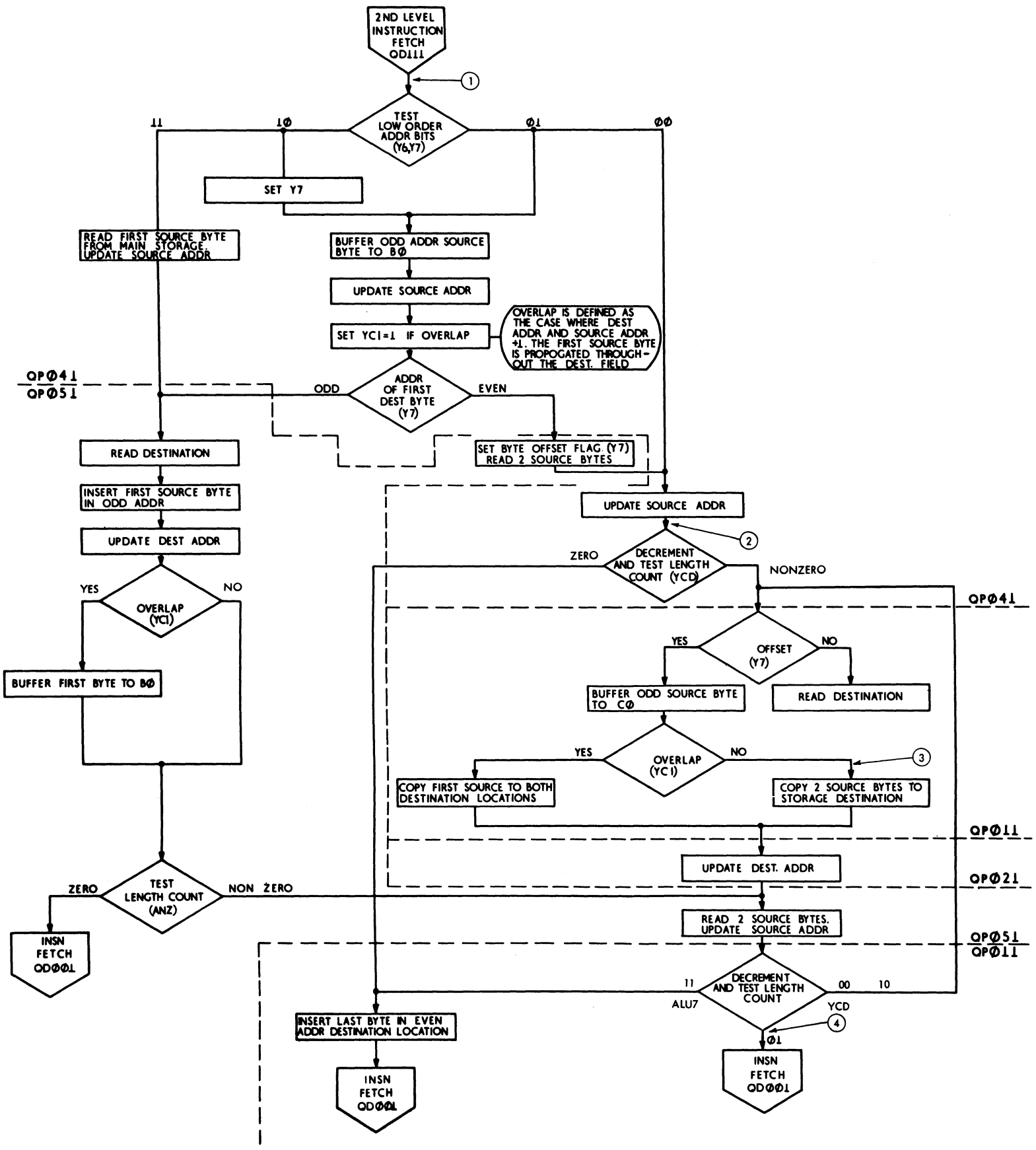
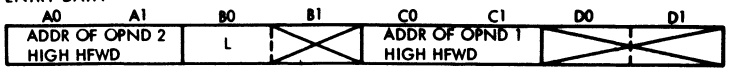
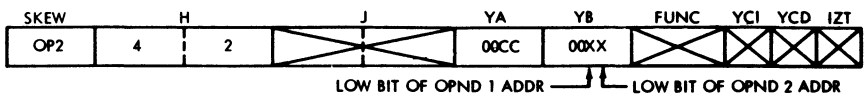


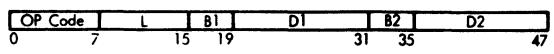
FIGURE 648.SS LOGICAL OPERATIONS, MOVE COMPLETE

ENTRY DATA





SS



DEST = OPERAND 1
SOURCE = OPERAND 2

COMPARE LOGICAL CLC DMC SECTION 141, ROUTINE 08, STOP ON MS 631C

	ROS	A REG	B REG	C REG	D REG
①	COA	6C00	FD00	6C00	0001
②	CB0	6C00	FDFD	6C02	0001
③	C59	6C00	0001	6CFC	0001
④	CB5	6C02	6CFC	6CFA	0203

OBJECTIVE

THE FIRST OPERAND IS COMPARED WITH THE SECOND OPERAND AND THE CONDITION CODE IS SET ACCORDINGLY. THE OPERATION PROCEEDS FROM LEFT TO RIGHT AND TERMINATES WHEN EQUALITY IS FOUND.

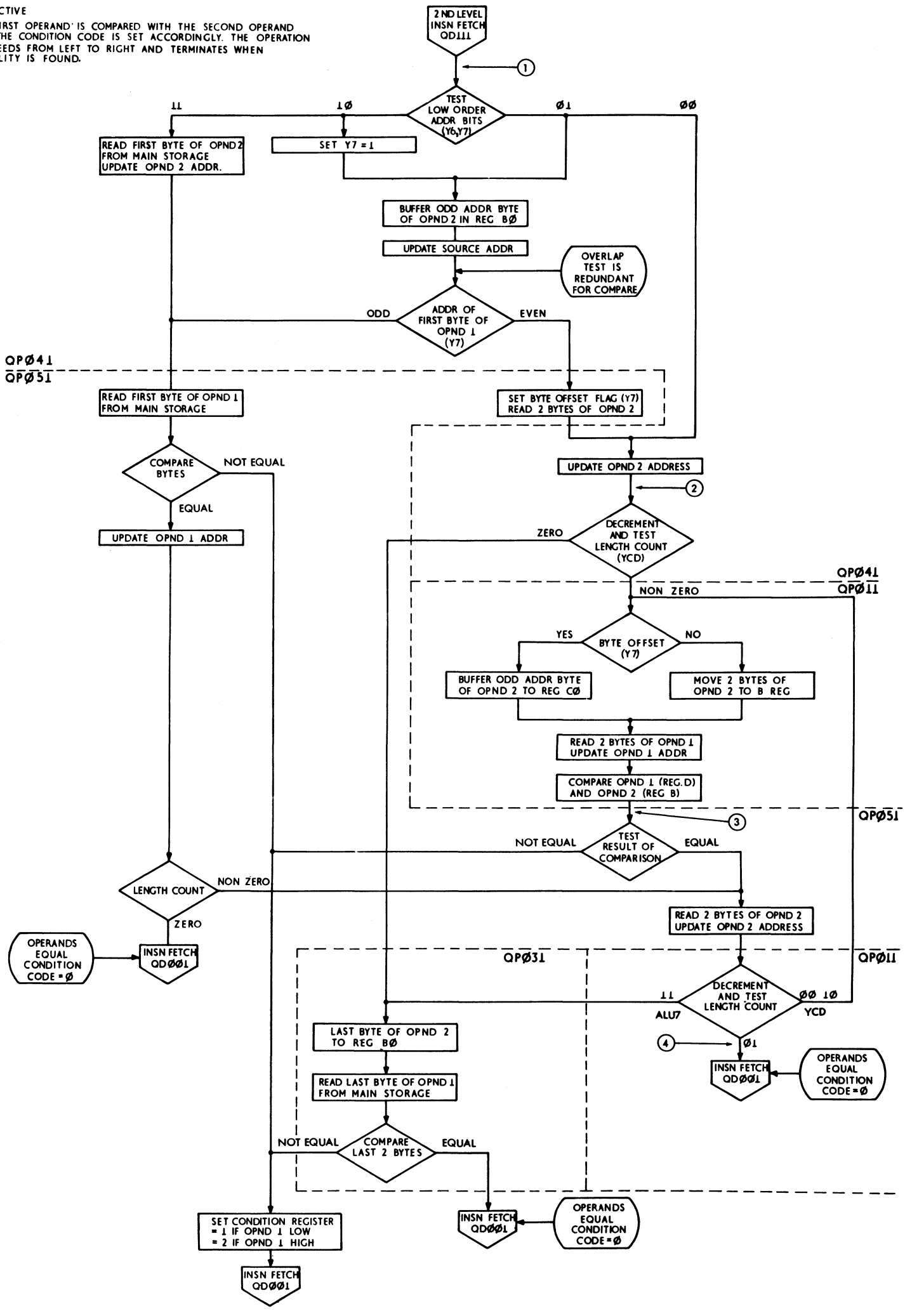
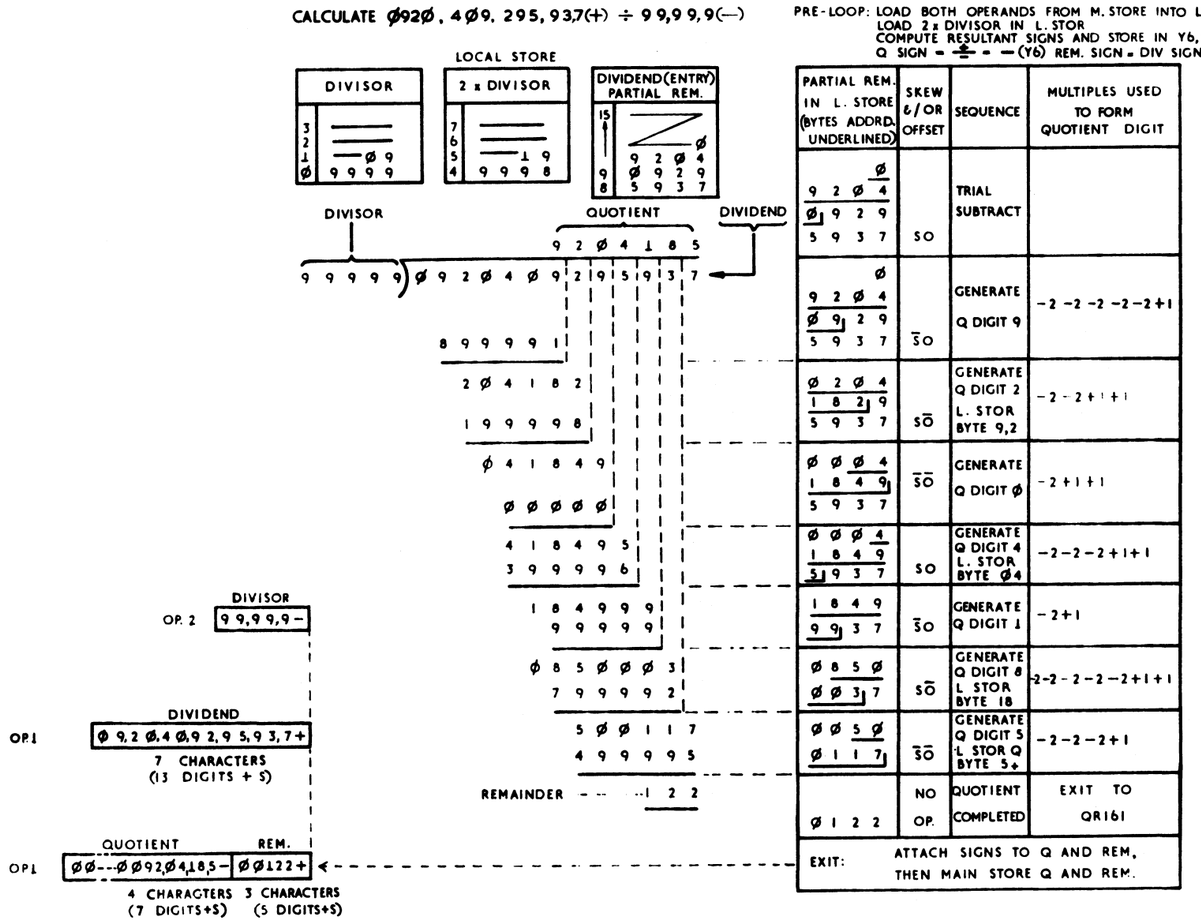


FIGURE 649.SS LOGICAL OPERATIONS, COMPARE

FIGURE 651. DECIMAL DIVIDE EXAMPLE



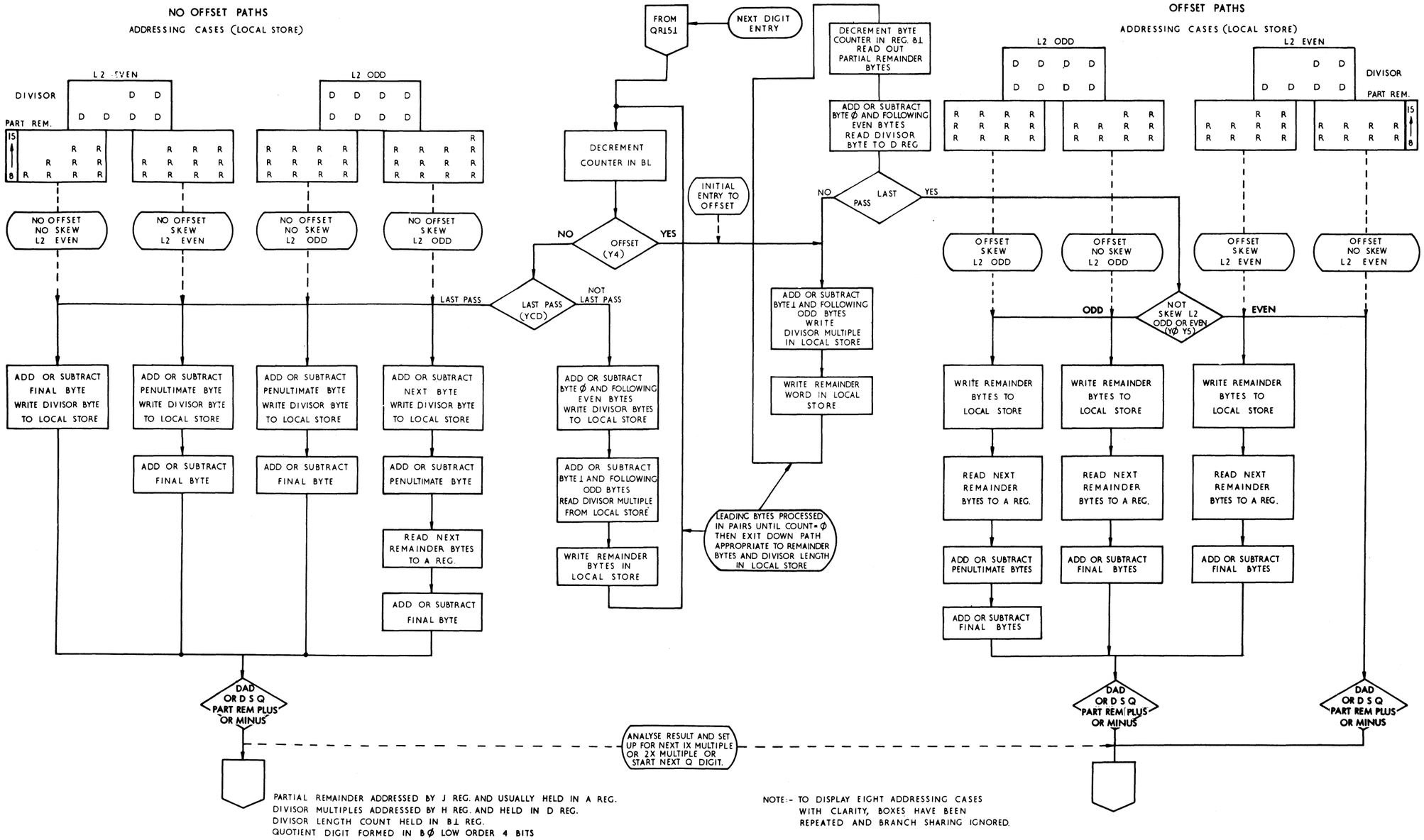


FIGURE 652 DECIMAL DIVIDE ADD/ SUBTRACT PATHS (QR141)

ENTRY DATA

A0	A1	B0	B1	C0	C1	D0	D1
ADDR OF OPND 2 PLUS L2		OP2	0	L2	0	ADDR OF OPND 1 PLUS L1	

SKEW	H	J	YA	YB	FUNC	YCI	YCD	IZT
	4	2	4	0	00CC	0000				

SS

OP Code	L1	L2	B1	D1	B2	D2
0 7	11	15	19	31	35	47

DECIMAL MULTIPLY DMC SECTION 1E8, ROUTINE 02, STOP ON MS 6192

	ROS	A REG	B REG	C REG	D REG
①	E02	6790	C000	6771	F790
②	E06	678E	000C	670C	5C01
③	E3B	6787	0020	F00C	0000
④	E47	6771	0300	10C2	0060
⑤	E62	6771	5C00	0700	005C
⑥	EB2	6771	0700	0020	5C00
⑦	E72	6771	0200	07FB	5CE0
⑧	4F7	6771	025C	025C	025C

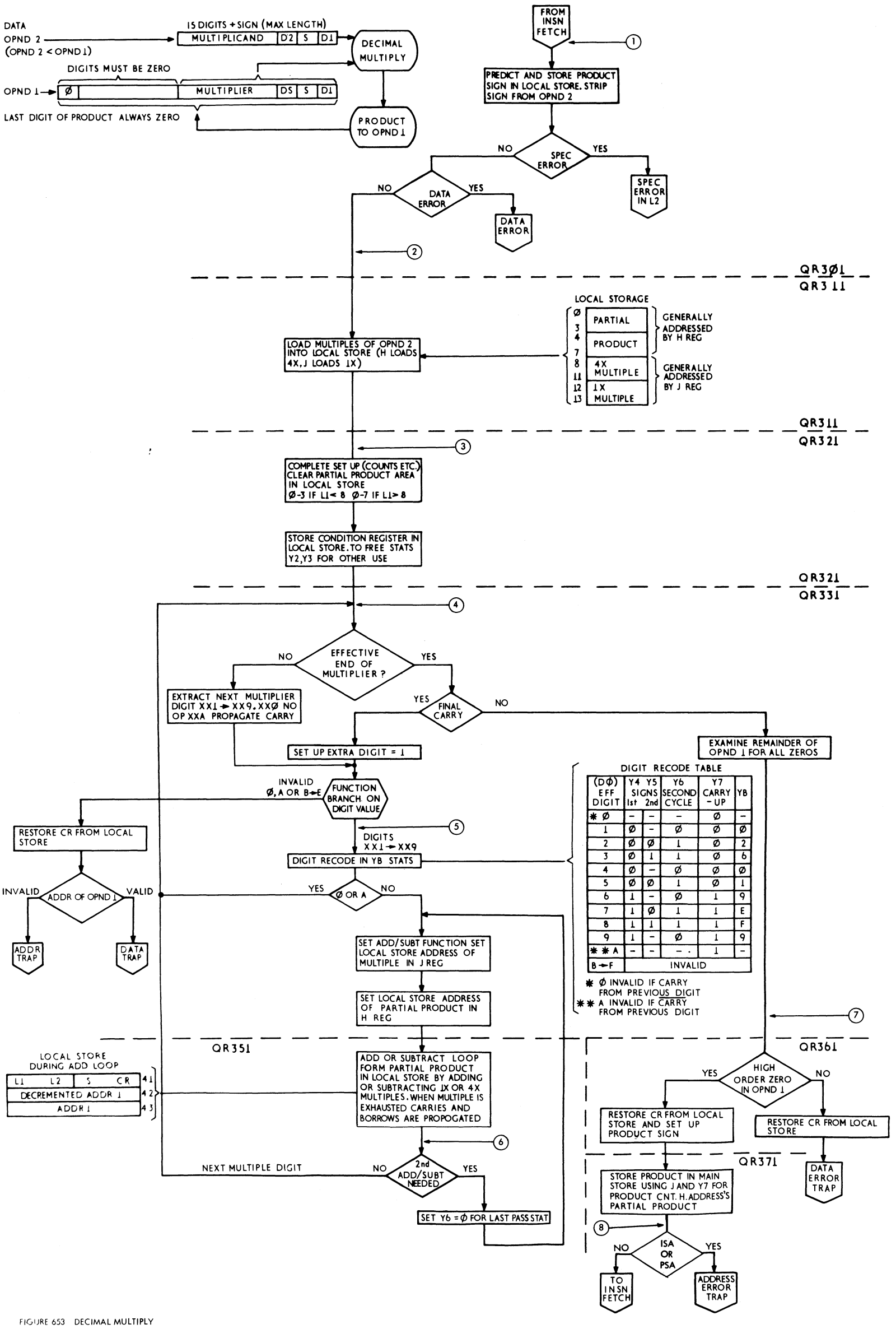


FIGURE 653 DECIMAL MULTIPLY

ENTRY DATA

A0 A1	B0	B1	C0 C1	D0 D1
ADDR OF OPND 2 PLUS L2	OP2 | 0	L2 | 0	ADDR OF OPND 1 PLUS L1	

SKEW	H	J	YA	YB	FUNC	YCI	YCD	IZT
	4	2 | 4	0	00CC	0000			

PACK DMC SECTION 181, ROUTINE 02, STOP ON MS 622C, THEN STOP ON MS 62D2. THIS WILL INSURE THAT THIS IS THE FIRST PASS THROUGH THE LOOP.

	ROS	A REG	B REG	C REG	D REG
①	C05	65CE	2010	65D0	2000
②	866	65CD	0F00	65CB	0000
③	86F	65D0	EF00	65CF	0000
④	841	62D8	EFFF	6501	0000

SS

OP Code	L1	L2	B1	D1	B2	D2
0 7	11	15	19	31	35	47

OBJECTIVE

THE FORMAT OF THE SECOND OPERAND IS CHANGED FROM ZONED TO PACKED AND THE RESULT IS PLACED IN THE FIRST OPERAND LOCATION

Z	d3	Z	d2	SIGN	d1	SOURCE UNPACKED
d3	d2	d1	SIGN			DEST PACKED BCD

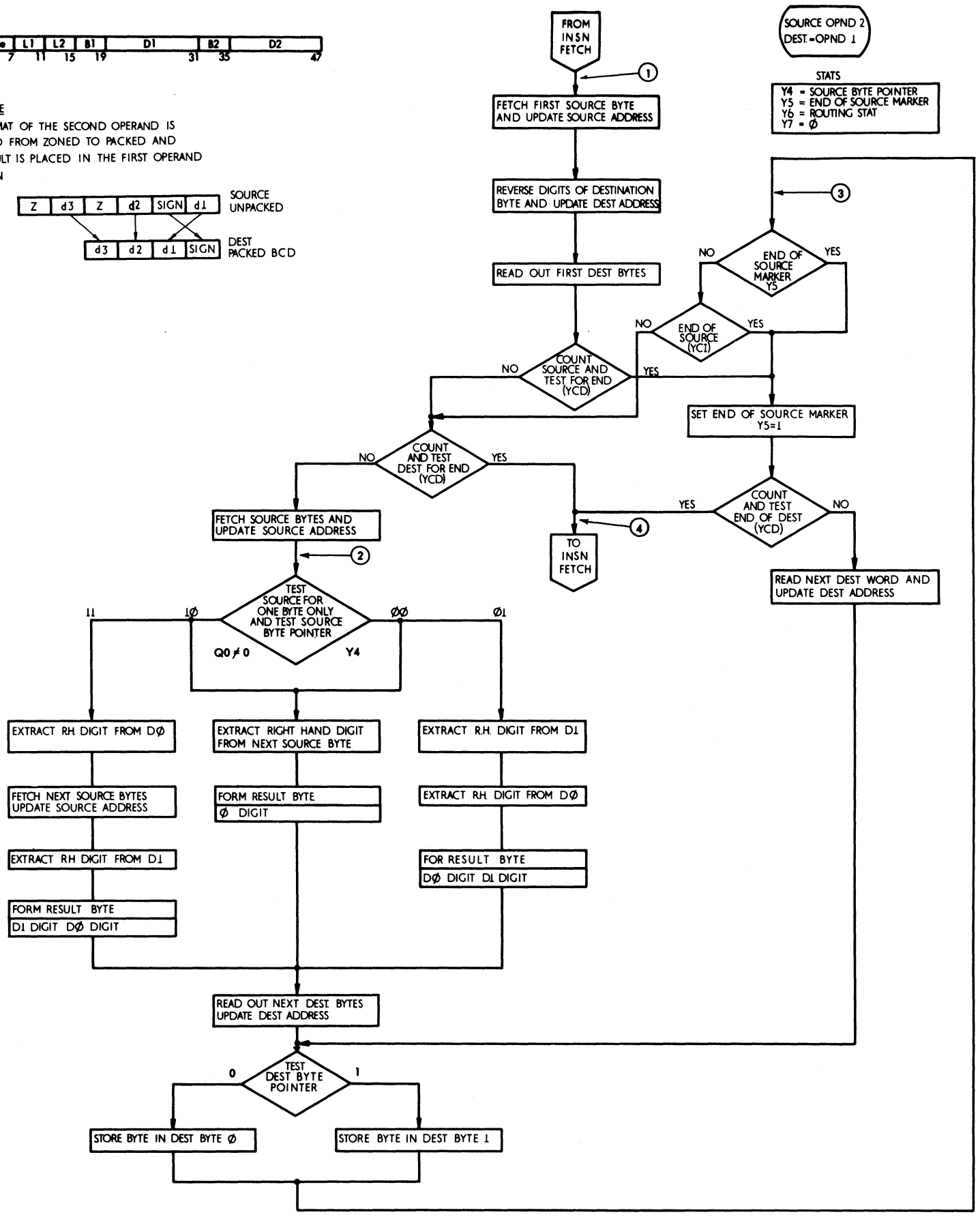
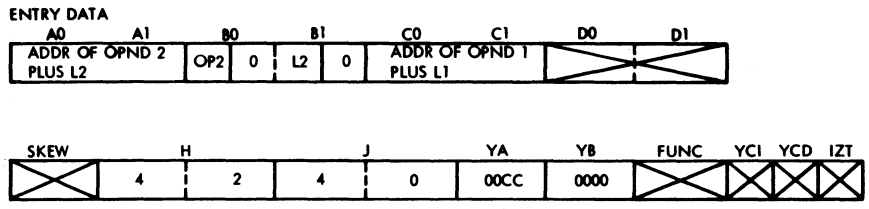
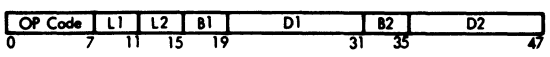


FIGURE 654 DECIMAL PACK (QR101, QR111)

ENTRY DATA



SS



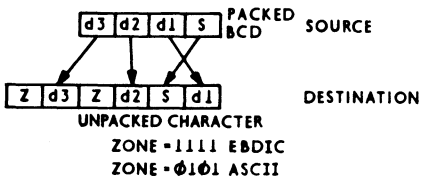
UNPACK DMC SECTION 181, ROUTINE 01, STOP ON MS 60E4, THEN STOP ON MS 618A. THIS WILL INSURE THAT THIS IS THE FIRST PASS THROUGH THE LOOP.

	ROS	A REG	B REG	C REG	D REG
①	C07	65CE	3010	65D0	2000
②	440	65D0	1000	65CF	EF00
③	474	65CD	0000	65CC	0000
④	47F	65CF	F0E0	65CF	FEF0

OBJECTIVE

TO PLACE THE DIGITS AND ZONES OF OPERAND 2 IN UNPACKED FORMAT IN OPERAND 1 USING THE ZONES 1111 FOR EBDIC OR Ø1Ø1 FOR ASCII

DATA FORMAT EXAMPLE



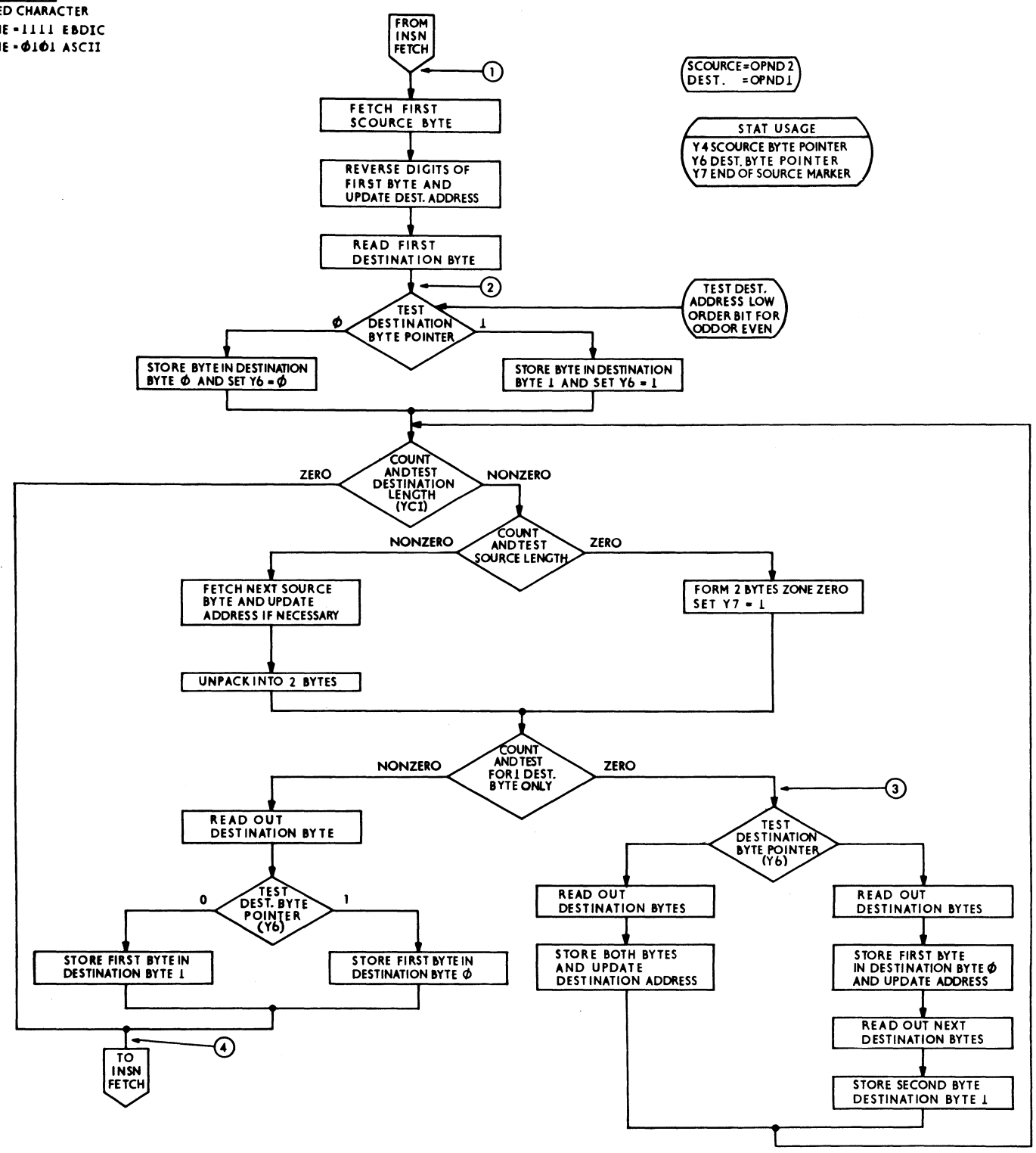
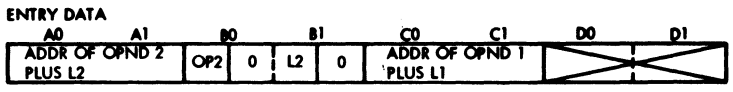
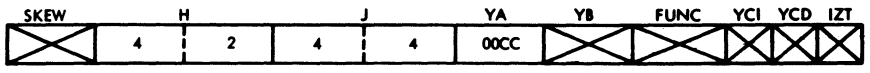


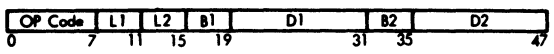
FIGURE 655 DECIMAL UNPACK (QR101, QR121)

ENTRY DATA





SS



OBJECTIVE

THE SECOND OPND IS PLACED TO THE LEFT OF AND ADJACENT TO THE LOW ORDER 4 BITS OF THE FIRST OPND.

NOTE

OPND 1 = DEST
OPND 2 = SOURCE

MOVE WITH OFFSET MVO DMC SECTION 17C, ROUTINE 10, STOP ON MS 658E

	ROS	A REG	B REG	C REG	D REG
①	C03	68C2	1010	699B	F8C1
②	855	699B	2010	699A	0000
③	868	68C1	0000	68C0	0011
④	841	6594	00FF	6901	1220

FIGURE 656 DECIMAL MOVE WITH OFFSET

ENTRY DATA

A0	A1	B0	B1	C0	C1	D0	D1
ADDR OF OPND 2		OP2	0	L2	0	ADDR OF OPND 1	

SKEW	H		J	YA	YB	FUNC	YCI	YCD	IZT
	4	2	4	0	00CC	0000			

SS

OP Code	L1	L2	B1	D1	B2	D2
0 7	11	15	19	31	35	47

NO EXAMPLE SHOWN.

OBJECTIVE

TO SHOW BY MEANS OF A GENERAL FLOW CHART DECIMAL ADD, SUBTRACT, COMPARE AND ZERO ADD.

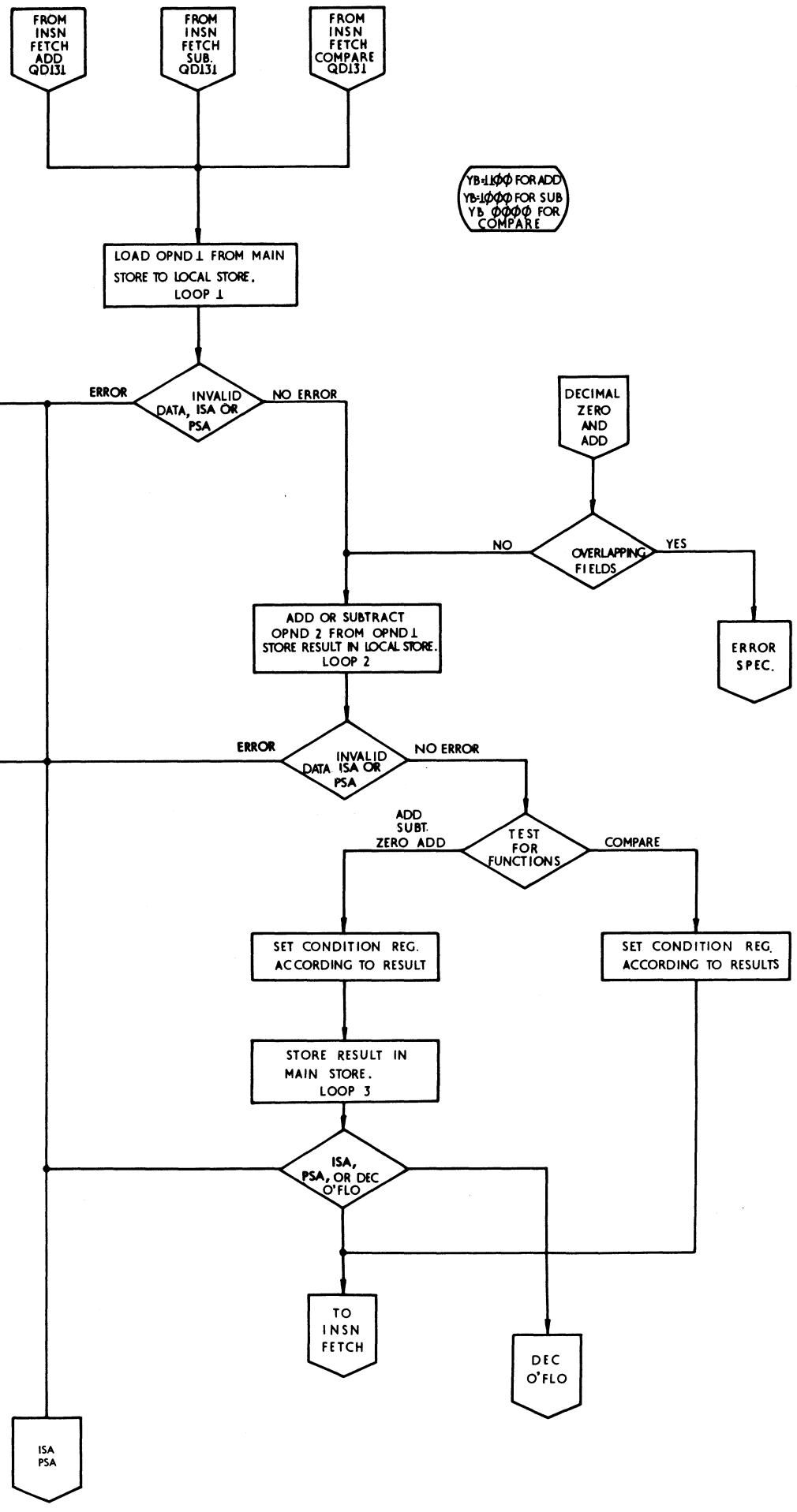


FIGURE 657 GENERAL FLOW CHART FOR DECIMAL ADD, SUBTRACT, COMPARE AND ZERO ADD (QR001, QR021, QR011, QR031)

ENTRY DATA

A0	A1	B0	B1	C0	C1	D0	D1
ADDR OF OPND 2 PLUS L2	OP2	0	L2	0	ADDR OF OPND 1 PLUS L1		

SKEW	H	J	YA	YB	FUNC	YCI	YCD	IZT
	4	2	4	0	00CC	0000				

SS

OP Code	L1	L2	B1	D1	B2	D2
0 7	11	15	19	31	35	47

OBJECTIVE
TO LOAD OPERAND 1 INTO LOCAL STORE
IN PREPARATION TO THE MAIN ADD LOOP
FOR SS DECIMAL COMPARE, SUBTRACT AND ADD

DECIMAL ADD FIGURES 658 , 660, 661 DMC SECTION 1E2, ROUTINE 01, STOP ON MS 60EA

	ROS	A REG	B REG	C REG	D REG
1	C15	6C67	A060	69AE	FC61
2	D6A	69AE	051C	69AC	1CFF
3	D4A	69A8	051C	6910	1010
4	D4B	6C67	6005	0265	000C
	FIGURE 660				
5	D53	6C67	6F10	0265	000C
6	D23	6C67	6F10	024F	0000
7	D16	6C61	6910	10EF	0000
8	D38	6C61	0C10	10EF	0000
	FIGURE 661				
9	D7D	6C61	0C10	10EF	0000
10	D46	69AE	0C0D	104F	1CFF
11	D32	69A8	1010	10A8	1010
12	D5C	69A8	1000	4000	1010

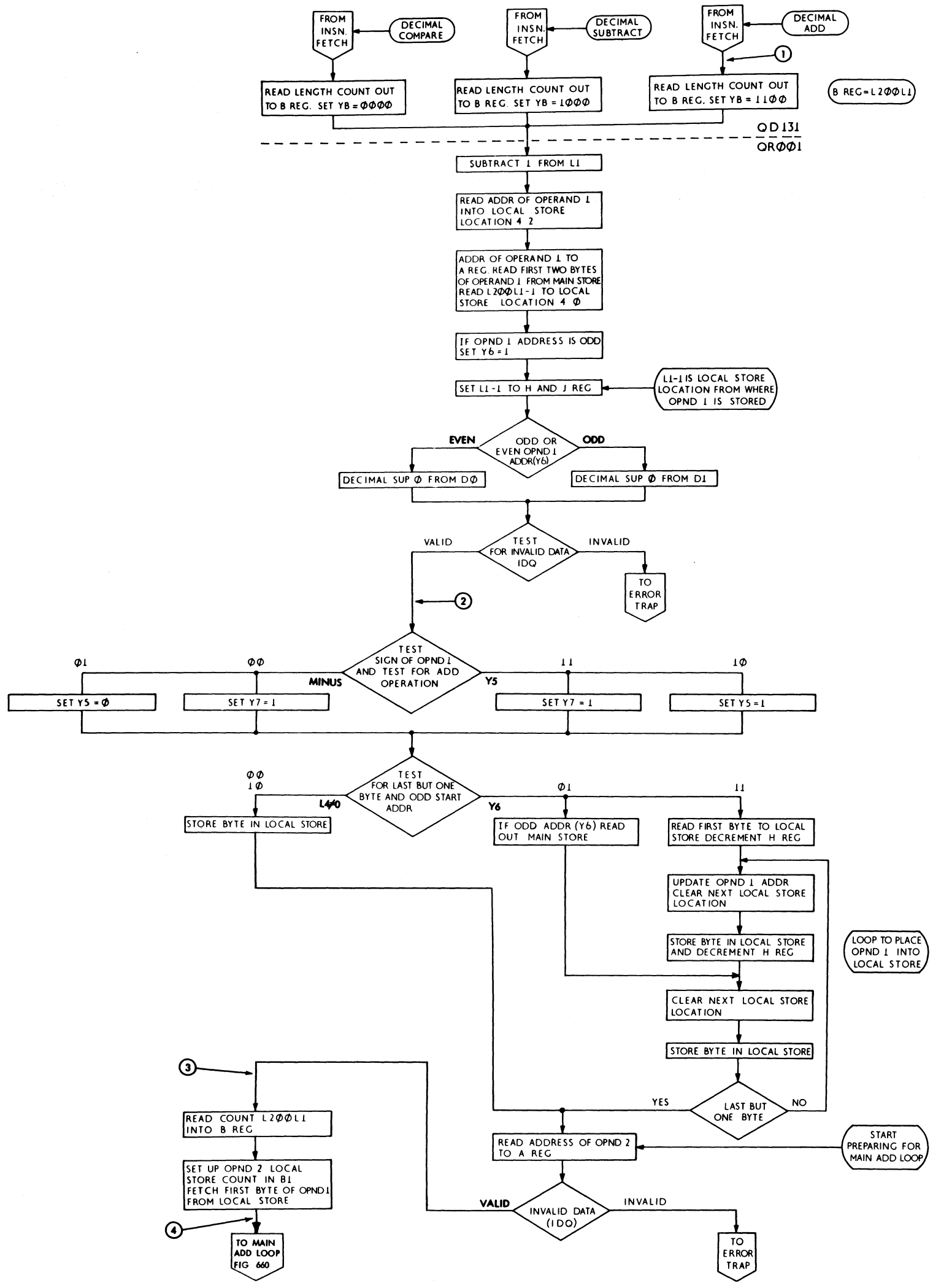


FIGURE 658 SS DECIMAL LOAD OPND 1 (ADD, SUBTRACT, COMPARE) (QD131, QR001)

ENTRY DATA

A0	A1	B0	B1	C0	C1	D0	D1
ADDR OF OPND 2 PLUS L2	OP2	0	L2	0	ADDR OF OPND 1 PLUS L1		

SKEW	H	J	YA	YB	FUNC	YCI	YCD	IZT
	4	2	4	0	00CC	0000				

SS

OP Code	L1	L2	B1	D1	B2	D2
0 7	11	15	19	31	35	47

OPND 1 = DEST
OPND 2 = SOURCE

ZERO AND ADD ZAP DMC SECTION 1E5, ROUTINE 01, STOP ON MS 60F0

	ROS	A REG	B REG	C REG	D REG
①	C11	64A8	8000	6488	F4A8
②	EB1	64A8	0F00	6488	64A8
③	EA2	64A8	0F00	64A7	0C00
④	D28	64A7	0F00	64F0	0C00

OBJECTIVES

TO CHECK FOR OVERLAPPING FIELDS, PROCESS THE SIGN OF OPND 2 AND PROVIDE ENTRY TO SS DECIMAL MAIN ADD LOOP (FIG. 660)

INVALID OVERLAP CASE

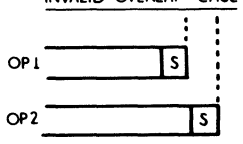


FIGURE 659 SS DECIMAL LOAD ZERO AND ADD ENTRY (QR041)

OBJECTIVE
TO SHOW THE METHOD OF PROCESSING FOR THE DECIMAL OPERATIONS ADD, SUBTRACT,COMP, ZERO AND ADD.
(FOR THE OVERALL FLOWCHART SEE FIGURE 657)

(See Figure 658 for program)

SETTING OF YB STATS
11ØØ FOR ADD
1ØØØ FOR SUBT
ØØØØ FOR COMP.

OPND 1 = DEST
OPND 2 = SOURCE

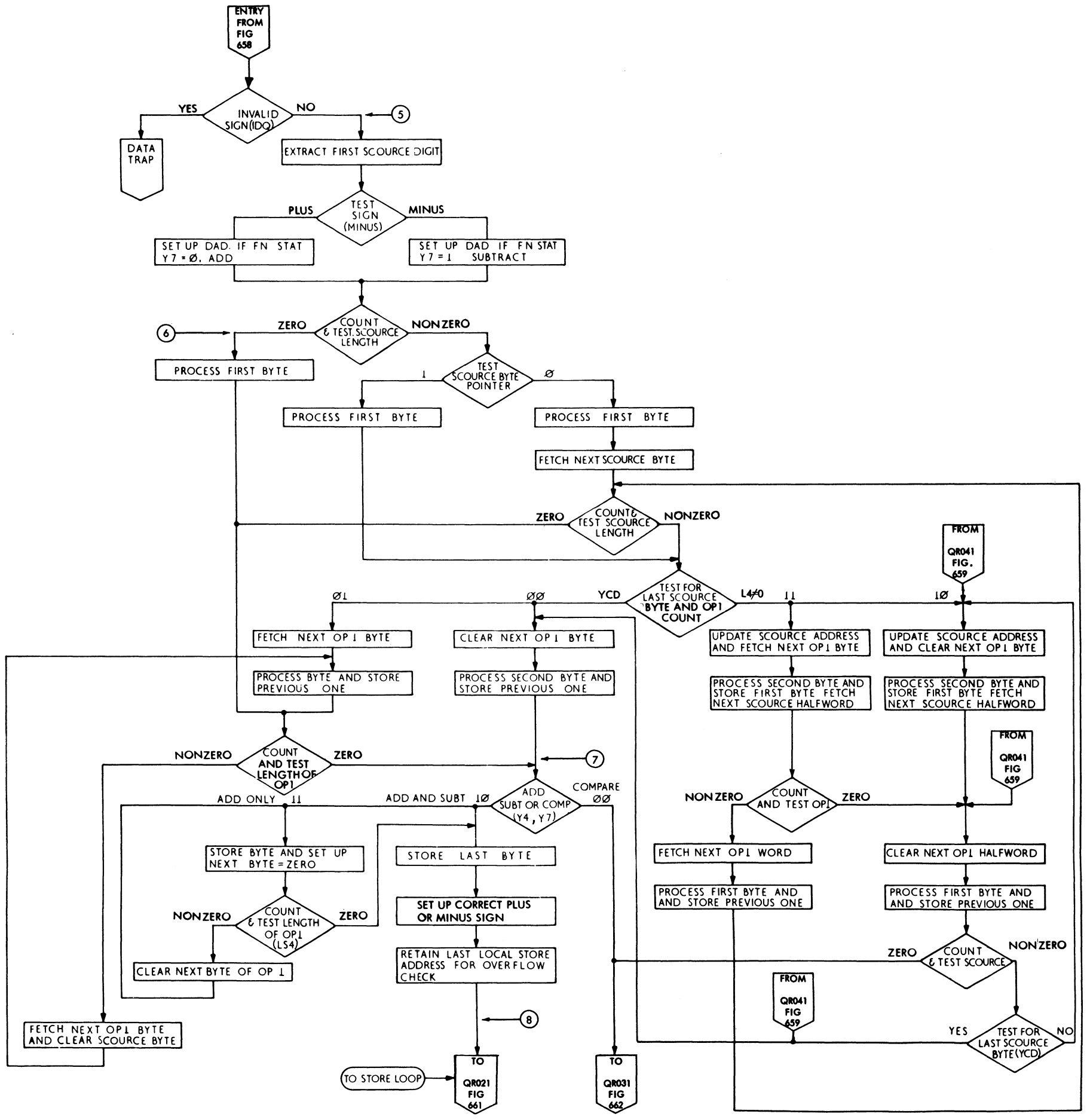


FIGURE 660. SS DECIMAL LOAD OPND 2 AND PROCESS (ADD, SUBT,COMP, ZERO AND ADD) (QR011)

(See Figure 658 for program)

OBJECTIVE

TO STORE RESULT OF THE DECIMAL ADD, SUBTRACT OR ZERO ADD IN THE MAIN STORE LOCATION OF OPND.⊥ AND SET CONDITION REGISTER ACCORDING TO RESULT.

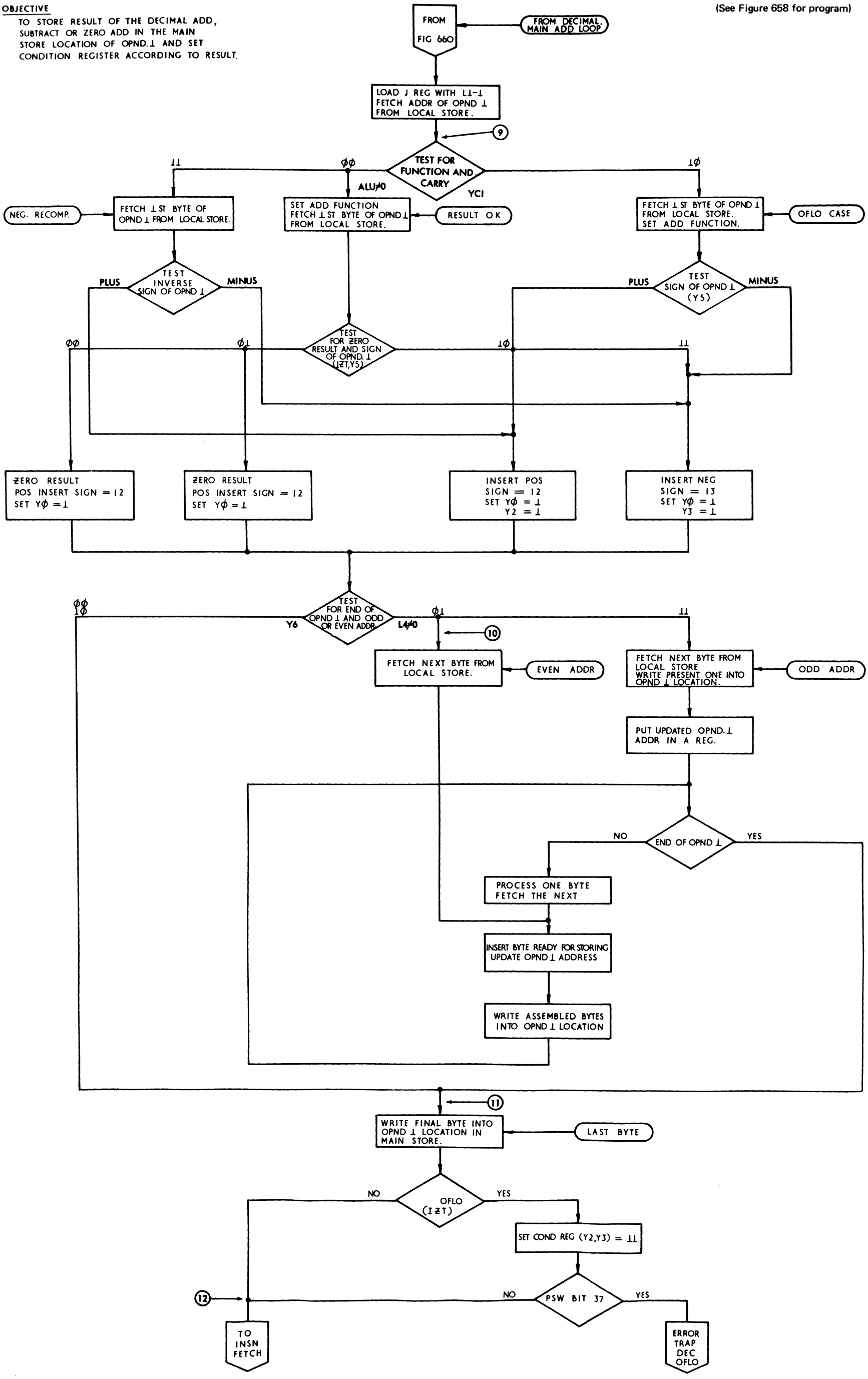


FIGURE 661.SS DECIMAL TERMINATE (ADD SUBTRACT AND ZERO AND ADD) (QR021)

DECIMAL COMPARE DMC SECTION 1E4, ROUTINE 01,
STOP ON MS 60F0

	ROS	A REG	B REG	C REG	D REG
①	C13	6358	9000	6348	F358
②	D7E	6348	0F0F	6346	0C00
③	D18	6356	0F00	00F0	0099
④	D3C	6116	0F00	0018	4780

OBJECTIVE

TO SHOW THE EXIT TO I FETCH AND SETTING OF CONDITION REGISTER FOR SS DECIMAL COMPARE

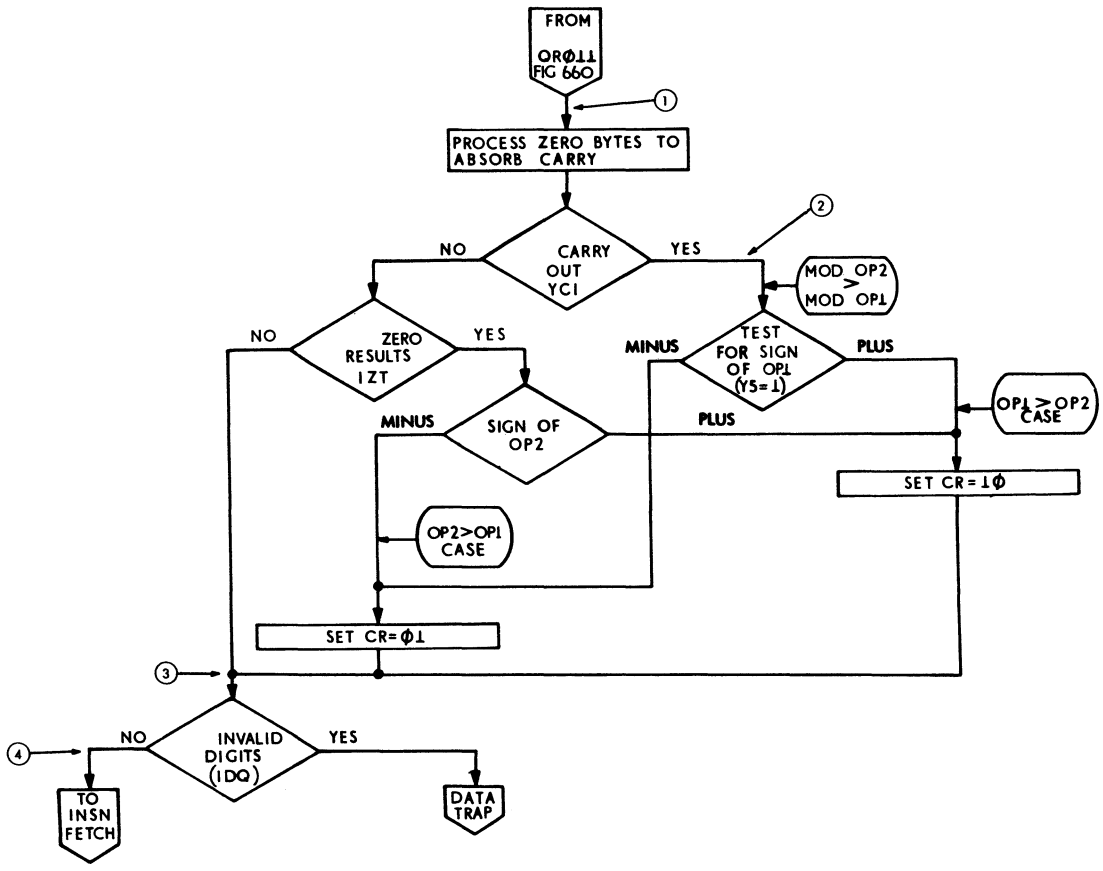


FIGURE 662 SS DECIMAL COMPARE (QR001)

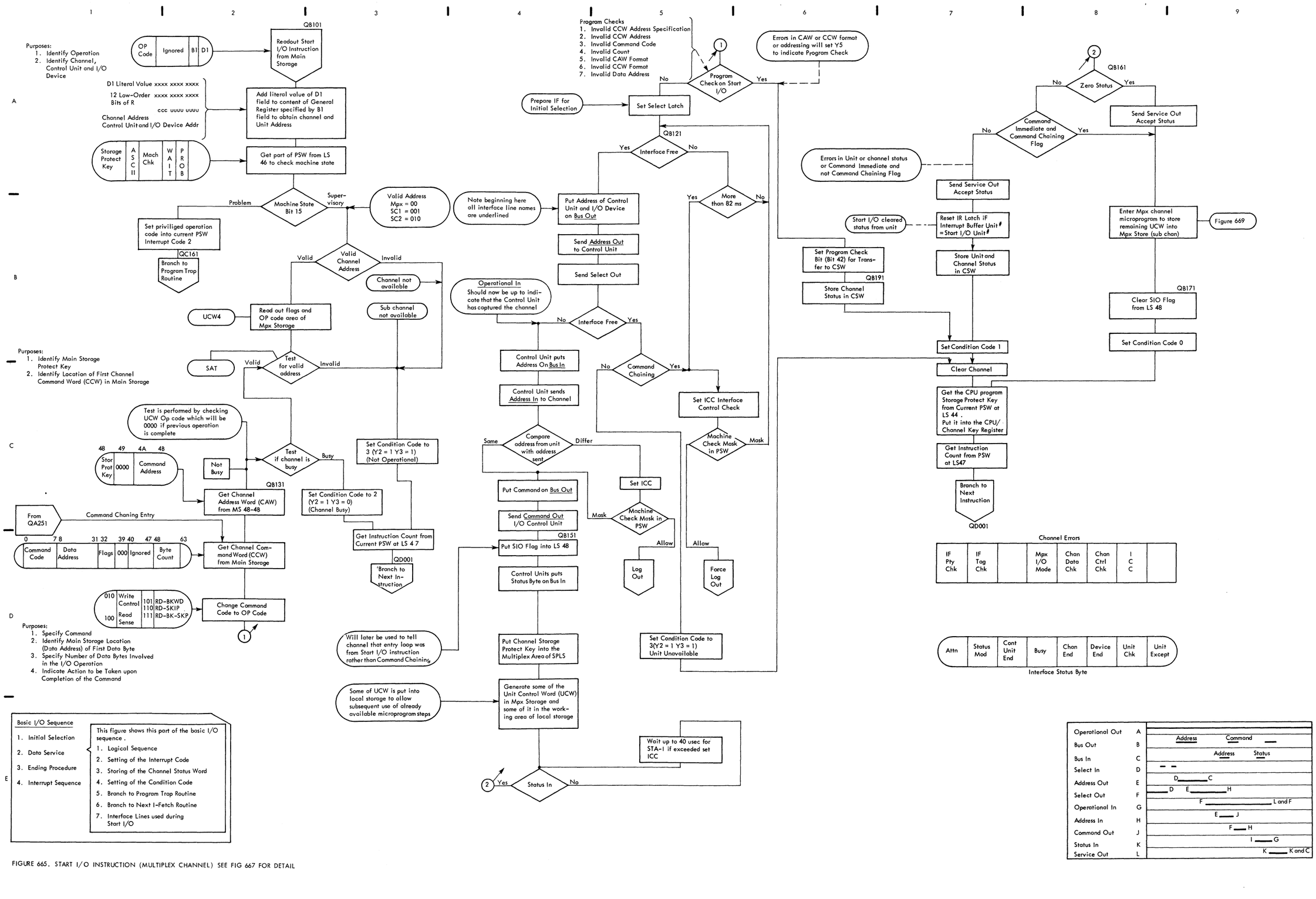


FIGURE 665. START I/O INSTRUCTION (MULTIPLEX CHANNEL) SEE FIG 667 FOR DETAIL

Condition Code Setting on Multiplex Halt I/O

Cond Code	Action on CSW	Notes
0	Unchanged	Channel busy with interrupt
1	Unit Status only is stored. Channel Status is zero	Channel Busy or Channel not busy but control unit busy
	Channel and unit status is zero	Channel and Control Unit not busy (Normal CU), or: Channel not busy – device is stopped by HIO command (special CU)
2	Condition Code 2 is not set on the Multiplex Channel Halt I/O	
3	Unchanged	Invalid Channel or UCW address
	Unchanged	Unit does not reply to initial selection – it is stopped the next time it requests service

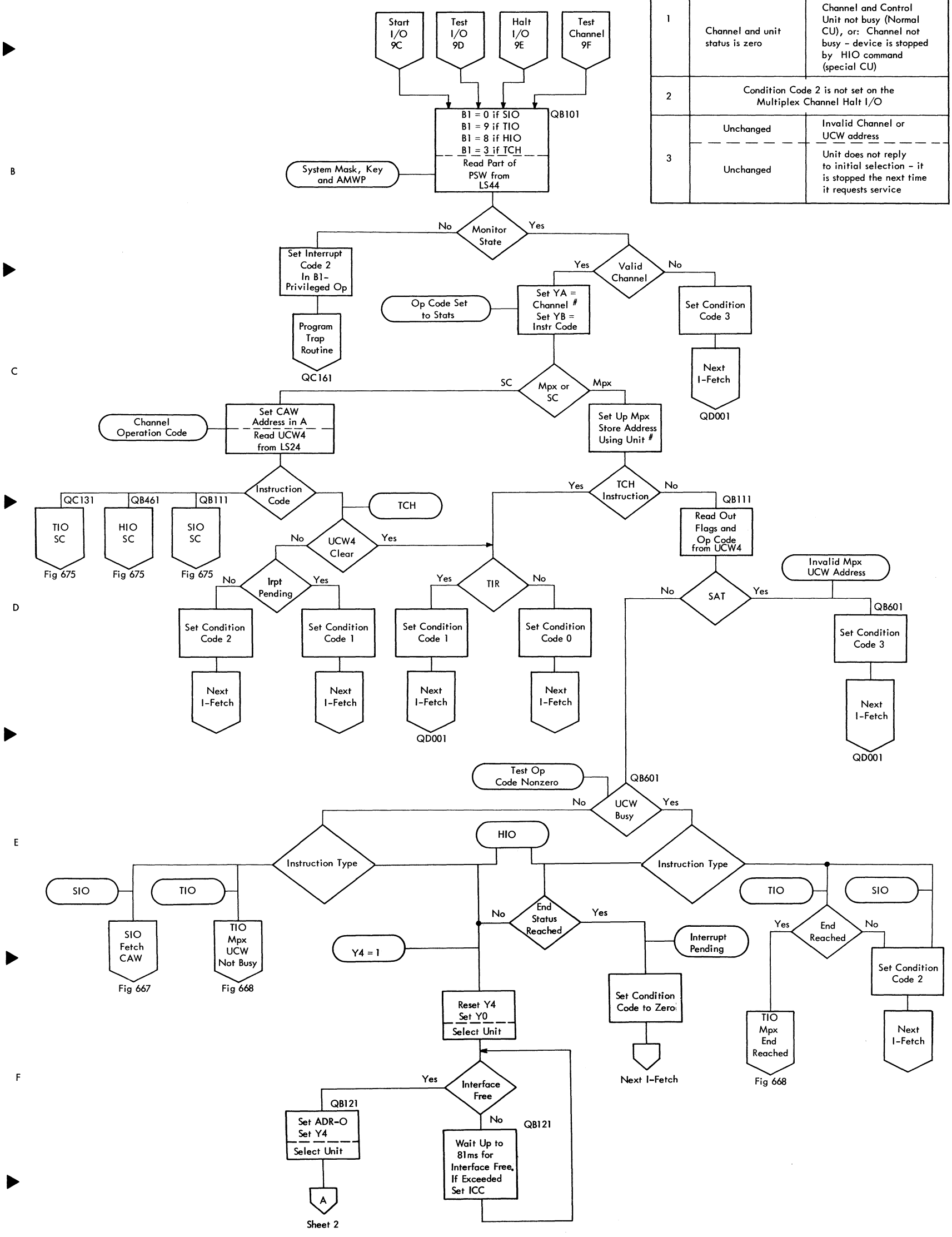
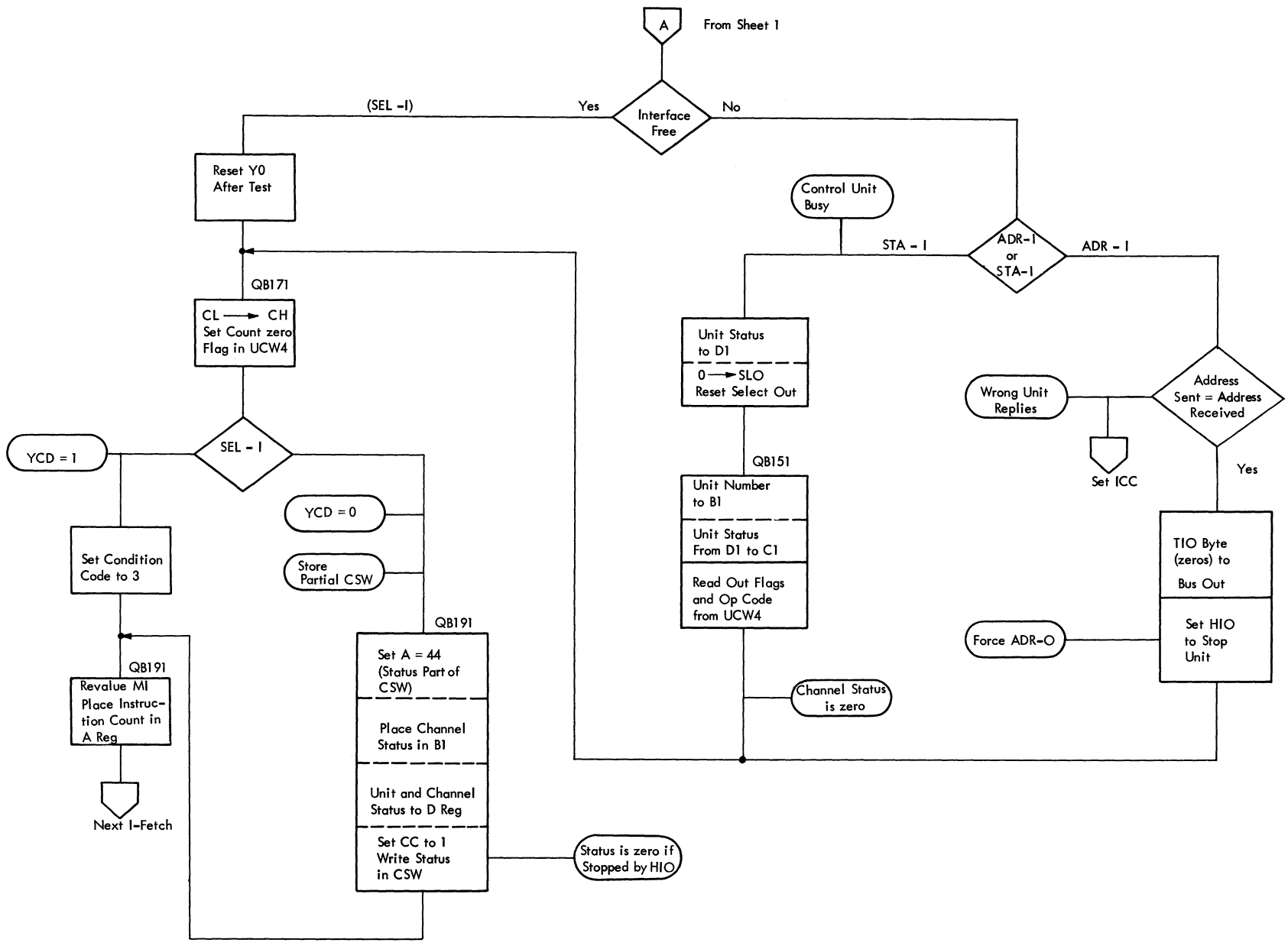
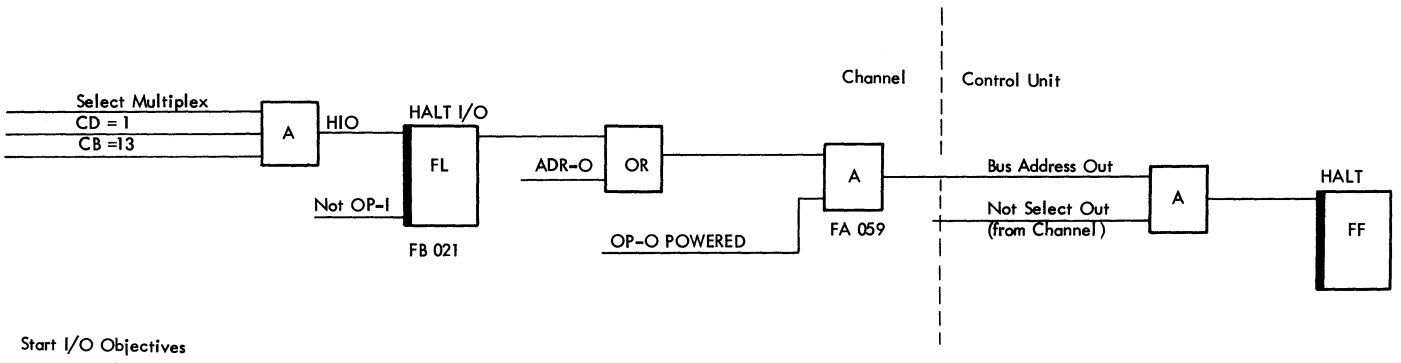


FIGURE 666. I/O CODES, COMMON DECODING; TEST CHANNEL AND MPX HALT I/O (SHEET 1 OF 2)





Start I/O Objectives

Figure	Objective
666-1	1. Test for UCW busy.
667-1	2. Fetch CAW. 3. Fetch CCW. 4. Begin initial selection. 5. Store partial UCW (UCW6 and UCW8). 6. Complete initial selection.
667-2	7. Examine status. 8. Subtract 1 from count.
669-1	9. Test for burst mode.
669-4	10. Add 1 to count. 11. Store remaining UCW into multiplex storage (Y4 on).
667-2	12. Set condition code. 13. Next I-fetch.

FIGURE 666. I/O CODES, COMMON DECODING; TEST CHANNEL AND MPX HALT I/O (SHEET 2 OF 2)

Condition Code Setting on Start I/O

Cond Code	Action on CSW	Notes
0	Unchanged	Operation Successfully Initiated at the Device
1	CSW Stored	TIC on SIO - Program Check on SIO - Control Unit Busy - Command Immediate Errors - Interrupt Conditions
2	Unchanged	Path Busy
3	Unchanged	Channel or Device Unavailable

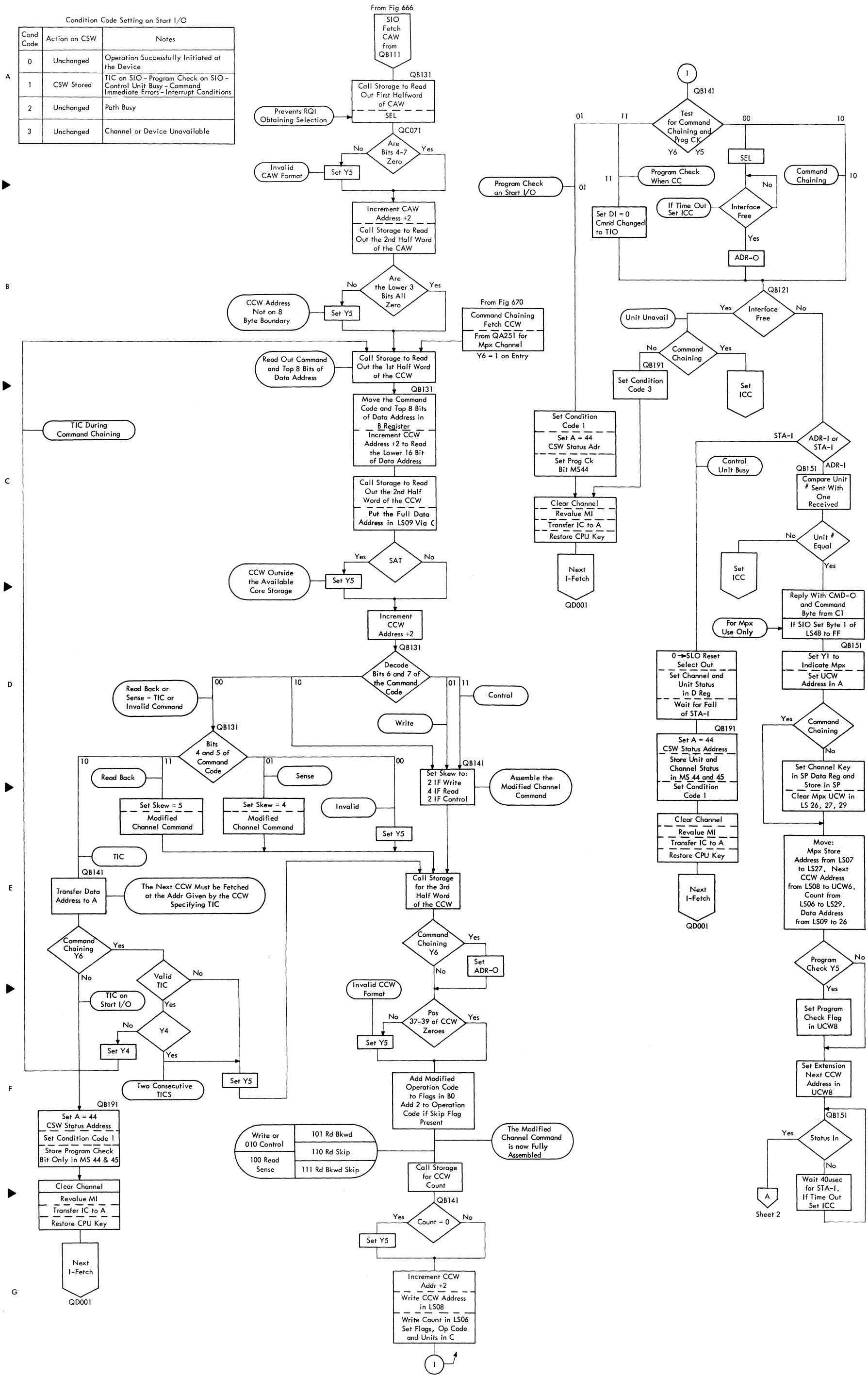


FIGURE 667. START I/O MICROPROGRAM MPX CHANNEL (SHEET 1 OF 2)

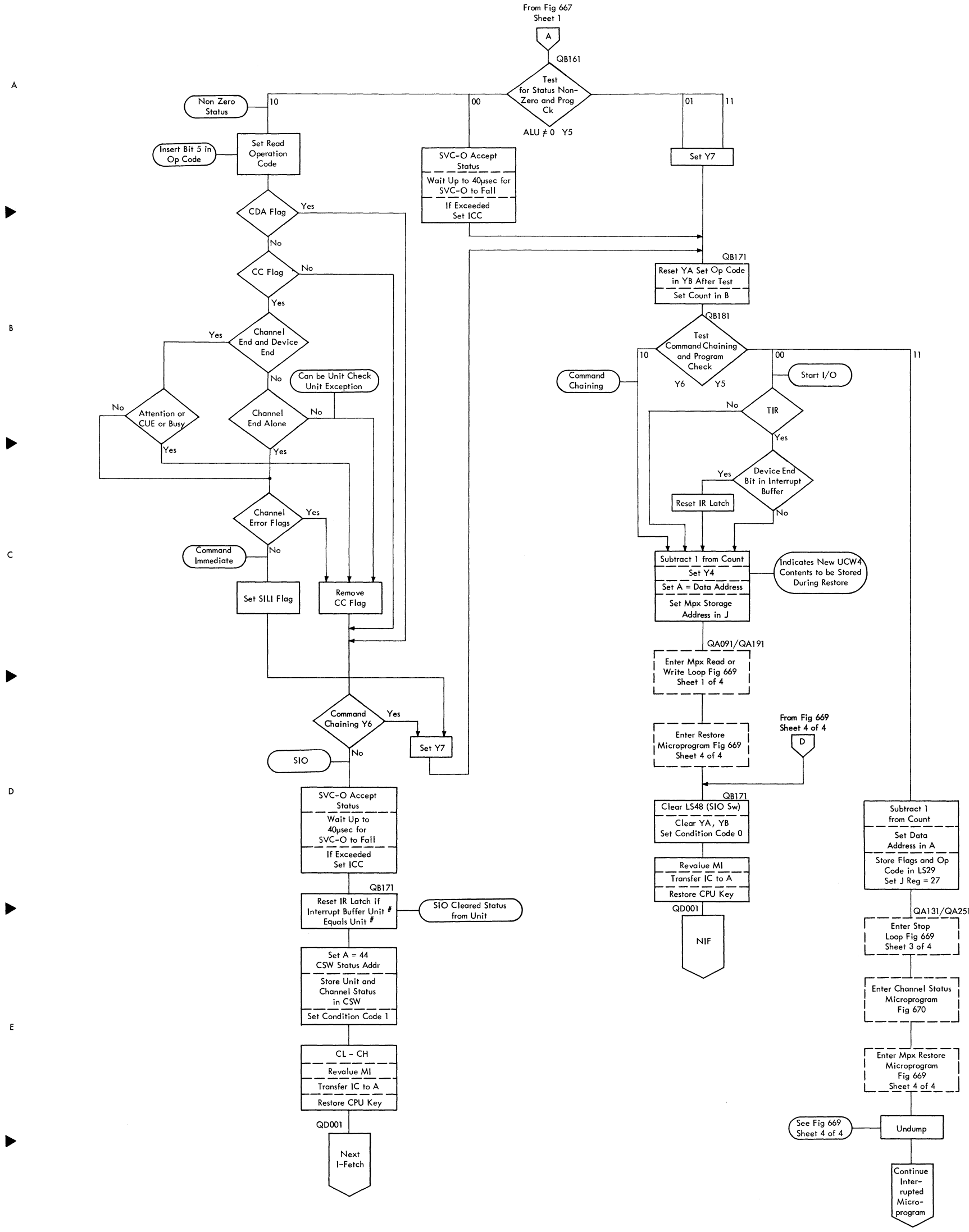
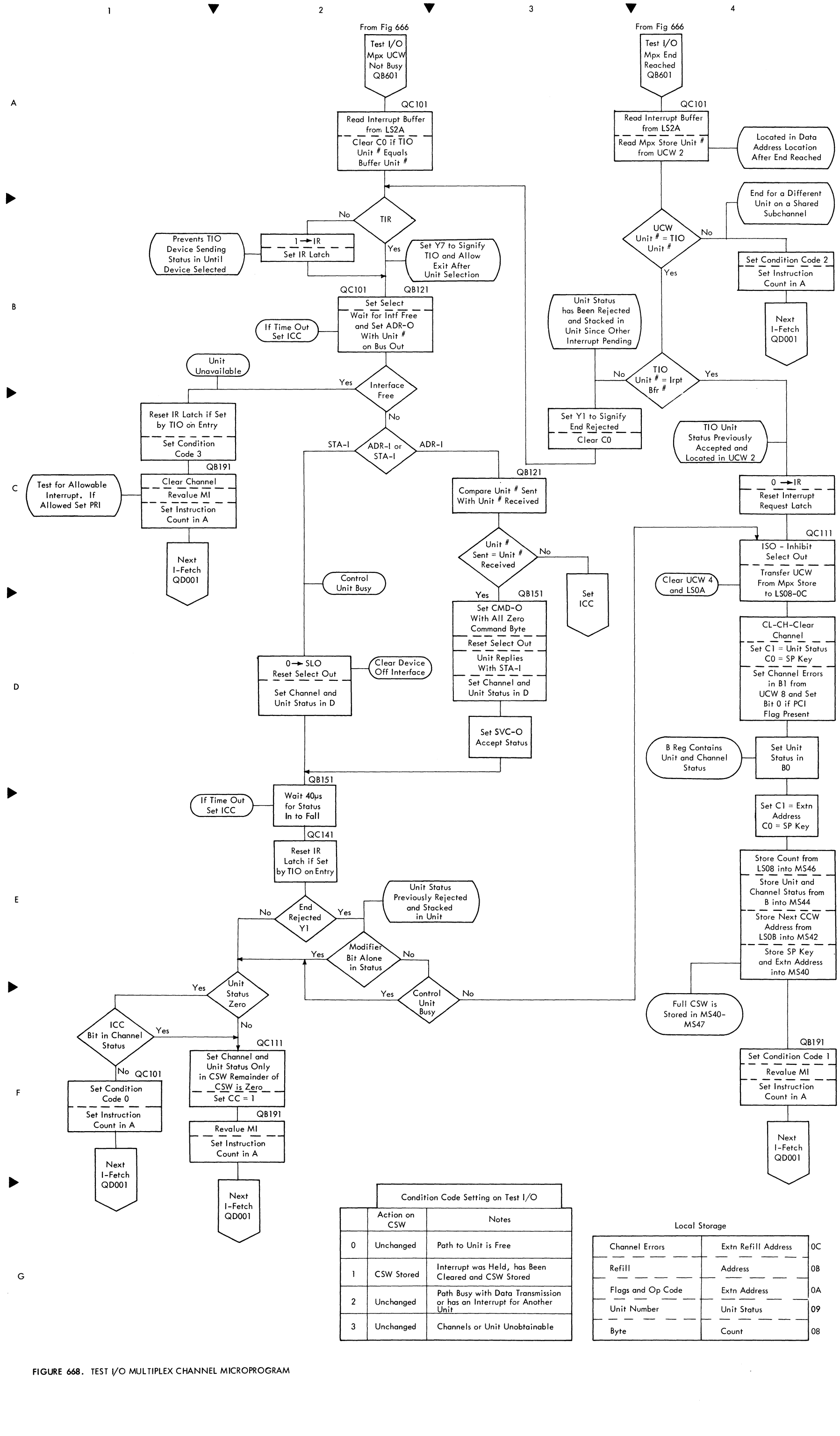


FIGURE 667. START I/O MICROPROGRAM MPX CHANNEL (SHEET 2 OF 2)



Condition Code Setting on Test I/O

	Action on CSW	Notes
0	Unchanged	Path to Unit is Free
1	CSW Stored	Interrupt was Held, has Been Cleared and CSW Stored
2	Unchanged	Path Busy with Data Transmission or has an Interrupt for Another Unit
3	Unchanged	Channels or Unit Unobtainable

Local Storage

Channel Errors	Extn Refill Address	0C
Refill	Address	0B
Flags and Op Code	Extn Address	0A
Unit Number	Unit Status	09
Byte	Count	08

FIGURE 668. TEST I/O MULTIPLEX CHANNEL MICROPROGRAM

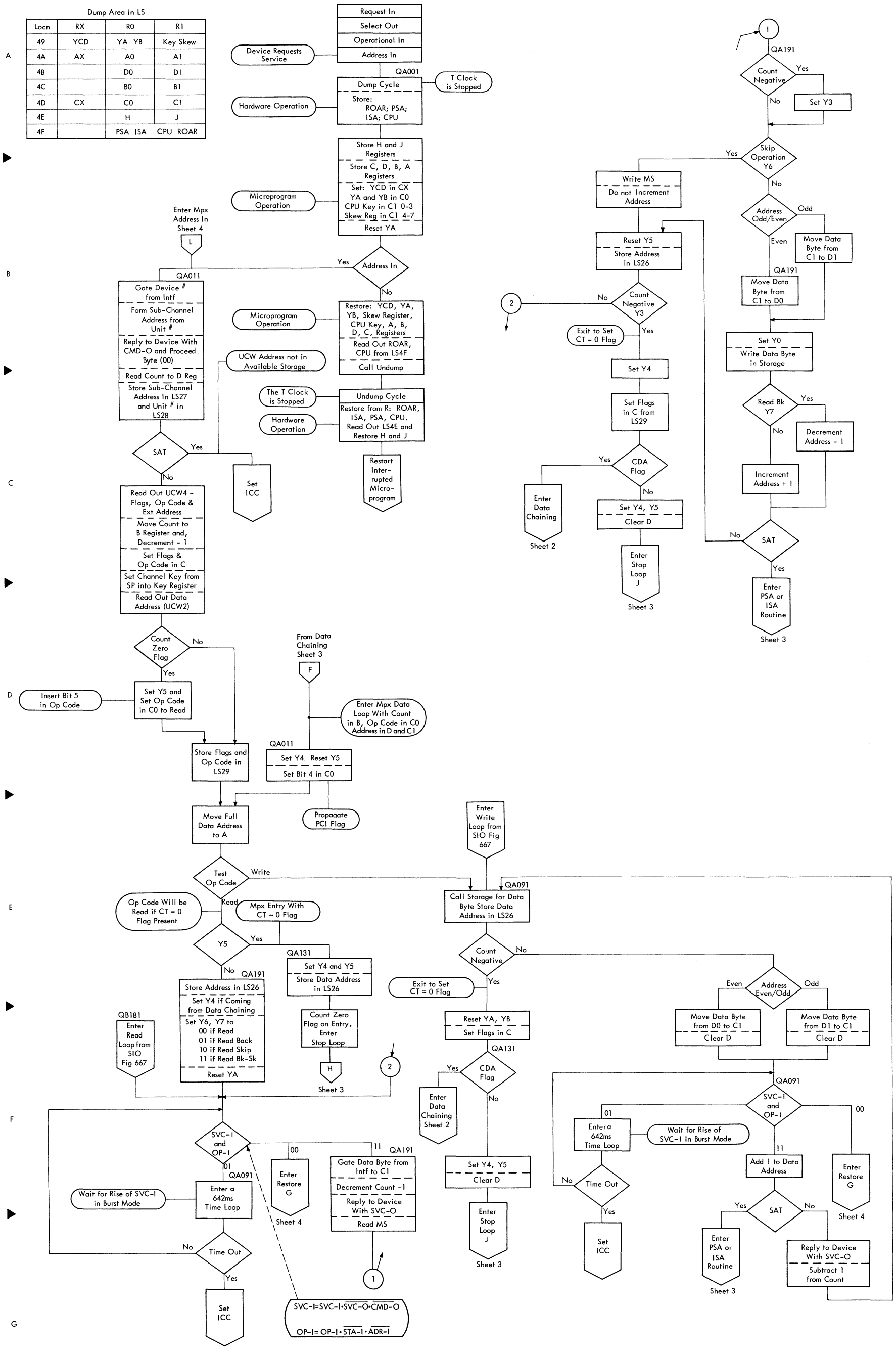
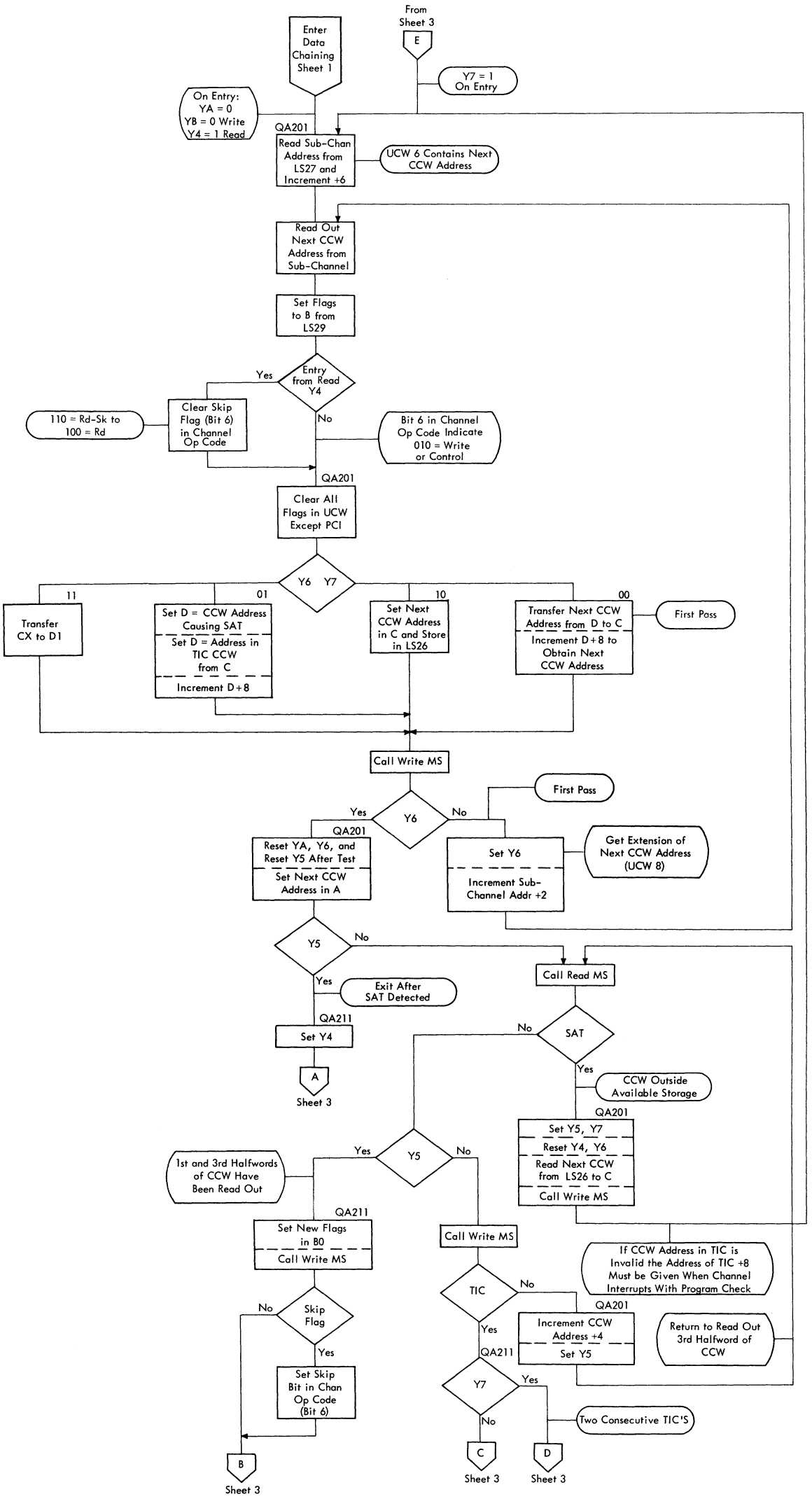


FIGURE 669. MULTIPLEX CHANNEL MICROPROGRAM (SHEET 1 OF 4)

9 10 11 12



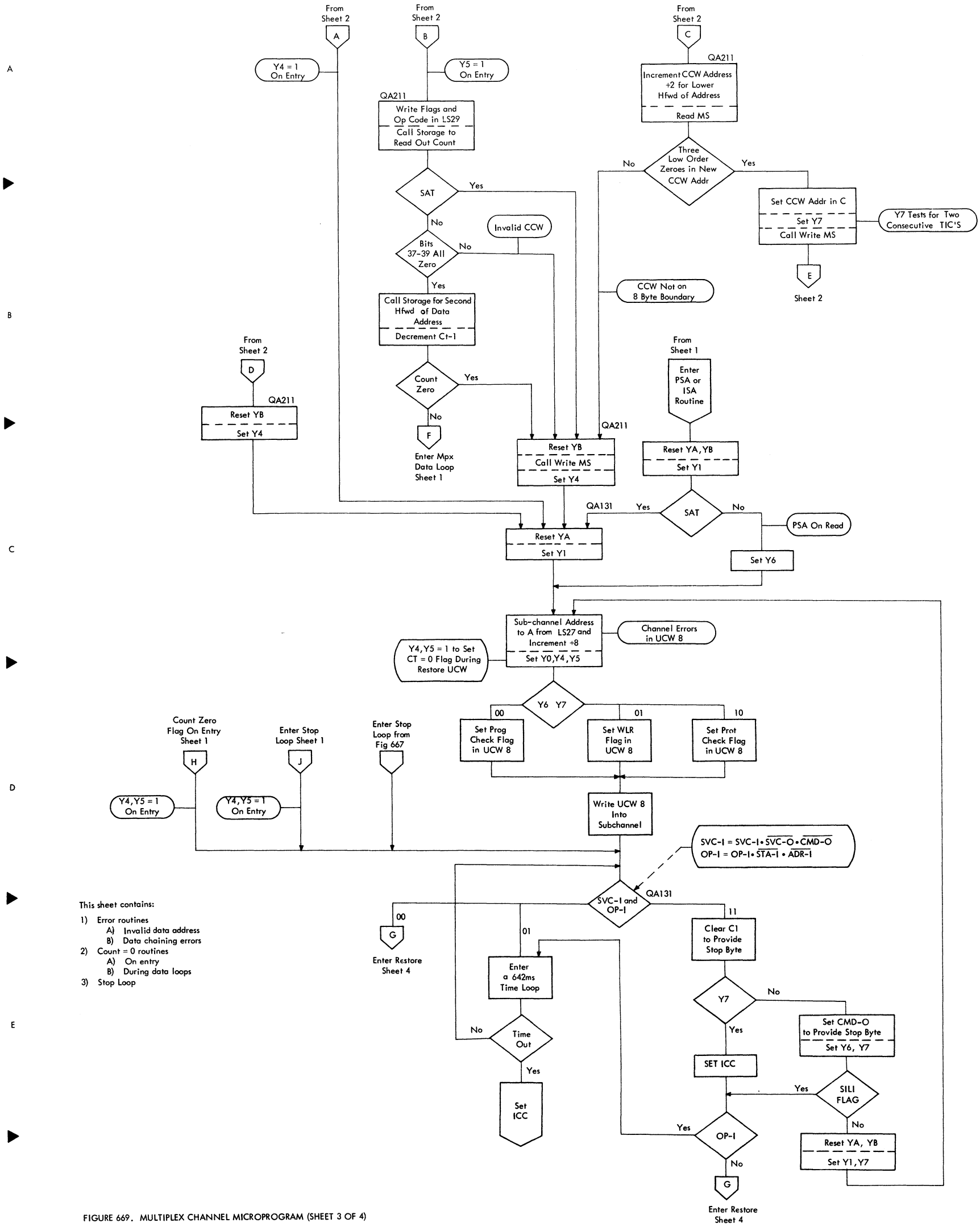
This sheet starts data chaining routine--

1) Fetch UCW addr and incr to loc of next CCW
2) Fetch next CCW address
3) Repeat UCW fetch for ext byte of CCW addr
4) Incr CCW addr by 8 for following CCW addr and rewrite in Mpx store UCW
5) Fetch 1st hfwd of CCW
6) Test for TIC command
7) If TIC, fetch rmdr of data addr (is next CCW addr) incr this addr by 8 and store in mpx store UCW
8) If no TIC continue fetch CCW in order 3rd, 4th, 2nd hfwds and test errors (all on QA211)
9) Then fetch new CCW, test TIC
10) If no second TIC, do as step 8
11) If second TIC, exit to error routine

Sub-Channel Format

UCW8		WLR	Prog Ck	Prot Ck			ICC				Extention Next CCW Address					
UCW6							Next CCW Address									
UCW4	CDA	CC	SILI	SKIP	PCI		Op Code		CT=0	End Stat Rch	Extention Data Address					
UCW2			Unit Number at the End Time					Data Address			Unit Status at End Time					
UCW0								Count								

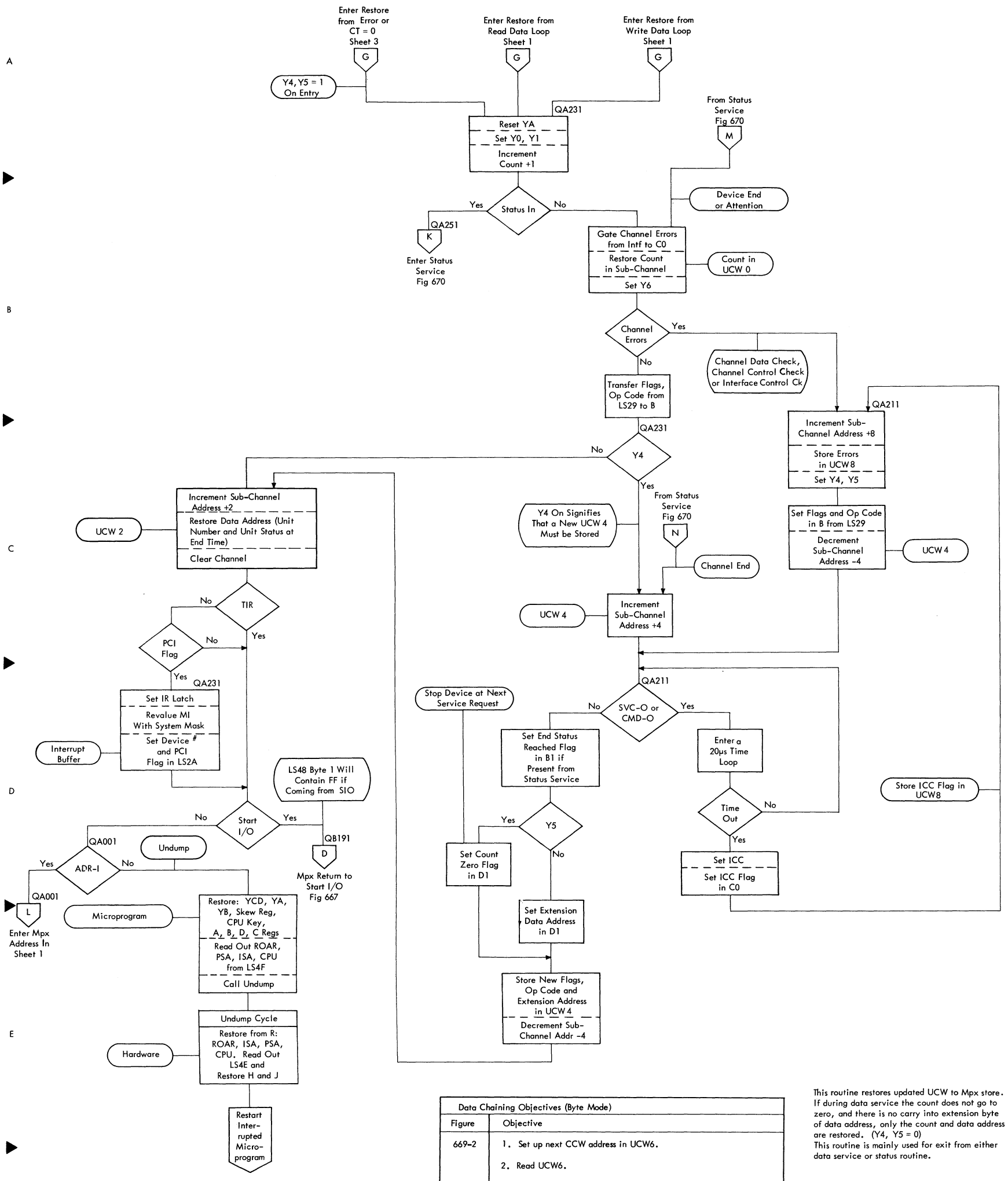
FIGURE 669. MULTIPLEX CHANNEL MICROPROGRAM (SHEET 2 OF 4)



This sheet contains:

1) Error routines
   - A) Invalid data address
   - B) Data chaining errors
2) Count = 0 routines
   - A) On entry
   - B) During data loops
3) Stop Loop

FIGURE 669. MULTIPLEX CHANNEL MICROPROGRAM (SHEET 3 OF 4)



Data Service Objectives (Byte Mode)	
Figure	Objective
669-1	1. I/O unit initiates selection with request-in. 2. Dump CPU. 3. Subtract 1 from count. 4. Test count zero flag. 5. Test for read op code. 6. Subtract 1 from count. 7. Access main storage.
669-4	8. Add 1 to count. 9. Update UCW (Y4 and Y5 off except for last byte). 10. Undump. 11. Continue interrupted microprogram.

Data Chaining Objectives (Byte Mode)	
Figure	Objective
669-2	1. Set up next CCW address in UCW6. 2. Read UCW6. 3. Set up extension address in UCW8. 4. Read UCW8. 5. Assemble full CCW address in C register. 6. Read first halfword of CCW. 7. Test for TIC command. a. If second consecutive TIC (Y7 on), exit to error routine.
669-3	b. If first Tic: • Read second halfword (next CCW address), increment address +8, and store in UCW6 and UCW8.
669-2	• Fetch new CCW (step 1). c. If not a TIC, read out third halfword of CCW.
669-3	8. Read fourth halfword of CCW (Y5 on). 9. Read second halfword of CCW (Y5 on).
669-1	10. Reset Y5 and set Y4 11. Test for burst mode.
669-4	12. Add 1 to count. 13. Store new data in UCW0, UCW2, and UCW4. (Y4 on). 14. Undump.

This routine restores updated UCW to Mpx store. If during data service the count does not go to zero, and there is no carry into extension byte of data address, only the count and data address are restored. (Y4, Y5 = 0)
This routine is mainly used for exit from either data service or status routine.

FIGURE 669. MULTIPLEX CHANNEL MICROPROGRAM (SHEET 4 OF 4)

1 2 3 4

A

Mpx Input Bus

0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
IF Parity	IF Tag Check		Mpx I/O Mode	Channel Data Check	Channel Control Check	IF Contr . Check		Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7

Byte 0 = Channel Errors — Byte 1 = Data

Interface Status Byte

0	1	2	3	4	5	6	7
Atn	Status Mod	Ctrl Unit End	Busy	Channel End	Device End	Unit Check	Unit Except

B

Mpx Local Storage Working Space

Current Data Address															26
Mpx Store Address															27
							Bus - In Unit Number								28
CDA	CC	SILI	SKIP	PCI	Operation Code		CT = 0	End Stat Rch	Extension Data Address						29
				Chan Ctrl Check On Log Out	Atn Or Dev End	END	PCI	Interrupt Buffer Unit Number							2A

C

D

E

Figure	Objectives
670	1. Test for command chaining. 2. Accept status. 3. Set Y6 status.
667-1	4. Fetch new CCW. (step 4 SIO objectives)

MPX Store Format

UCW8		WLR	Prog Check	Prot Check			ICC	Extension Next CCW Address							
UCW6	Next CCW Address														
UCW4	CDA	CC	SILI	SKIP	PCI	Operation Code		CT = 0	End Stat Rch	Extension Data Address					
UCW2	Unit Number At End Time							Data Address	Unit Status At End Time						
UCW0	Count														

F

From Fig 669 Sheet 4

K

QA251

Channel OP Code Zero

Yes: Device End or Attention Type Status

No: UCW 8 — Transfer Mpx Storage Address to A and Increment +8; Read Out Mpx Channel Errors; Set Channel and Unit Status in B

From Intf Status Byte

Count = 0 — Yes / No

Set WLR Flag in B0 if: CDA Flag Present or $\overline{CDA} \cdot \overline{SILI}$

QA251

Set in D0 Any Channel Errors from B0

Restore Channel Errors from D to UCW 8

Any Errors in D0 — No / Yes

Command Chaining — Yes / No

Device End — Yes / No

QA251

DE Alone or DE With Channel End and or Modifiers — Yes / No

Channel End Alone — No / Yes

Unit Check or Unit Exception

UCW 6

Set A to Next CCW Address; Set SUP-O to Indicate CC; Set Y6; Propagate PCI Flag if Present; Set ISO

Prevents:
1. SEL-O in Reply to ADR-I
2. Dump on ADR-I
Allow: SEL-O With ADR-O

Set SVC-O Accept Status; Read Mpx Store and Update CCW Address + 8 if Modifier Bit Present; Save PCI Bit and Unit # in LS26

CCW Jump

Enter SIO for Command Chaining

B1 = CCW Address Extn
D = CCW Address

Fetch CCW
Figure 667

Bits Other Than CE, DE or Modifiers

Set Mpx Store Address in A; Set Y5, Y6, Y7; Set SUP-O to Indicate Command Chaining; SVC-O Accept Status

QA261

Set Flags in B0

Set Bit 6 (End Bit) in A0 and Unit Number in A1

TIR — No / Yes

Read Interrupt Buffer from LS2A

QA261

Interrupt Buffer Unit # = Unit # on Intf — No / Yes

Interrupt from Another UCW

End Status Reached Flag Present — Yes / No

Set End Bit and Unit # in Interrupt Buffer

Set Mpx Storage Address in A; Set Y5, Y6, Y7; CMD-O Reject Status; Restore Interrupt Buffer

Channel End — Yes / No

Set Unit # and Unit Status to LS26

SVC-O Accept Status; Set Mpx Storage Address in A; Set Y5, Y6, Y7

Set End Reached Flag in B1; Remove CC Flag from B0

QA231

N

To Fig 669 Sheet 4

TIR — Yes / No

QA261

Set System Mask in D; I → IR; Set IR Latch

Set Bit 5 (Device End or Attention) In A0 and Unit Number in A1

Set Interrupt Buffer from A

Channel Mask Off — Yes / No

Set MI STAT Y9

Op Code and Flags to LS29

QA261

Set System Mask in D; I → IR; Set Interrupt Request Latch

Set Interrupt Buffer from A

Channel Mask Off — No / Yes

Set MI STAT Y9

QA261

CMD-O Reject Status; Set Mpx Store Address in A; Set Y4

M

Fig 669 Sheet 4

FIGURE 670. MULTIPLEX CHANNEL STATUS

	I/O Interrupts Contents of CSW Stored
PCI	Treated as an end type interrupt– A full CSW is stored but unit status is zero.
END	A full CSW is stored
Device End	Device is selected to obtain in status. Only the unit and channel status appear in the CSW. The remainder of the CSW is zero.
Channel Control Check or Machine Chk.	Treated as an end type interrupt. The CSW contains unit and channel status only. The remainder of the CSW is zero.

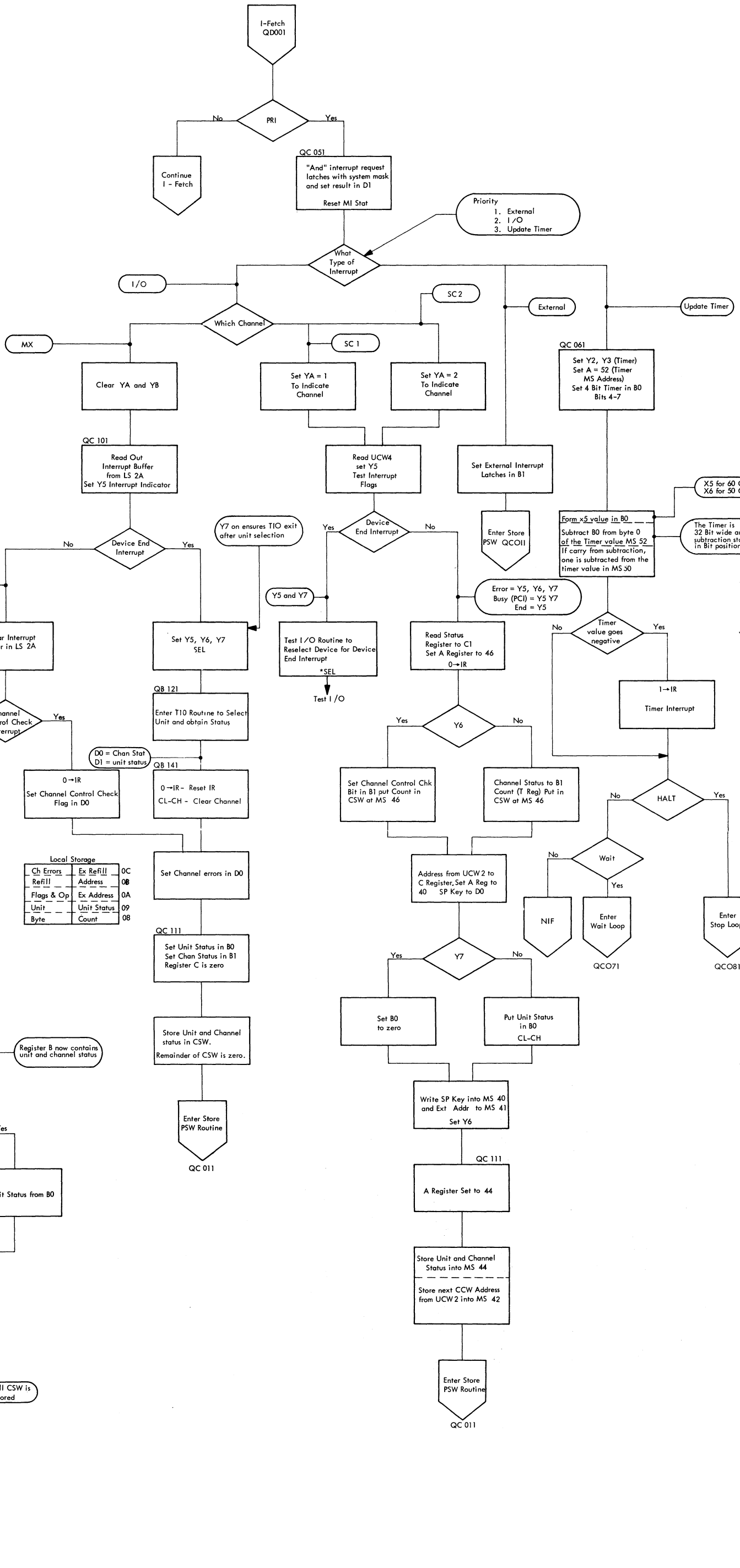


FIGURE 671. I/O INTERRUPTS AND UPDATE TIMER MICROPROGRAM

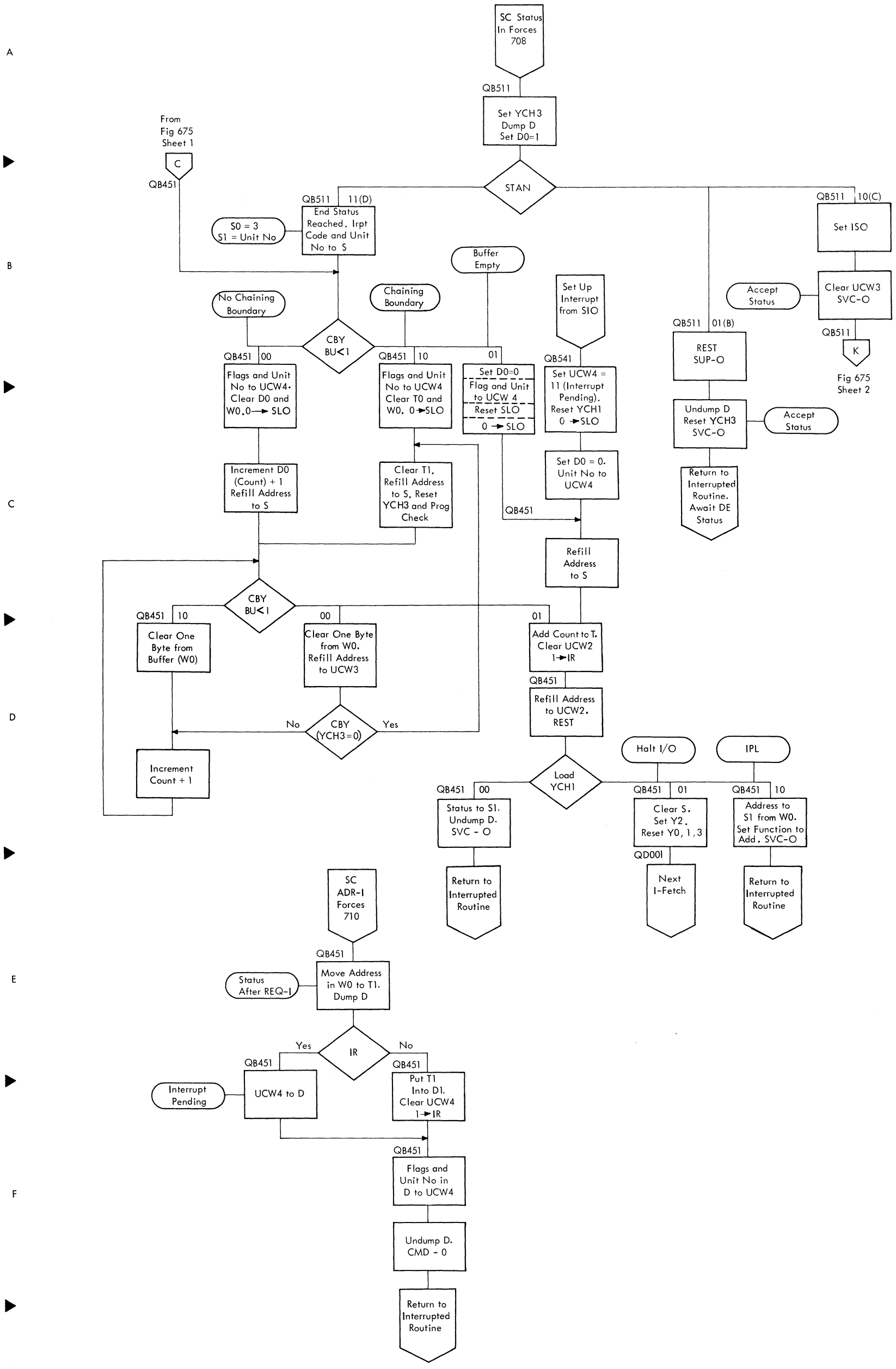


FIGURE 674. SELECTOR CHANNEL STATUS

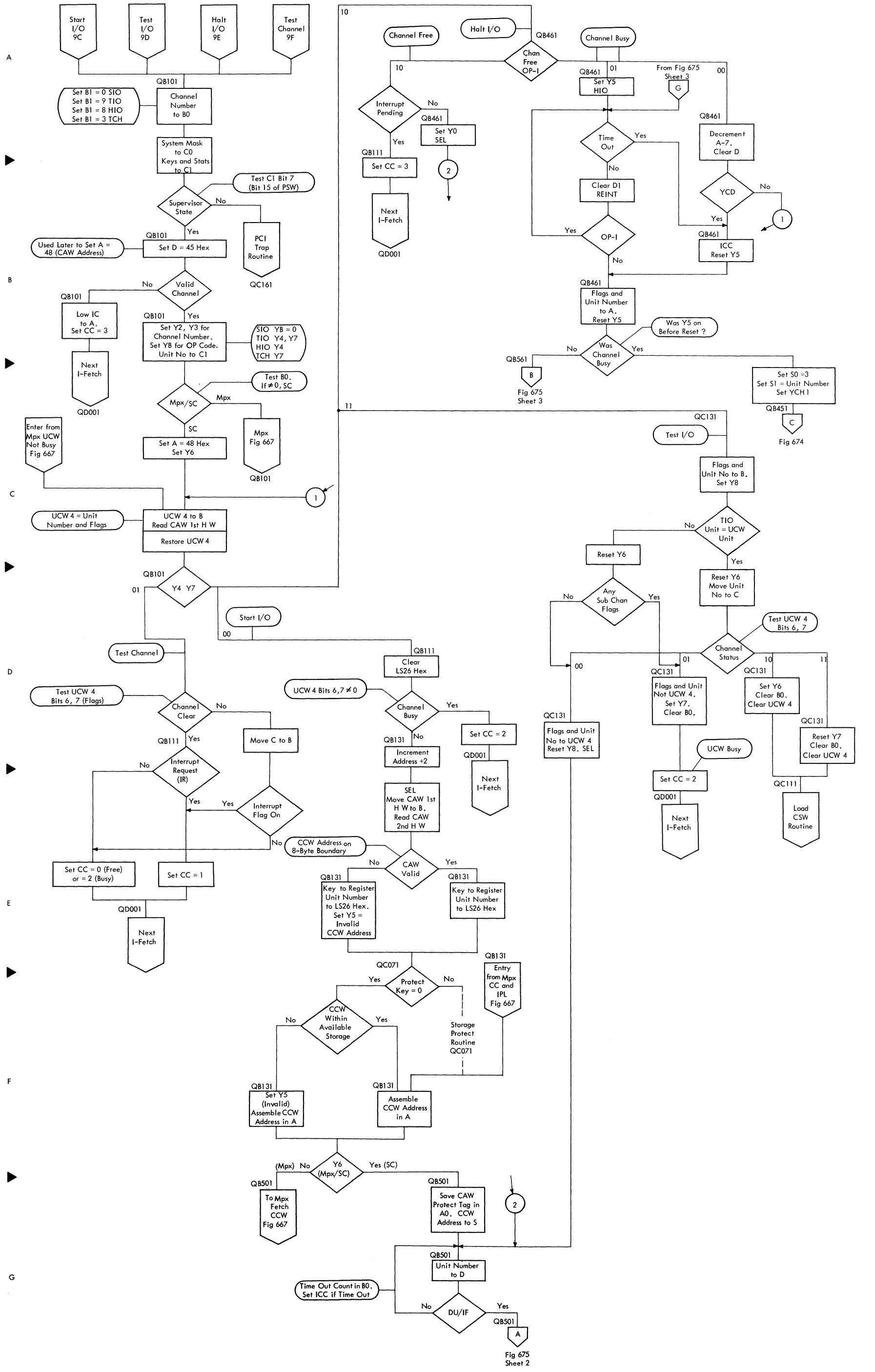


FIGURE 675. SELECTOR CHANNEL I/O INSTRUCTIONS MICROPROGRAM (SHEET 1 OF 4)

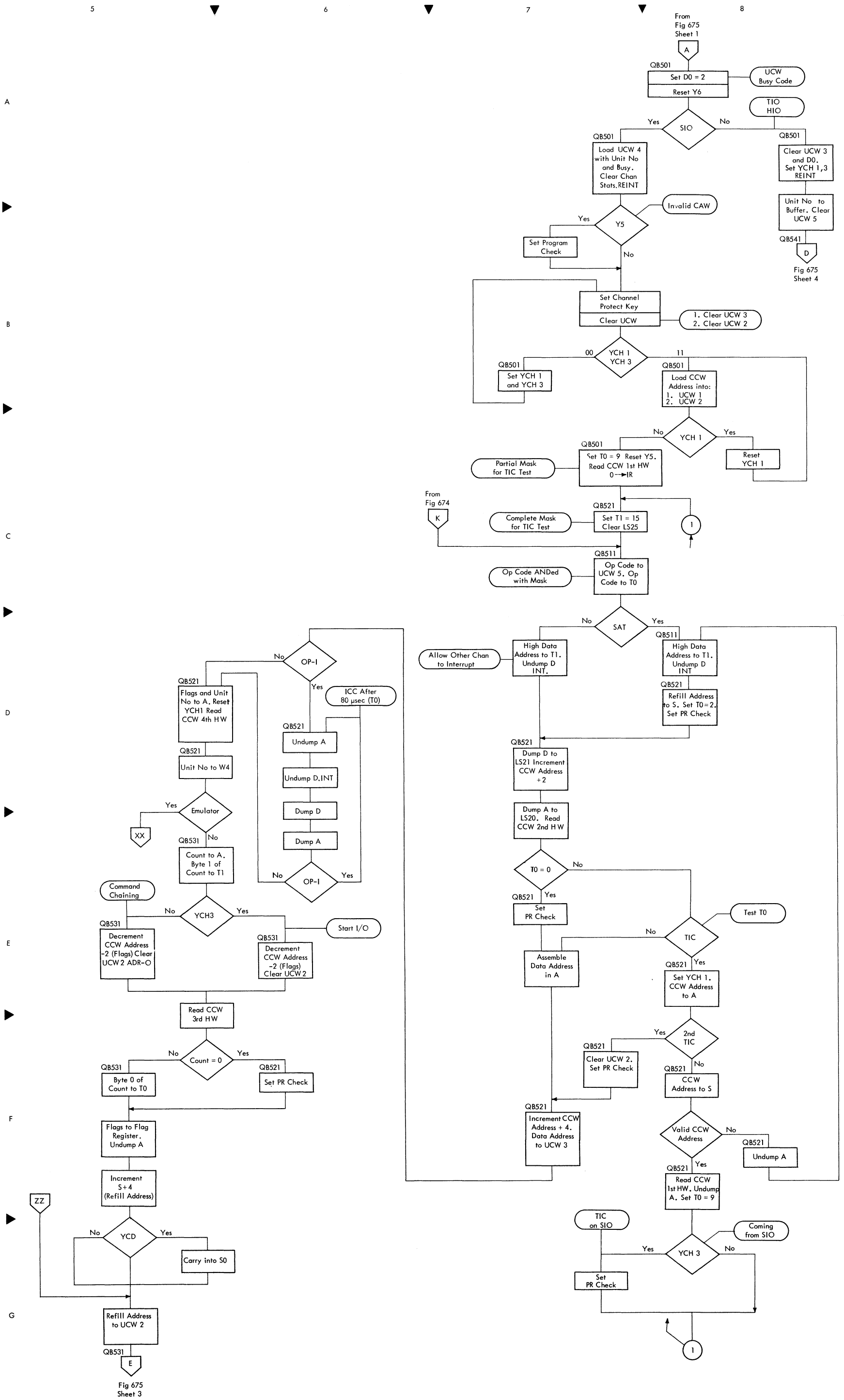


FIGURE 675. SELECTOR CHANNEL I/O INSTRUCTIONS MICROPROGRAM (SHEET 2 OF 4)

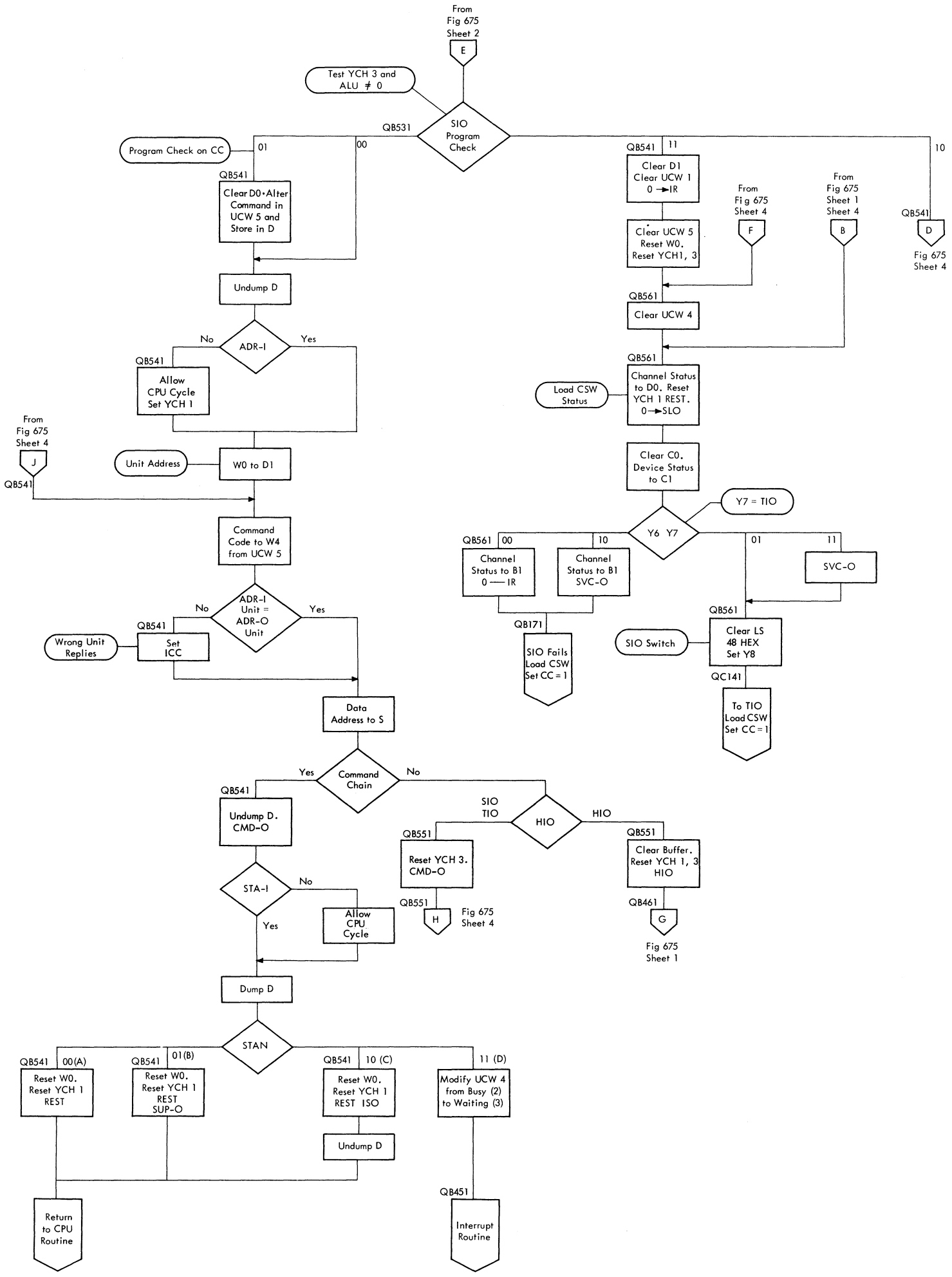


FIGURE 675. SELECTOR CHANNEL I/O INSTRUCTIONS MICROPROGRAM (SHEET 3 OF 4)

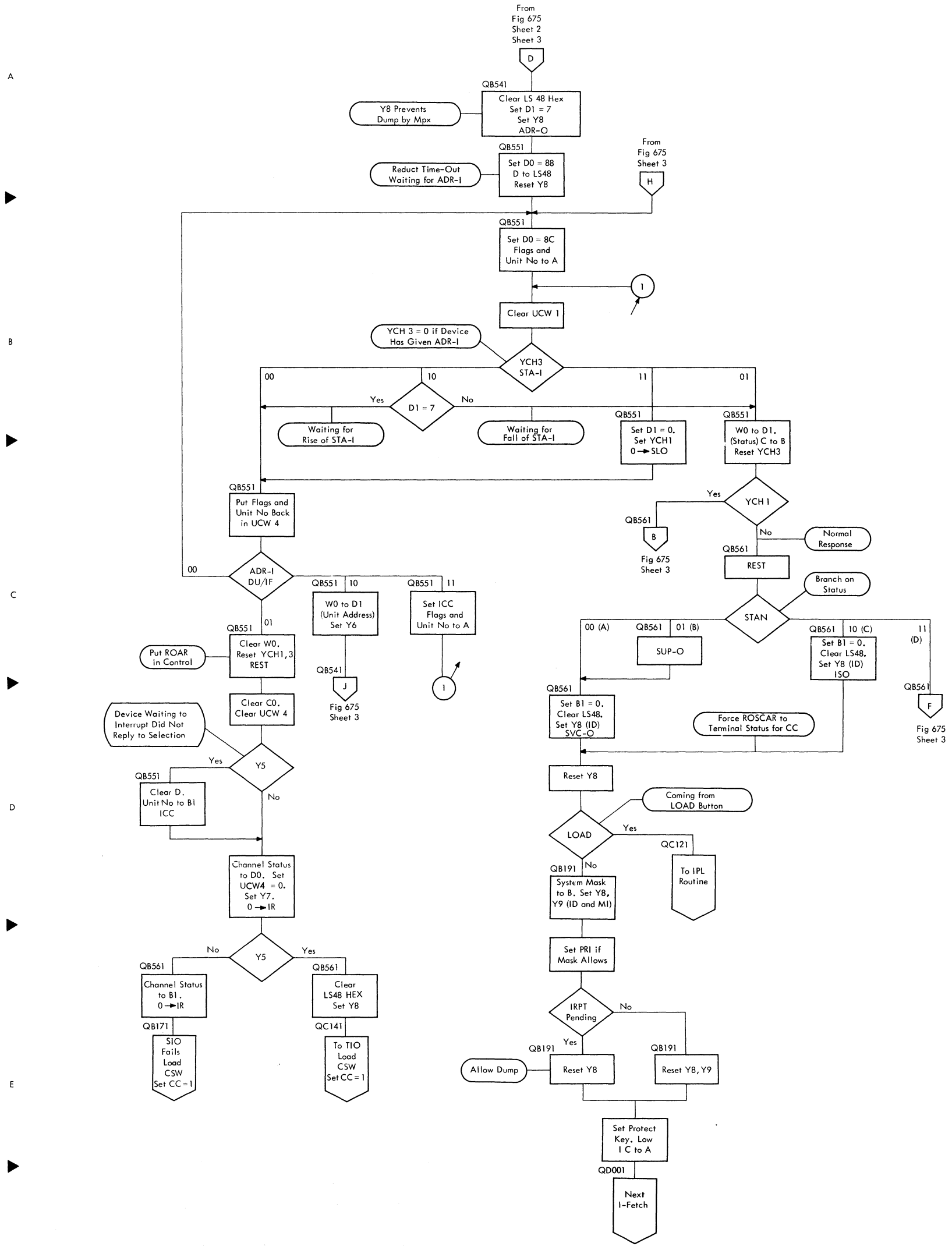
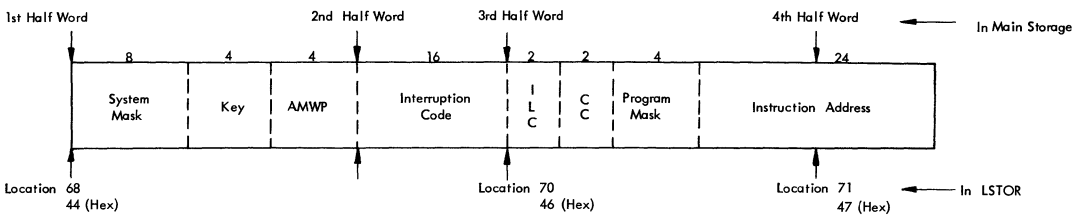


FIGURE 675. SELECTOR CHANNEL I/O INSTRUCTIONS MICROPROGRAM (SHEET 4 OF 4) (11/2/65)



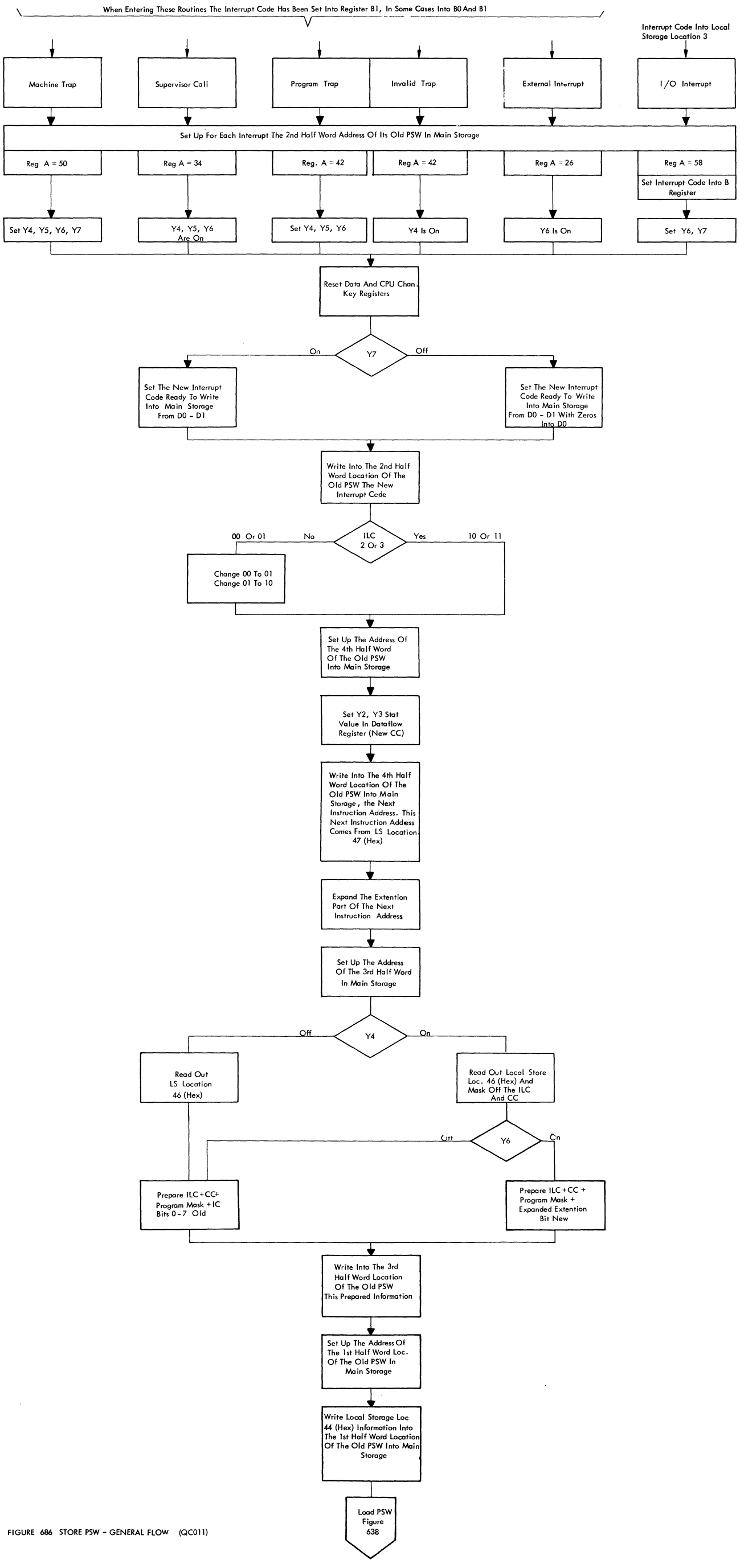


FIGURE 686 STORE PSW - GENERAL FLOW (QC011)

BASIC HINTS FOR THE CE

1. The philosophy behind MAPS is basically to guide the CE and to minimize wrong conclusions
2. Analyse the console display carefully. Do not start logic page analysis until you are satisfied that the information provided by the indicators has been fully utilized
3. Locate the source check. This is particularly important in the case of channel checks
4. Some checks are more explicit than others. Always establish the amount of hardware funneled into a check by means of the appropriate ECAD. The amount of hardware involved will influence the procedure adopted to pinpoint the fault, i.e. in certain cases you can go straight to the card (s) involved
5. Switch on your scope at the beginning of the card-changing activity
6. Where possible change cards in preference to scoping since this process of elimation is far less susceptible to misleading conclusions
7. Always look for some common relationship between fault symptoms

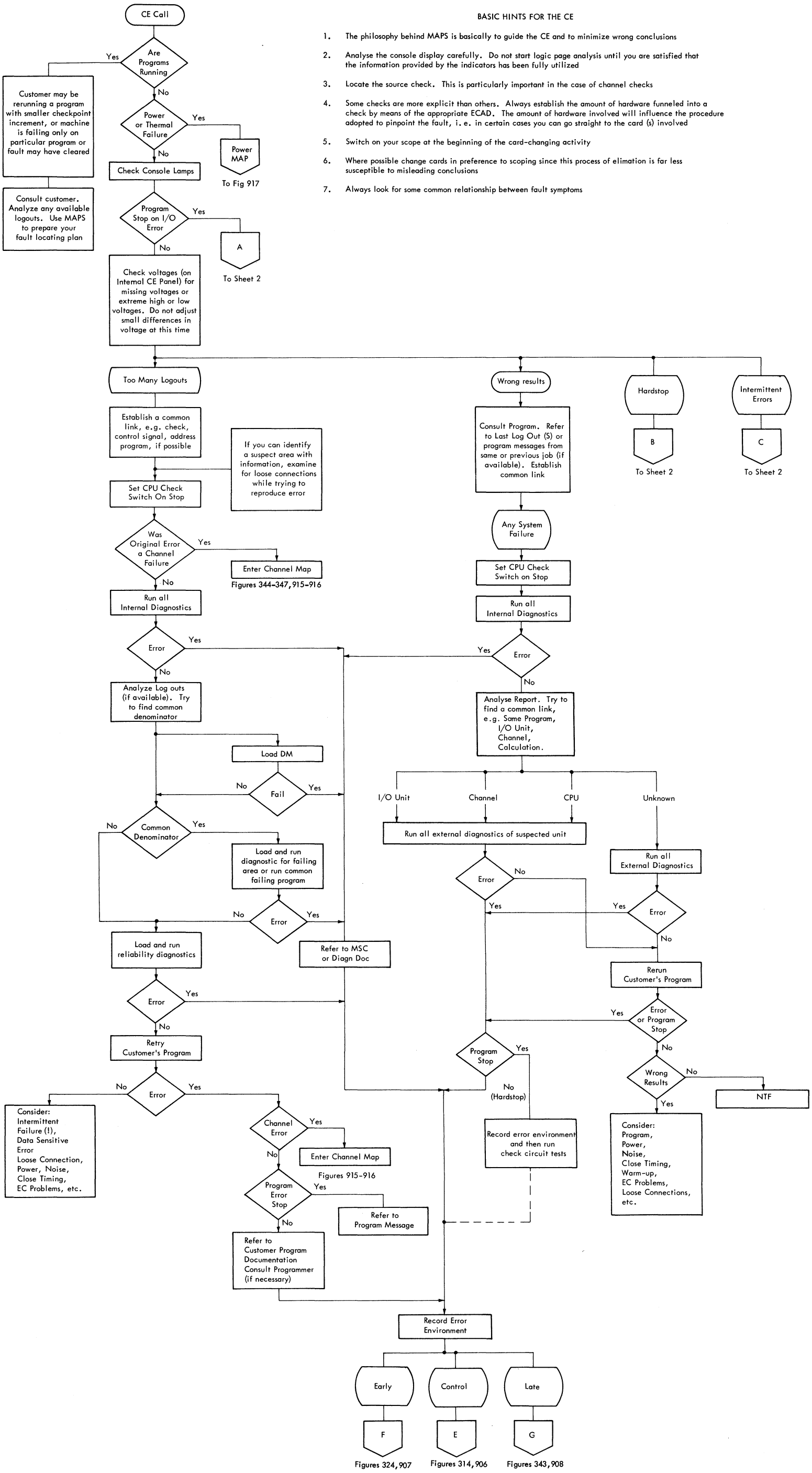


FIGURE 901. INTERPRET ERRORS (SHEET 1 OF 2)

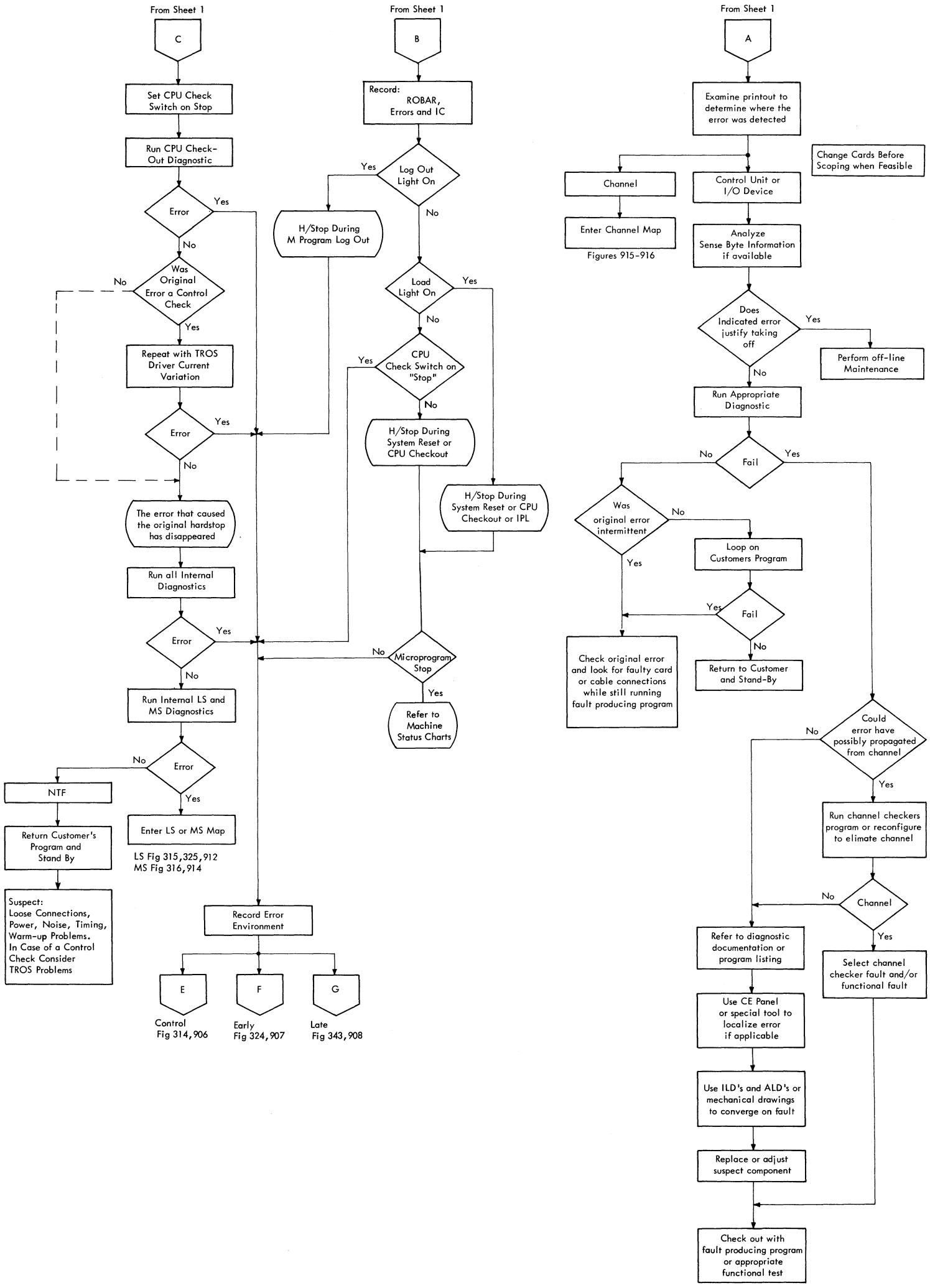


FIGURE 901. INTERPRET ERRORS (SHEET 2 OF 2)

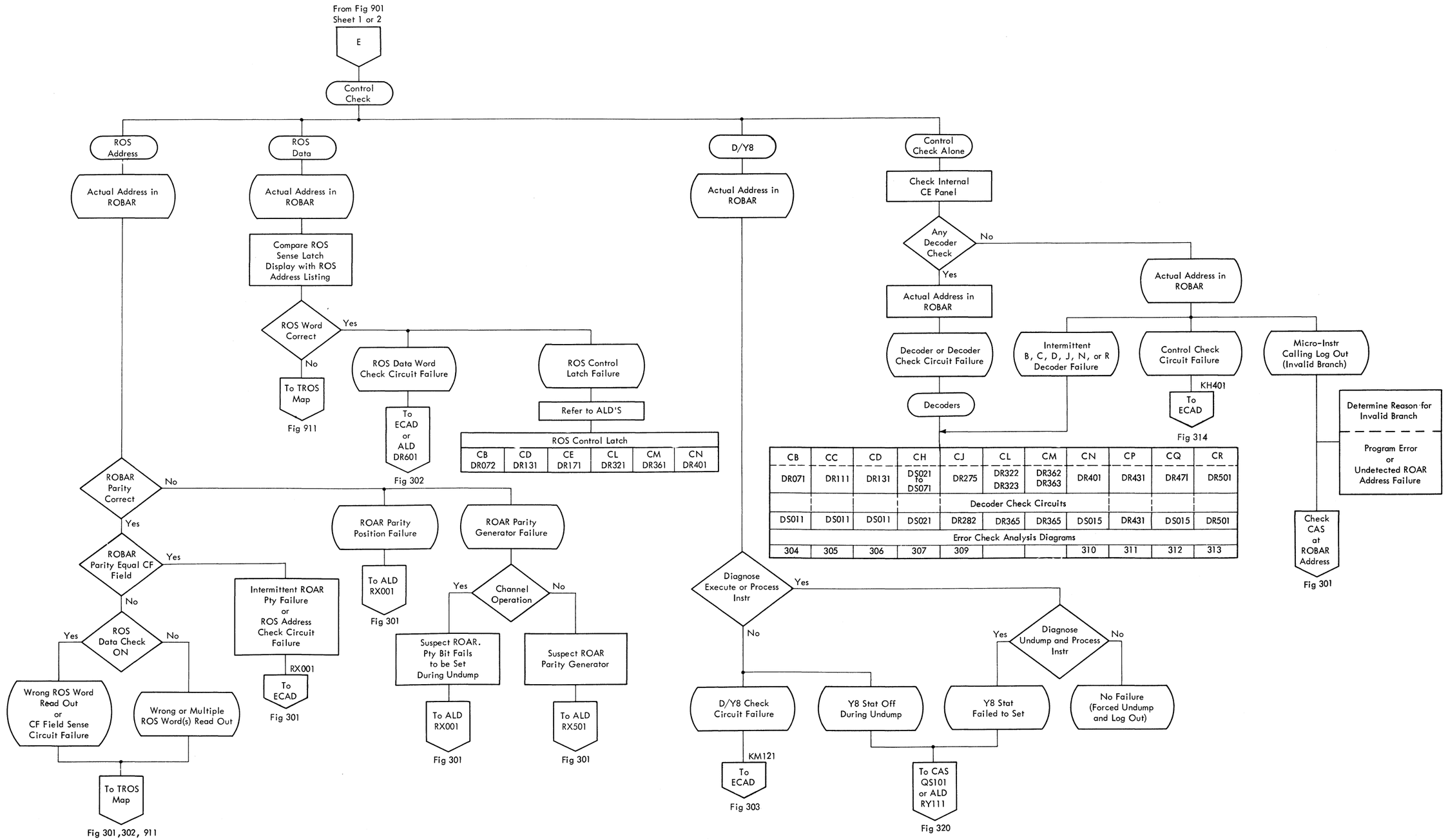


FIGURE 906. CONTROL CHECK

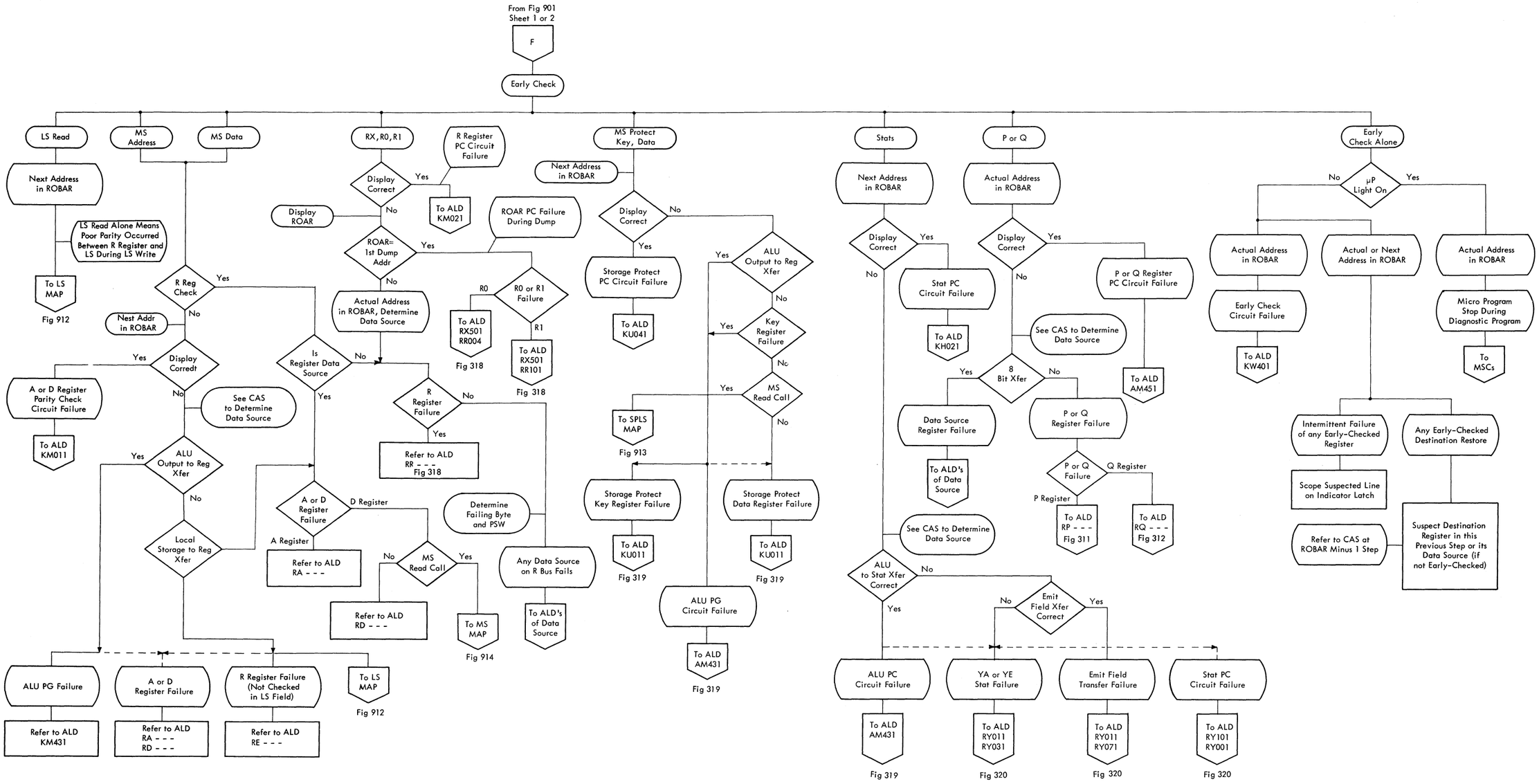


FIGURE 907. EARLY CHECK

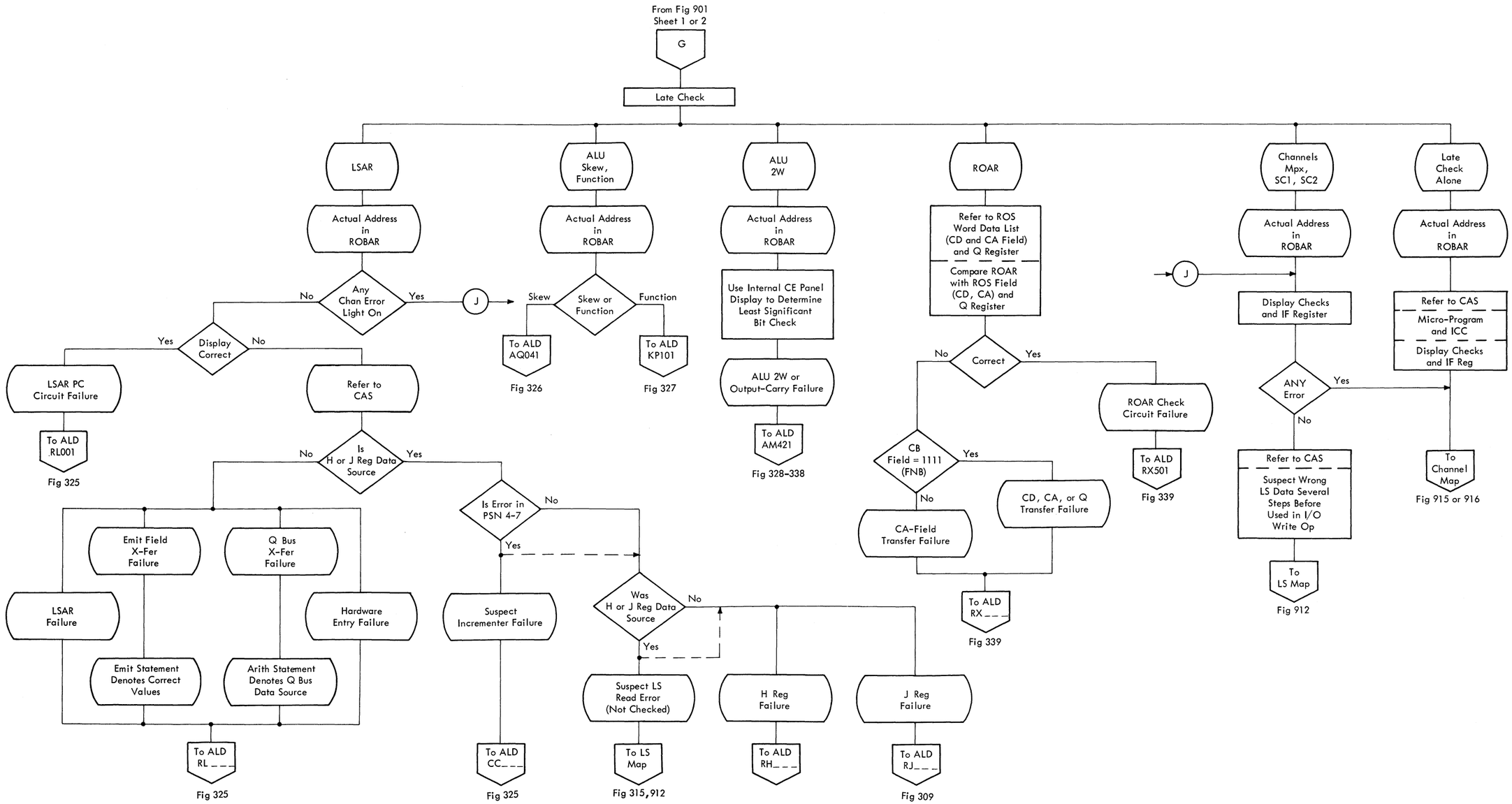


FIGURE 908. LATE CHECK

All 1's Test Word Address= 020
All 0's Test Work Address= 010

(a) ROS Error Suspected or Machine Stop with some ROS Check Lights On

(b) Note Check Lights, Note ROBAR Address and Control Word Contents

(c) Test all Check Lights to Prevent false deductions

(d) Check System Power Supplies +3, -3 +6, volts + 4% Check drive current — Do Not Adjust at this Point

Any Bits In Error — No: Complete Word No '1's out; Yes: (1)

(1a) Manually Select All '1's Test Word (00800)

(2) See Note 1

(2a) Some or all '1's out: Revert to Error Word adr → Invert adr bits 0, 1, 2, 3, 4, 5 → (3)

(3a) Some '1's out: Invert adr bits 3, 4, 5 → (4)

(4a) Some '1's out: Invert adr bits 0, 2, 3, 4, 5 → (5)

(5a) Some '1's out: Revert to Error Word Add → Invert Adr Bit 7 → (6)

(6a) Some '1's out: Invert adr bit 7 → Short Error Word Diode on Diode Card → (7)

(7a) Some '1's out: Change Diode

No '1's out (7b): Check Word Tape for Continuity on Diode Card Pins

No '1's out (6b): Change Driver Card

No '1's out (5b): Change Driver Decoder Card

No '1's out (4b): Change Driver Decoder → Change Driver Supplies Card

No '1's out (3b): Revert to Error Word Add → Invert Adr Bit 6 → (5)

(8a) Some '1's out: Invert Adr Bits 6, 7 → (9)

(9a) Some '1's out: Change Gate Card

No '1's out (9b): Change Gate Decoder Card

No '1's out (8b): Invert Adr Bit 7 → (10)

(10a) Some '1's out: Change Gate Decoder Card

No '1's out (10b): Change Gate Strobe Card → No '1's out → Change Gate Strobe Select Card H3 → No '1's out → (2)

No '1's out (2b): Invert Adr Bits 0, 1, 2 → (11)

(11a) Some '1's out: Invert Adr Bits 0, 1, 2, 3, 4, 5 → (12)

(12a) Some '1's out: Change Driver Card

No '1's out (12b): Change Driver Decoder Card → (13)

(13a) Some '1's out: Change Driver Decoder Card → Driver Supplies Card

No '1's out (13b): Revert to all '1' Test Word → Invert Adr Bit 6 → (14)

(14a) Some '1's out: Invert Adr Bits 6, 7 → (15)

(15a) Some '1's out: Change Gate Card

No '1's out (15b): Change Gate Decoder Card

No '1's out (14b): Invert Adr Bit 7

(16a) Some '1's out: Change Gate Decoder Card

No '1's out (16b): Interchange Gate Strobe Cards → (17a) If available → 18

(18a) Some '1's out: Change Gate Strobe Card

No '1's out (18b)

(17b) If not available: Change Gate Strobe Card

(2) → Check Level of +PWR Good 02A1 - G6D12 → Change Timing Card A → Change Timing Card B → Change Timing Card C → Change Timing Card D

LAYOUT 6K TROS (CARD SIDE)

Board 02A1

	A	B	C	D	E	F	G	H	J	K	L	M	N
1													
2				Gate Decoders 16				Gate Strobe Select					Gate Address
3	Gate 'C'	Gate 'B'	Gate 'D'										
4	Gate Decoder Outputs			Drivers 16	Drivers 16	Driver Decoders 16	Tmng. Card 'D'						Driver Address + Timing
5													
6	Drivers Outputs			Drivers 16	Drivers 16		Tmng. Card 'A'		Tmng. Card 'B'		Tmng. Card 'C'		
7													
8													

Board 02A2

	A	B	C	D	E	F	G	H	J	K	L	M	N
1													
2	0			0 1 2 3	4 5 6 7	8 9 10 11	12 13 14 15	16 17 18 19	0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15			0 19
3	19				SENSE					SENSE			
4	20			20 21 22 23	24 25 26 27	28 29 30 31	32 33 34 35	36 37 38 39	16 17 18 19 20 21 22 23	24 25 26 27 28 29 30 31	32 33 34 35 36 37 38 39		20 39
5	39				AMPLIFIERS					LATCHES			
6	60			40 41	44 45	48 49	52 53	56 57	40 41 42 43 44 45 46 47	48 49 50 51	½ Unused 56 57 58 59		40 59
7	59												
8													

SENSE AMP INPUTS — LATCH OUTPUTS

The shaded positions are used for cable connections.

Note 1: Numbers in parentheses refer to Map Chart Reference Notes, on Sheet 3.

FIGURE 911. READ ONLY STORAGE (SHEET 1 OF 3)

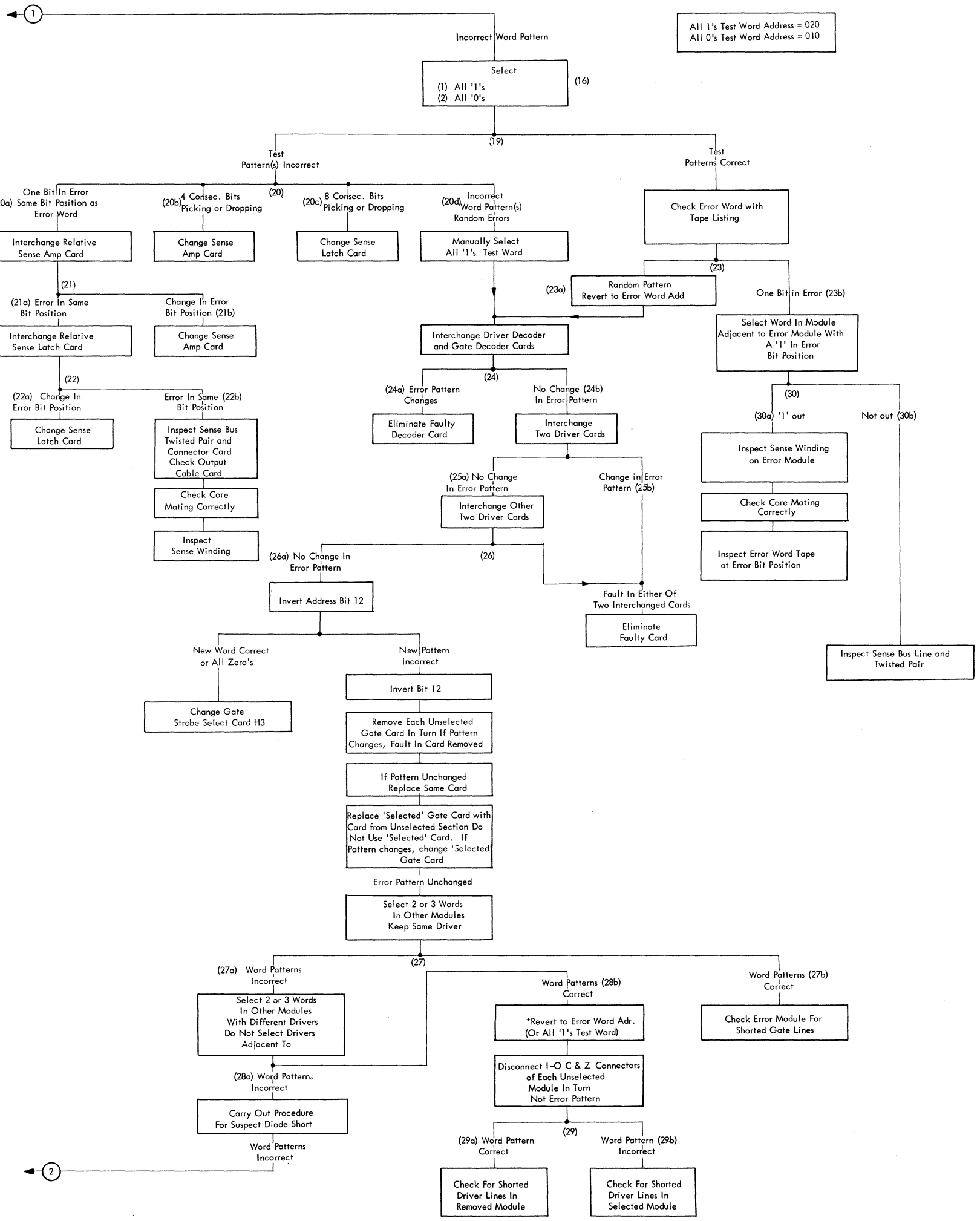
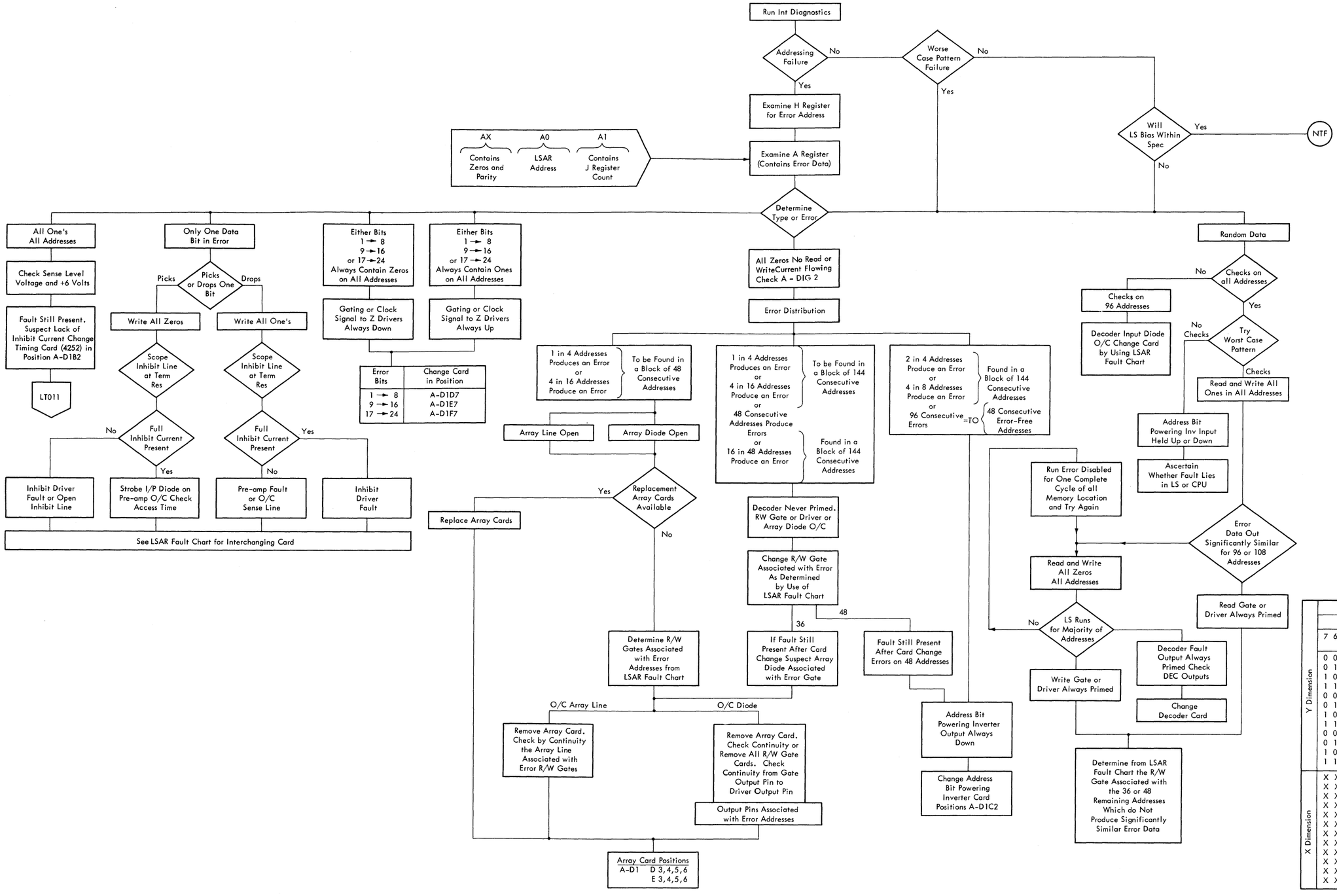


FIGURE 911. READ ONLY STORAGE (SHEET 2 OF 3)

MAP Chart Reference	Notes
a) b) c)	On entering the ROS MAP, the machine status, checks and supplies are noted before branching at 1. The presence of ROS drive current indicated by the CPU meter shows that drive current is available but does not necessarily mean that drive current is passing through the drive lines.
1 a)	The first branch is taken on the test for no 1's or incorrect word pattern. If no 1's, an attempt is made in the following tests to obtain some output, using the test word for all 1's if necessary, to prove the supplies, timing and sense circuits etc.
2	If some or all 1's out are obtained at branch 2 test, selecting the error address and inverting bits 0 through 5 will retain the gate of the error word but will select a different driver and driver decoder bits.
3 a) 4 5 5 7	Some 1's out at test 3 indicates a probable driver or driver decoder fault and tests 4, 5, and 6 are concerned with isolating such failures. Another possibility is an open circuit diode which is isolated by tests 6 and 7. Inverting bit 7 of the error address will retain the error word driver but will select a different gate in the same module.
4 b)	No 1's out at test 4 could indicate a fault in the tens driver decoder (address bits 6, 7, and 8). As the driver supply runs parallel to the decoded tens lines, it could also be a supply failure. The possible decoder error should be eliminated first.
3 b)	No 1's out at test 3 suggests a gate, a gate decoder, or a gate strobe failure. The various possibilities are isolated by tests 8, 9, and 10. Changing address bits 6 and 7 retains the original driver but changes the gates within the same module.
2 b) 11 through 18	If no 1's out are obtained at test 2, a similar series of tests to those listed above are conducted in tests 11 through 18, using the all 1's test word as the error word.
16 b) 17 b)	The interchange of gate strobe cards at 16 b) is only possible if both frames of ROS are used. If only one frame is used, a card change is made at 17 b).
1 b)	If incorrect word patters are obtained at test 1, the two diagnostic test words should be selected and the resulting outputs observed for a repeatable error pattern. Due to the packaging of the logic circuitry, this may give a clue to the faulty area as at branches 20 a), b), and c).
21 22	One particular bit in error is isolated by the interchange of cards and tests 21 and 22.
24 25 26	Random error patterns from the error word or all 1's test word are, as far as possible, isolated by the removal or interchange of card at tests 24, 25, and 26.
27 28	If the above tests do not affect the random error pattern, further tests are made to isolate the failing module. Drivers adjacent to the error drive should not be used in test 28, as a drive short would most likely be to an adjacent line.
28 b) 29	The error module is isolated by removal of the I/O connectors in branch 28 b) and the module checked for shorts at test 29.
28 a)	If throughout the tests branch 28 a) is reached without a correct pattern being obtained, a short-circuit diode should be suspected.
23 b) 30	Correct word patterns from the test words but one bit in error from the error word as in branch 23 suggest a sense circuit failure. The failing area can be isolated by test 30 and branches 30 a) and 30 b). Selecting a word in an adjacent module (i.e., on the same frame) with a 1 in the error position indicates a common fault (e.g., bus line) if there is no 1 out.

FIGURE 911. READ ONLY STORAGE (SHEET 3 OF 3)

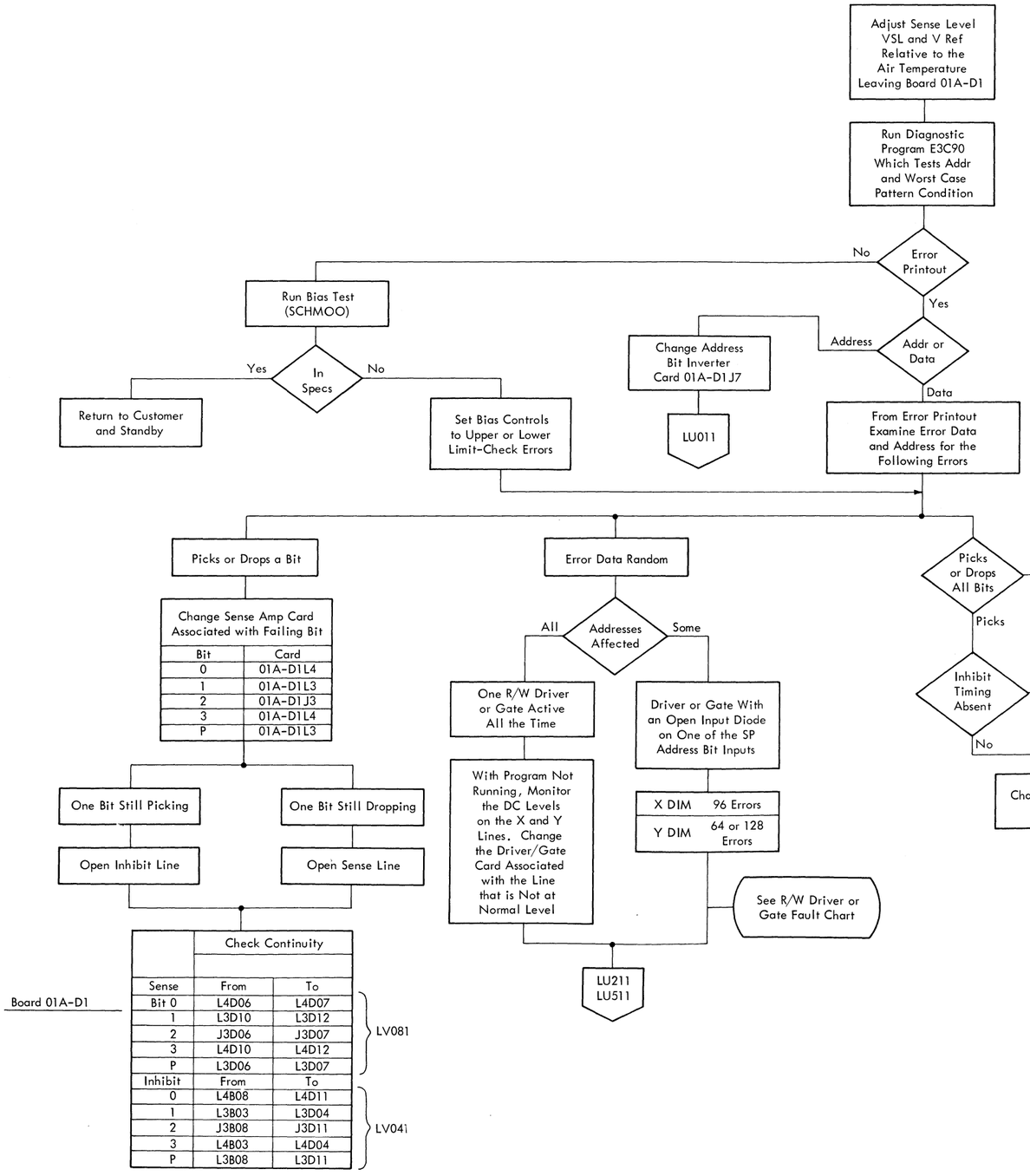


Bit Fails	Pre-amp Z Driver		Term Res or Output Pin
1	A-D1	B6	B03
2		C7	B08
3		B5	B08
4		C4	B08
5		C7	B03
6		C6	B08
7		C5	B08
8		C4	B03
9		B6	B08
10		C6	B03
11		B5	B03
12		C5	B03
13		F6	B03
14		G6	B08
15		F5	B03
16		F4	B08
17		G6	B03
18		F5	B08
19		G5	B03
20		F4	B03
21		F6	B08
22		G5	B08

LSAR Fault Chart

	LSAR Error Addresses								DEC and R/W Gate Faults	
	36 Error Addr							48 Error Addr		
	7	6	5	4	3	2	1	0	Replace Card in Position	Replace Card in Position
Y Dimension	0	0	X	X	0	0	X	X	A-D1C3	A-D1F3
	0	1	X	X	0	0	X	X		
	1	0	X	X	0	0	X	X		
	1	1	X	X	0	0	X	X		
	0	0	X	X	1	0	X	X		
	0	1	X	X	1	0	X	X		
	1	0	X	X	1	0	X	X		
	1	1	X	X	1	0	X	X		
	0	0	X	X	1	1	X	X		
	0	1	X	X	1	1	X	X		
	1	0	X	X	1	1	X	X		
	1	1	X	X	1	1	X	X		
X Dimension	X	X	0	0	X	X	0	0	A-D1D2	A-D1E2
	X	X	0	1	X	X	0	0		
	X	X	1	0	X	X	0	0		
	X	X	1	1	X	X	0	0		
	X	X	0	0	X	X	0	1		
	X	X	0	1	X	X	0	1		
	X	X	1	0	X	X	0	1		
	X	X	1	1	X	X	0	1		
	X	X	0	0	X	X	1	1		
	X	X	0	1	X	X	1	1		
	X	X	1	0	X	X	1	1		
	X	X	1	1	X	X	1	1		

FIGURE 912. LOCAL STORAGE

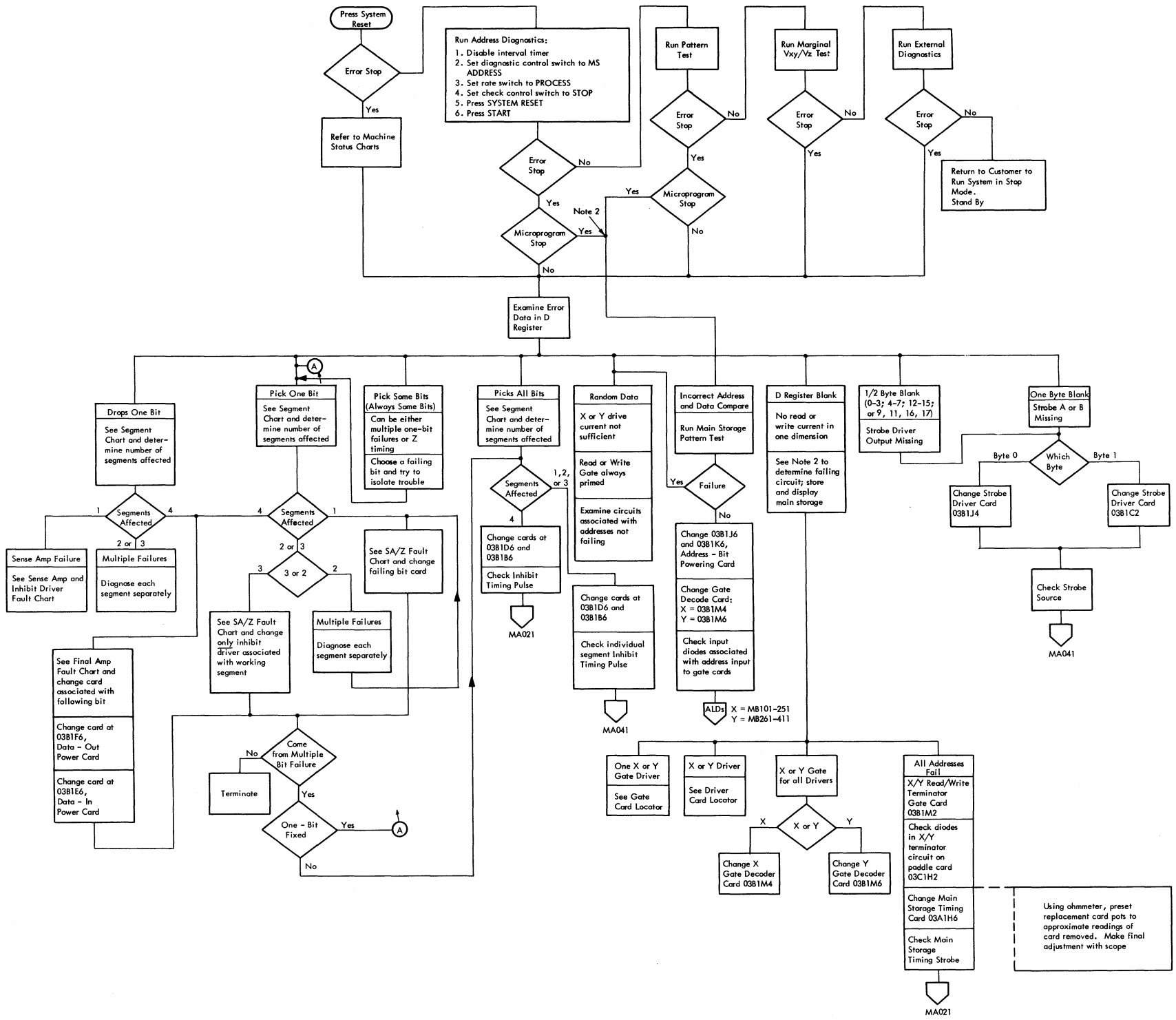


Read/Write Driver or Gate Fault Chart

KEY	DIM	Main Storage Address Bits Common to All Error Addresses																Card Location
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
CPU	X					X	X											01A-D1L5
				X	X													D1J5
	Y	X	X															D1L2
		X	X															D1L2
MPX	X												X	X				D1L5
											X	X						D1J5
	Y		X	X														D1L2
								X										D1J2

LU211 to LU511

FIGURE 913. STORAGE PROTECT MAP



Final Amplifier Fault Chart

Data Bits	Card
0-8	01A-B1J4
9-17	01A-B1C2

Segment Chart

Main Storage Address Bits			Segment for which the inhibit drivers are allowed to be activated
15	14	6	
0	0	0	A
0	0	1	B
0	1	0	C
0	1	1	D
			Array 1

Sense Amp and Inhibit Driver Fault Charts (Refer to ALD's MF010-MF060)

Data Bits	Array 1 - Segments			
	A	B	C	D
0, 1, 2, 3	B1L2	B1L4	B1K2	B1K4
4, 5, 6, 7	B1J2	B1H4	B1H2	B1G4
8, 9, 10, 11	B1G2	B1F4	B1F2	B1E4
12, 13, 14, 15	B1E2	B1D4	B1D2	B1C4
16, 17	B1B2	B1B4	A1M2	A1M4

Driver Card Locator

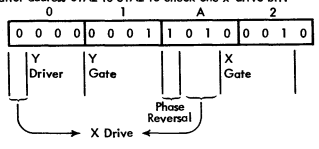
X and Y Drivers 0-3 = B1N4
X and Y Drivers 4-7 = B1N6

Gate Card Locator

X Gate				Y Gate			
SAB Bits				SAB Bits			
14	0	Read	Write	13	7	Read	Write
0	0	D1C6	D1C2	0	0	D1F2	D1G2
1	0	D1B6	D1B2	1	0	D1E2	D1D2
0	1	D1H6	D1H2	0	1	D1F6	D1G6
1	1	D1J6	D1J2	1	1	D1E6	D1D6

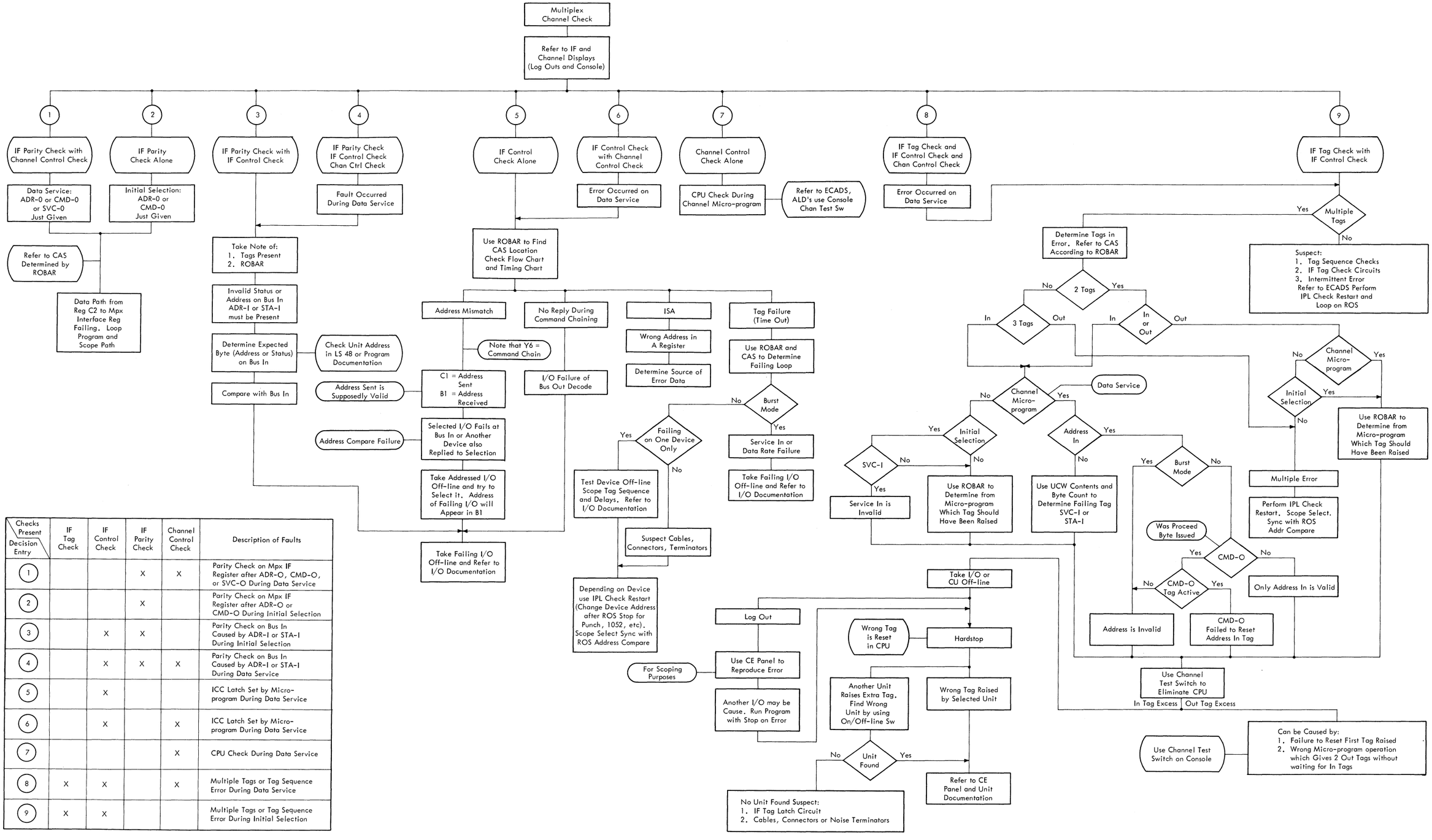
Notes:

1. Refer to maintenance manual section on internal storage diagnostics.
2. Alter A register bits to determine failing driver or gate. For example, alter address 01A2 to 81A2 to check one x-drive bit.



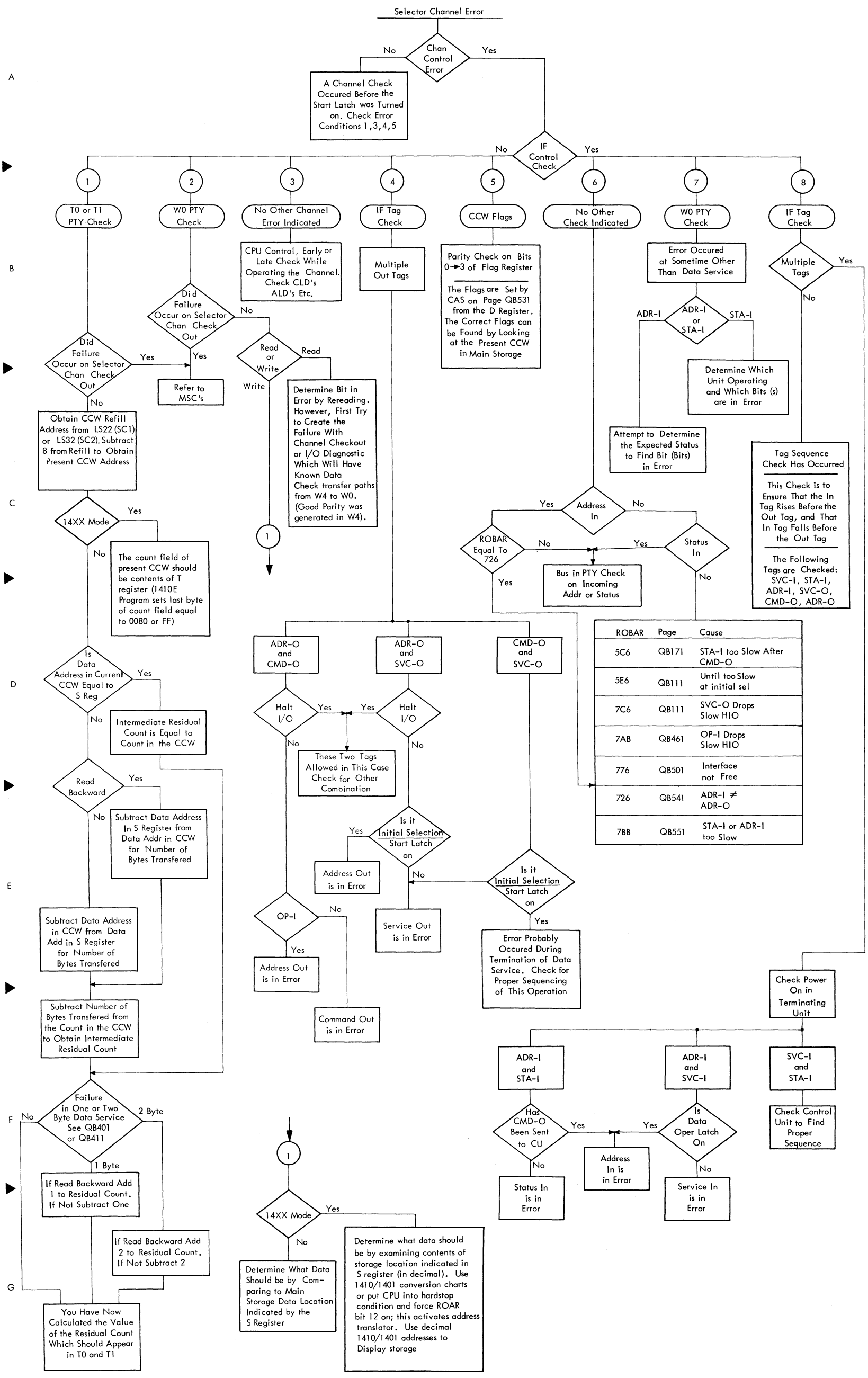
3. All card references are to frame 03.

FIGURE 914. MAIN STORAGE (64K)



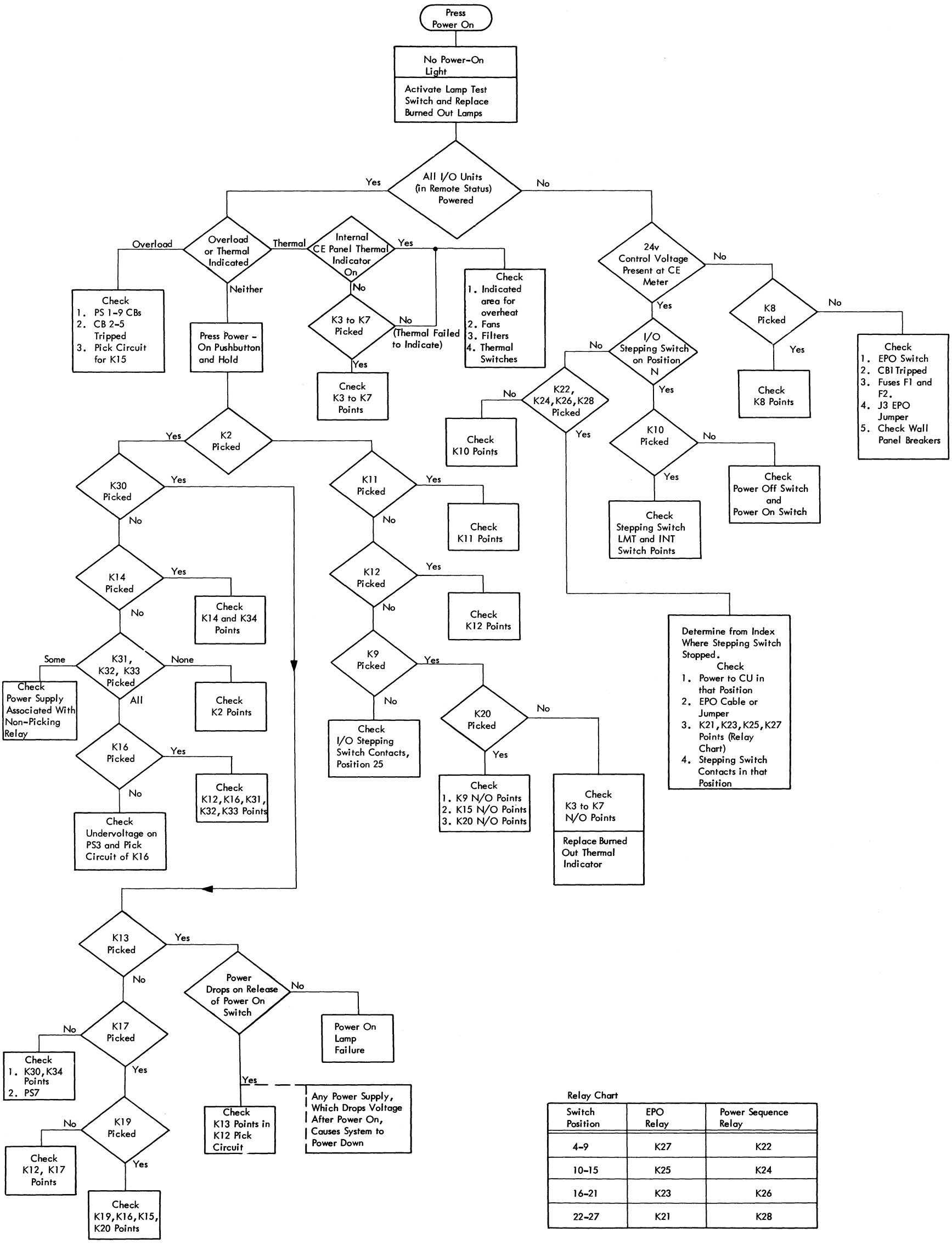
Checks Present / Decision Entry	IF Tag Check	IF Control Check	IF Parity Check	Channel Control Check	Description of Faults
1			X	X	Parity Check on Mpx IF Register after ADR-O, CMD-O, or SVC-O During Data Service
2			X		Parity Check on Mpx IF Register after ADR-O or CMD-O During Initial Selection
3		X	X		Parity Check on Bus In Caused by ADR-1 or STA-1 During Initial Selection
4		X	X	X	Parity Check on Bus In Caused by ADR-1 or STA-1 During Data Service
5		X			ICC Latch Set by Micro-program During Data Service
6		X		X	ICC Latch Set by Micro-program During Data Service
7				X	CPU Check During Data Service
8	X	X		X	Multiple Tags or Tag Sequence Error During Data Service
9	X	X			Multiple Tags or Tag Sequence Error During Initial Selection

FIGURE 915. MULTIPLEX CHANNEL MAP



ROBAR	Page	Cause
5C6	QB171	STA-I too Slow After CMD-O
5E6	QB111	Until too Slow at initial sel
7C6	QB111	SVC-O Drops Slow HIO
7AB	QB461	OP-I Drops Slow HIO
776	QB501	Interface not Free
726	QB541	ADR-I ≠ ADR-O
7BB	QB551	STA-I or ADR-I too Slow

FIGURE 916. SELECTOR CHANNEL



Relay Chart

Switch Position	EPO Relay	Power Sequence Relay
4-9	K27	K22
10-15	K25	K24
16-21	K23	K26
22-27	K21	K28

FIGURE 917 MID - PAC POWER SUPPLY

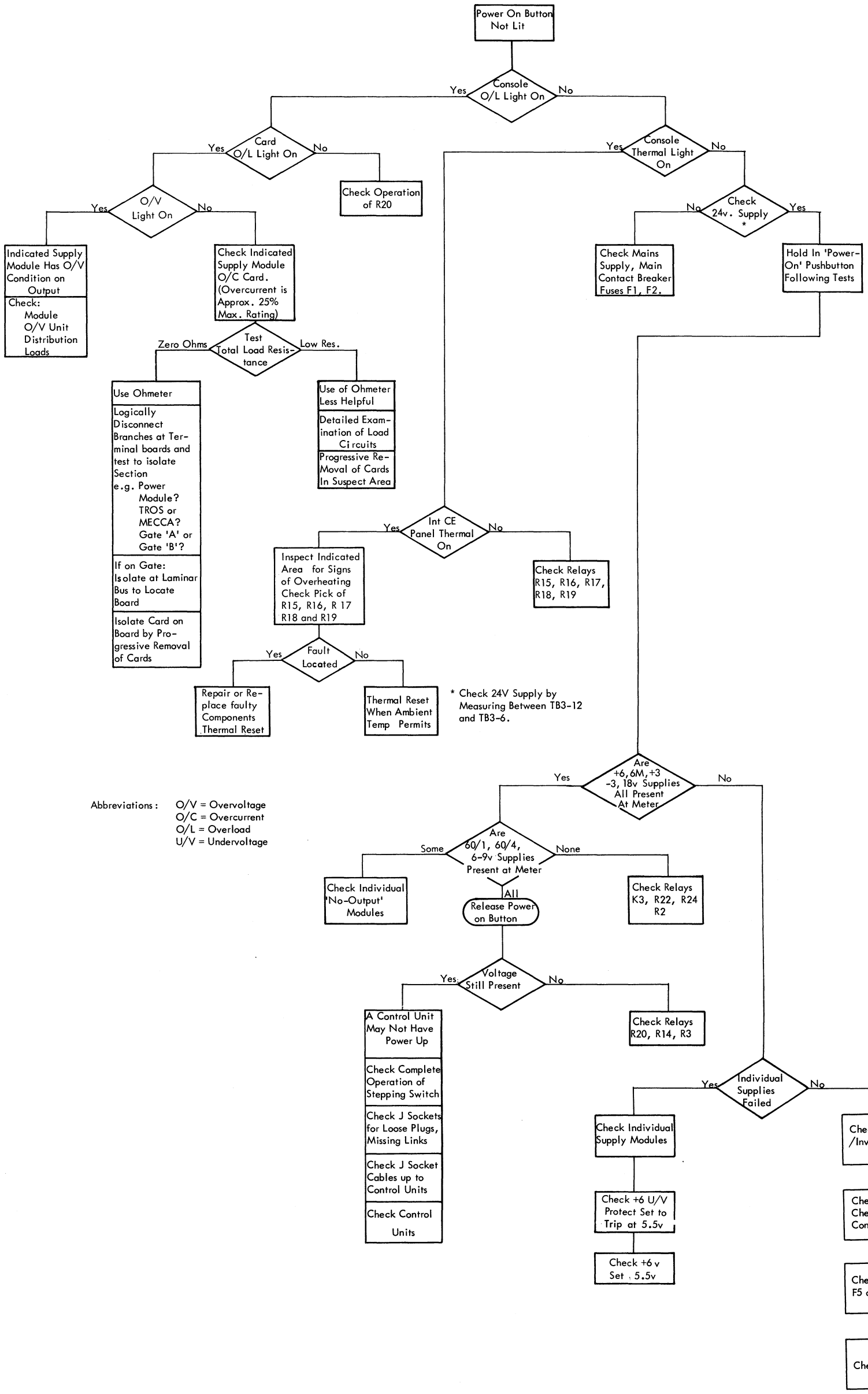
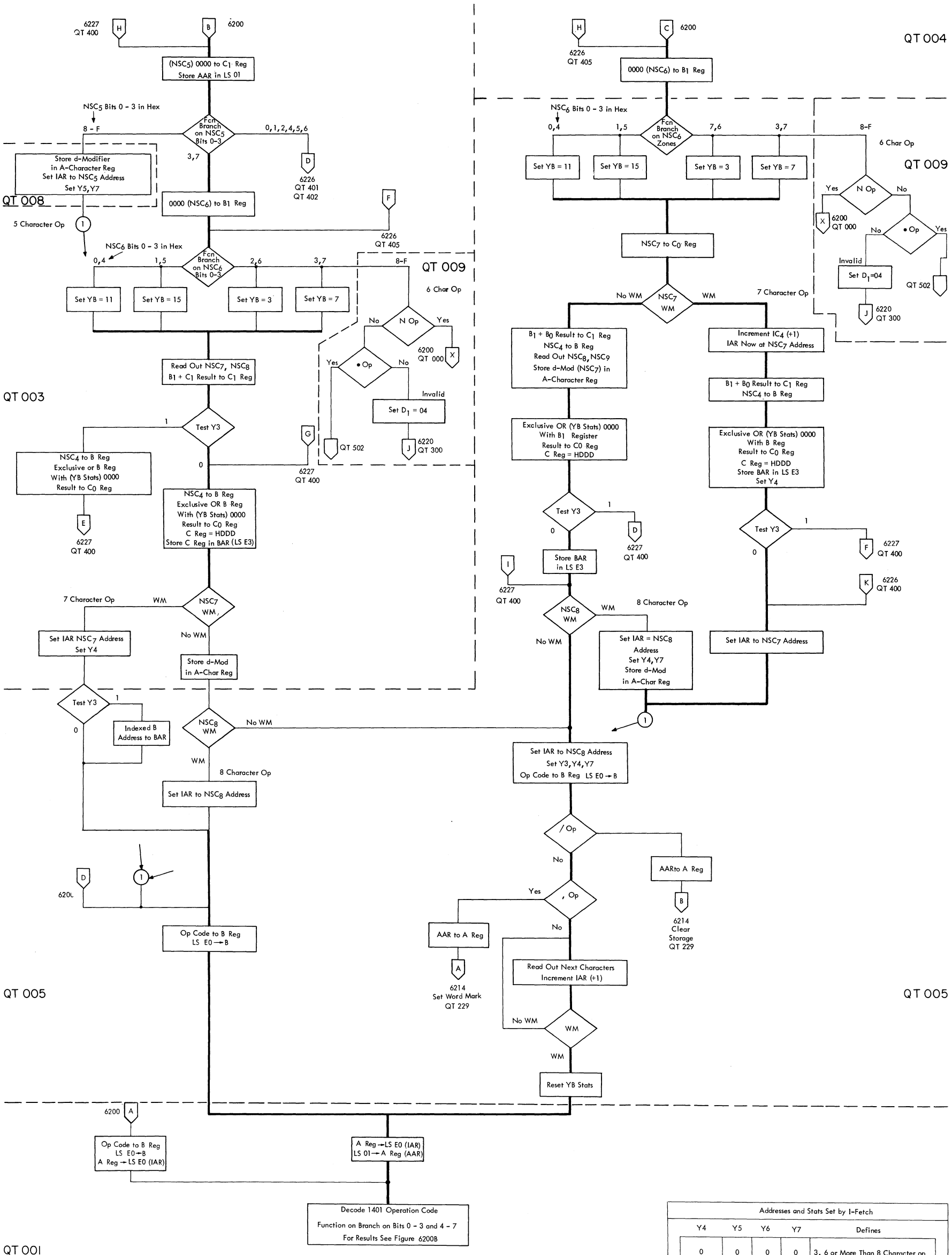


FIGURE 918. 2.5 KC HF POWER SUPPLY



Addresses and Stats Set by I-Fetch				
Y4	Y5	Y6	Y7	Defines
0	0	0	0	3, 6 or More Than 8 Character op
0	0	1	0	1 Character op
0	0	1	1	2 Character op
0	1	0	0	4 Character op
0	1	0	1	5 Character op
1	0	0	0	7 Character op
1 *	0	0	1	8 Character op

Local Storage Contains		Format
E0	1401 IAR	HDDD
E1	1401 A-Character Register	A-Char - N/A
01	1401 AAR (Old AAR in E2)	HDDD
E3	1401 BAR	HDDD
EE	Sense Switches Status Indicators	Bits

YA Stats are set to 0000 at end of I-Fetch
*Y4 and Y7 set for /Op and , Op

FIGURE 6200A. 1401 INSTRUCTION FETCH

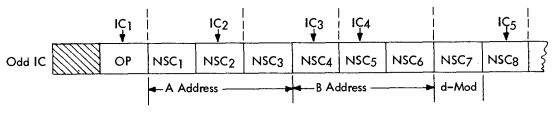
Op Code * Bits 0 – 3 / **	Bits 4 – 7 → XXXX0000	XXXX0001	XXXX0010	XXXX0011	XXXX0100	XXXX0101	XXXX0110	XXXX0111	XXXX1000	XXXX1001	XXXX1010	XXXX1011	XXXX1100	XXXX1101	XXXX1110	XXXX1111
1000XXXX	Invalid Op QT 307 or QT 308											• Halt QT 502	¤ Clear WM QT 221 6214	Invalid Op QT 307 or QT 308		
1001XXXX	Invalid Op QT 307 or QT 308															
1010XXXX	Invalid Op QT 307 or QT 308											Set WM QT 229 6214	% Divide QT 210 6206	Invalid Op QT 308		
1011XXXX	Invalid Op QT 307 or QT 308											# Modify Address 6213	@ Multiply QT 210 6208	Invalid Op QT 308		
1100XXXX	? Zero and Add QT 203 6209	A Add QT 117 6202	B Branch QT 303 6222	C Compare QT 117 6203	D Move Numeric QT 500	E Edit QT 229	F Carriage Control QT 225 6212	Invalid Op QT 307	H Store Bar QT 205 6204	Invalid Op QT 307 or QT 308						
1101XXXX	! Zero and Subtract QT 203 6209	Invalid Op QT 307	K Stacker Select QT 224 6212	L Load QT 203 6209	M Move QT 203 6209	N Nop QT 009 6201	Invalid Op QT 307	P Move to RM or GM QT 501	Q Store AAR QT 205 6204	Invalid Op QT 307 or QT 308						
1110XXXX	Invalid Op QT 308	/ Clear Storage QT 229 6214	S Subtract QT 117 6202	Invalid Op QT 308	U Tape Control QT 222 6210	V Branch WM or Zone QT 300 6219	W Branch Bit Equal QT 300 6219	Invalid Op QT 308	Y Move Zone QT 500	Z Move and Zero Suppress 6215	Invalid Op QT 307 or QT 308					
1111XXXX	Invalid Op QT 308	1 Read QT 223 6211	2 Print QT 223 6211	3 Read Print QT 223 6211	4 Punch QT 223 6211	5 Read Punch QT 223 6211	6 Print Punch QT 223 6211	7 Read Print Punch QT 223 6211	8 (Nop) QT 223 6211	9 (Nop) QT 223 6211	Invalid Op QT 307 or QT 308					

* Op code in EBCDIC–II representation
** Bits 0 – 3 = 0000 – 0111 are invalid and exit to QT 307

FIGURE 6200B. 1401 INSTRUCTION FETCH

Y	Means
0	Invalid NSC_1
2	Branch Op
3	Index

Local Storage Contains		Format
E0	1401 IAR	HDDD
E1	1401 A Character Register	A Char–N/A
E2	1401 AAR	HDDD
E3	1401 BAR	HDDD



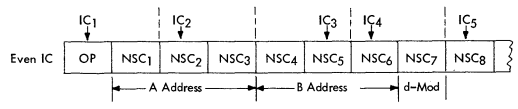


FIGURE 6200. 1401 INSTRUCTION FETCH

QT 009

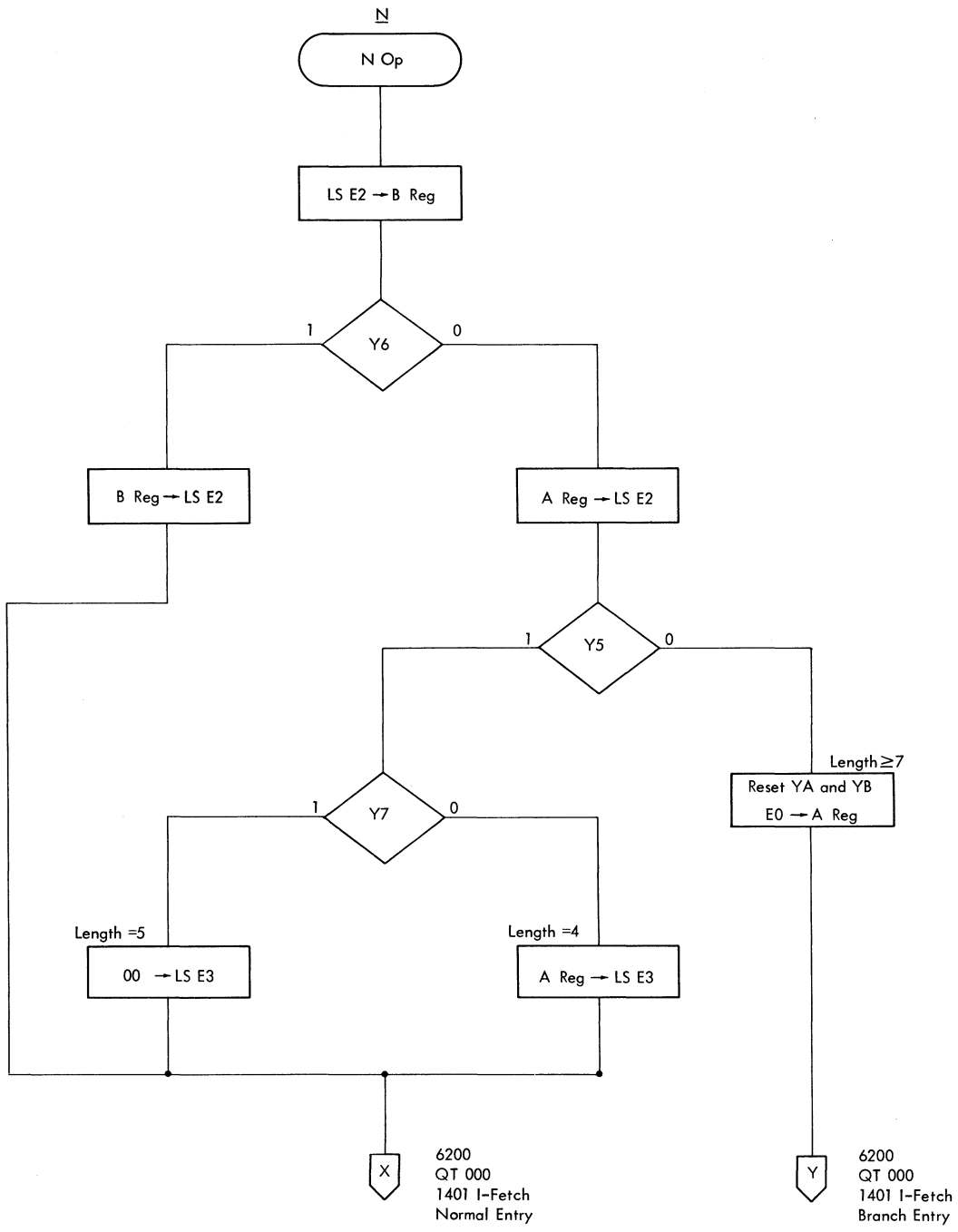


FIGURE 6201. 1401 N OPERATION

Stats	
Y	Means
4	7 Char Op
5	4 Char Op
6	1 Char Op

Address Test	
Y	Means
4	Address (A–B) = 1
5	B Address Odd
6	Both Addresses Odd
7	Address (A–B) Odd

Stats	
Y	Means
0	B Word Mark
2	Subtract
3	1401-Carry
6	A Word Mark

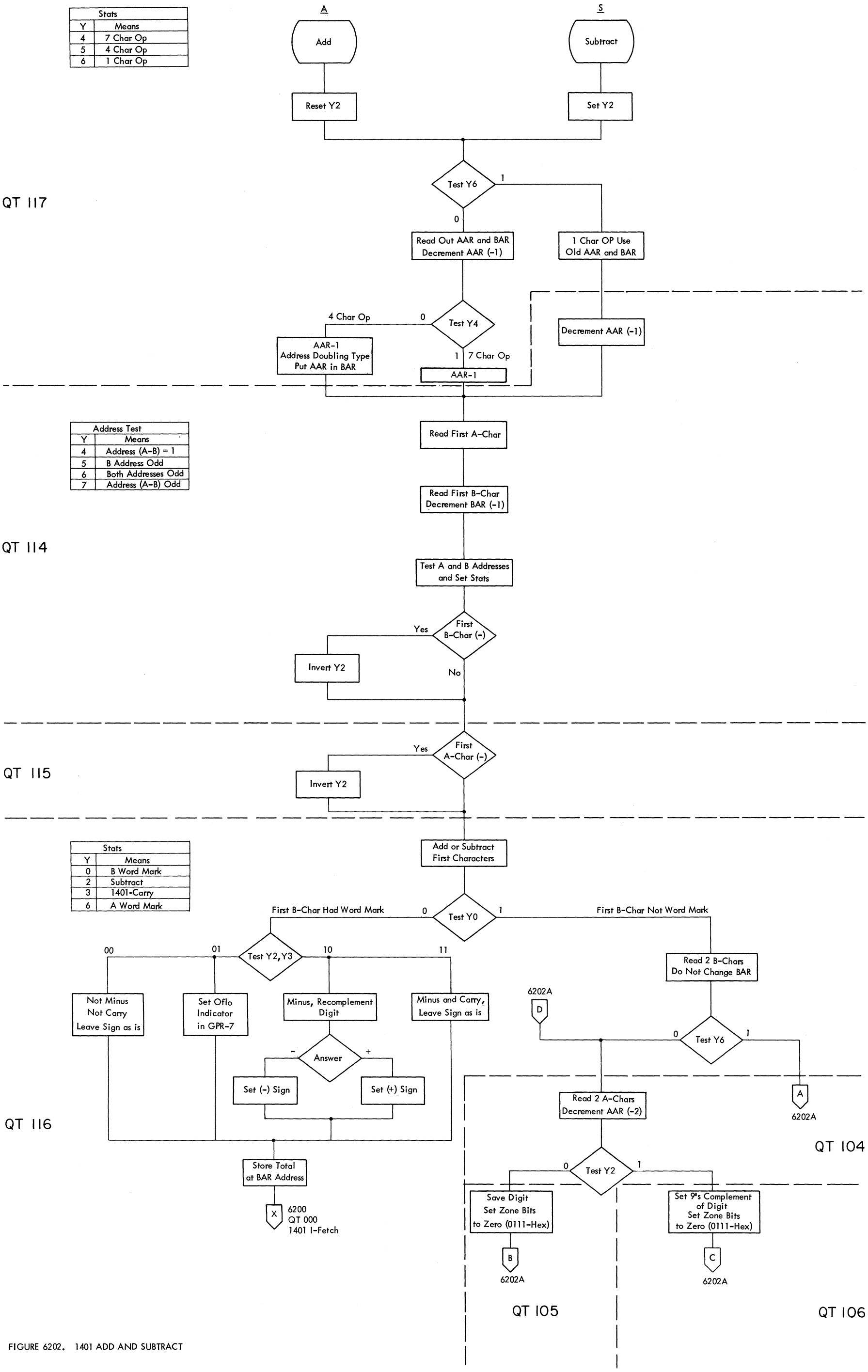


FIGURE 6202. 1401 ADD AND SUBTRACT

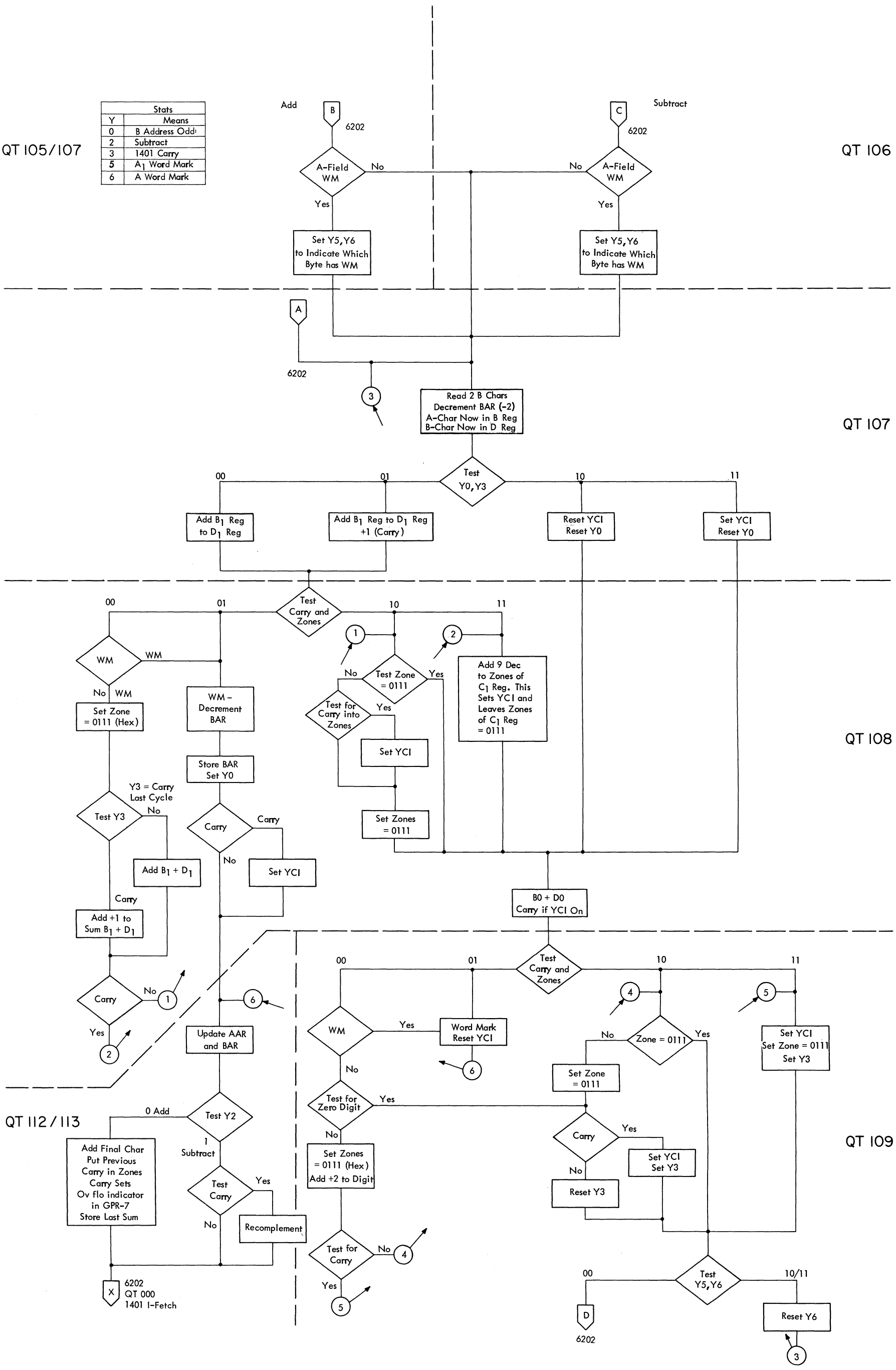
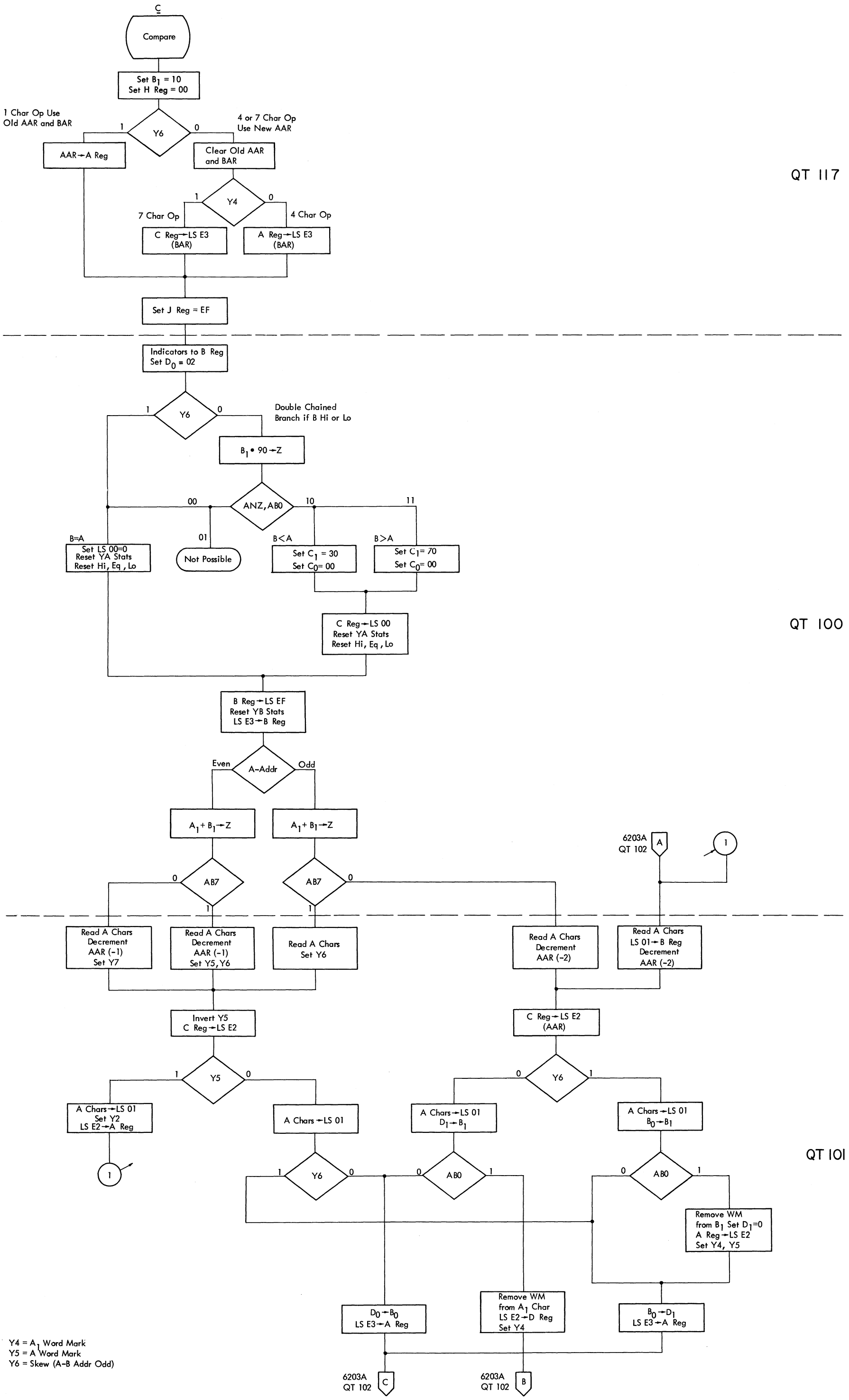


FIGURE 6202A. 1401 ADD AND SUBTRACT



QT 117

QT 100

QT 101

Y4 = A_1 Word Mark
Y5 = A Word Mark
Y6 = Skew (A-B Addr Odd)

FIGURE 6203. 1401 COMPARE

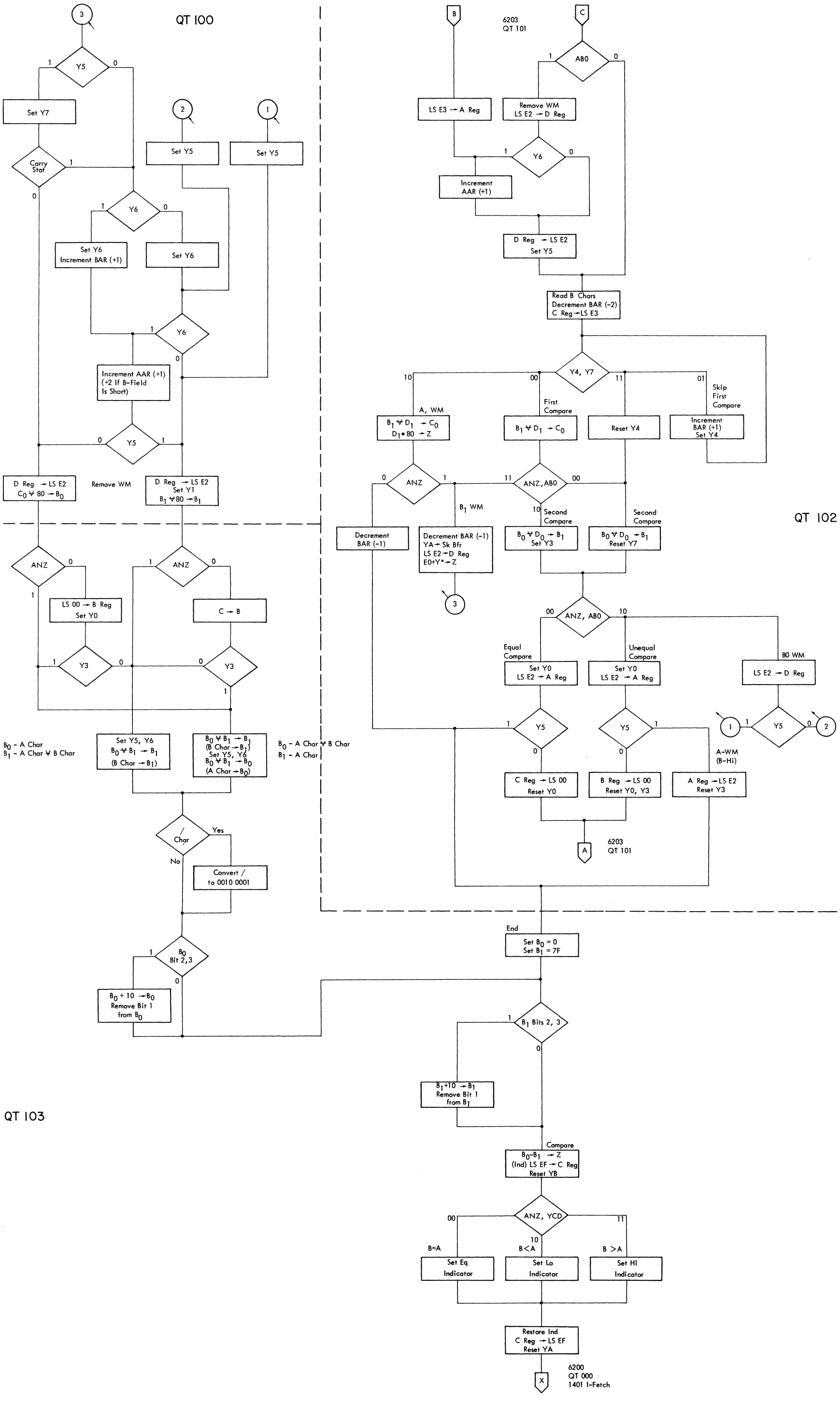
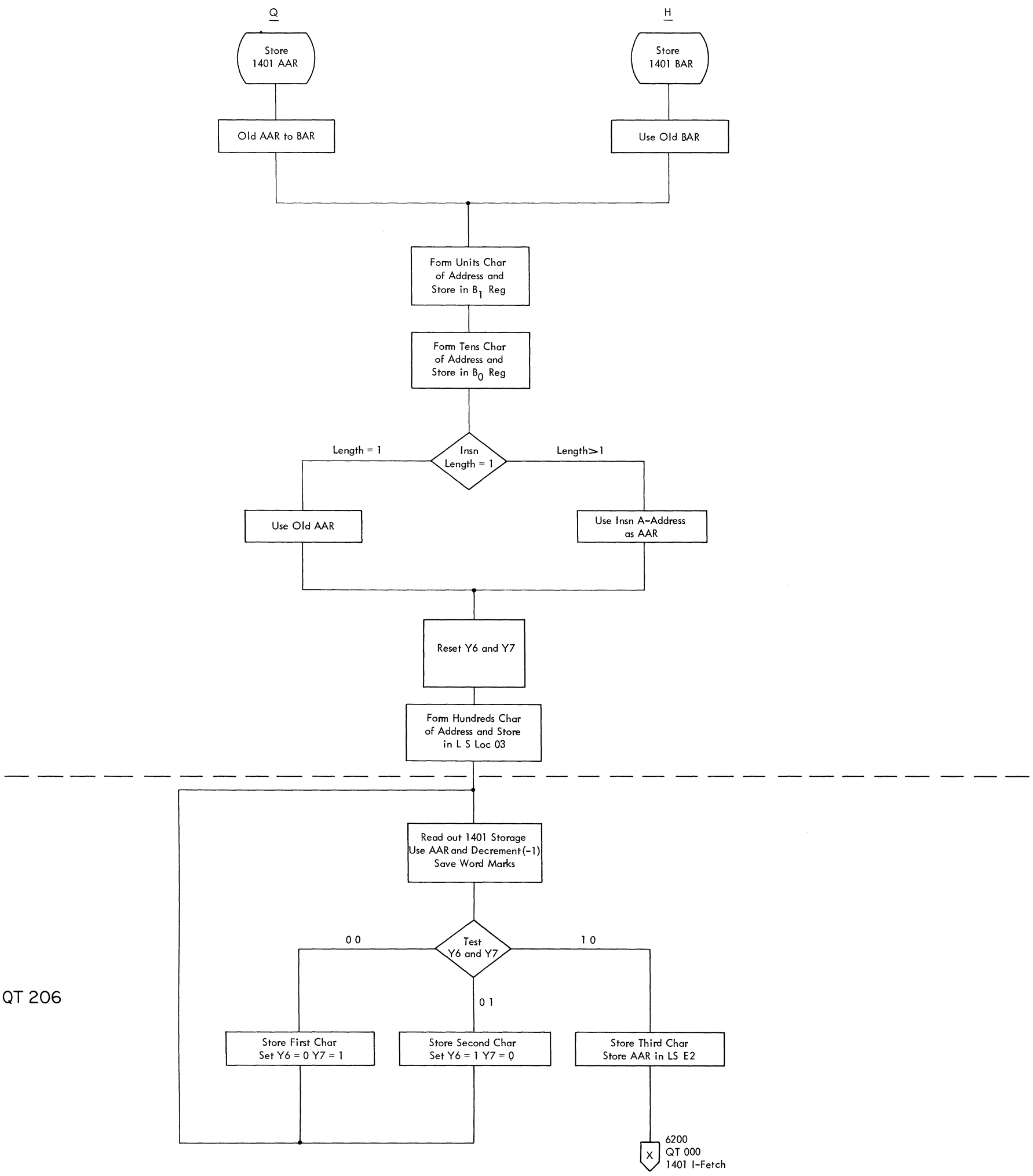


FIGURE 6203A. 1401 COMPARE



QT 206

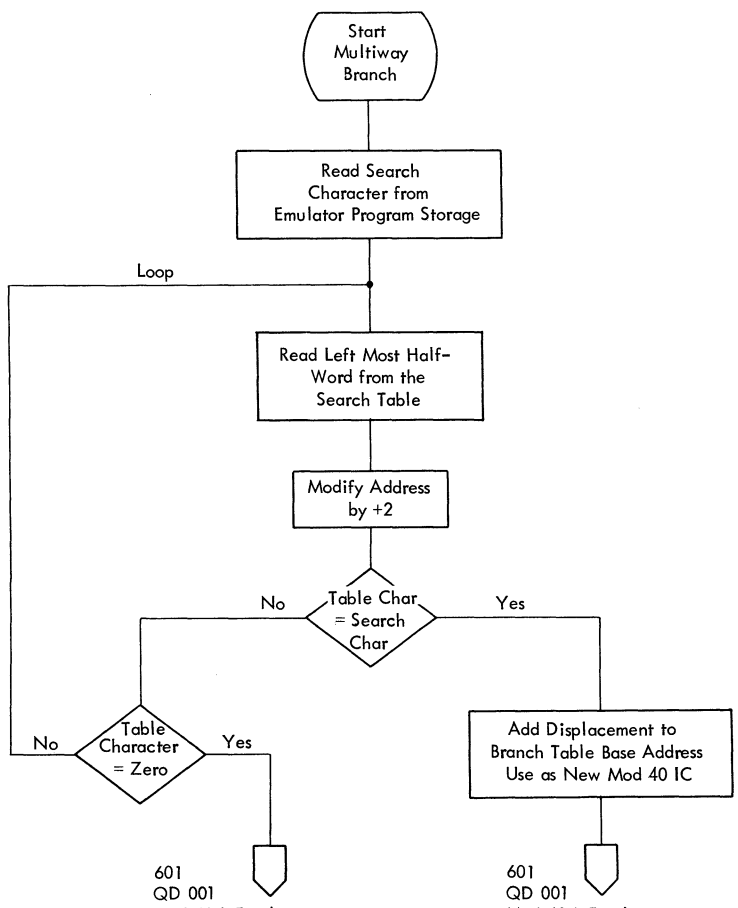
FIGURE 6204. STORE 1401 AAR OR BAR

Op - 83 000 741

GPR-8 | Search Table Address | Search Character Address

GPR-9 | Branch Table Base Address

QT 204



Multiway Branch is Entered via a Diagnose Instruction issued by the Emulator Program

Use Search Table Address

Table Entries

Table Character	Displacement

FIGURE 6205. MULTIWAY BRANCH

QT 207

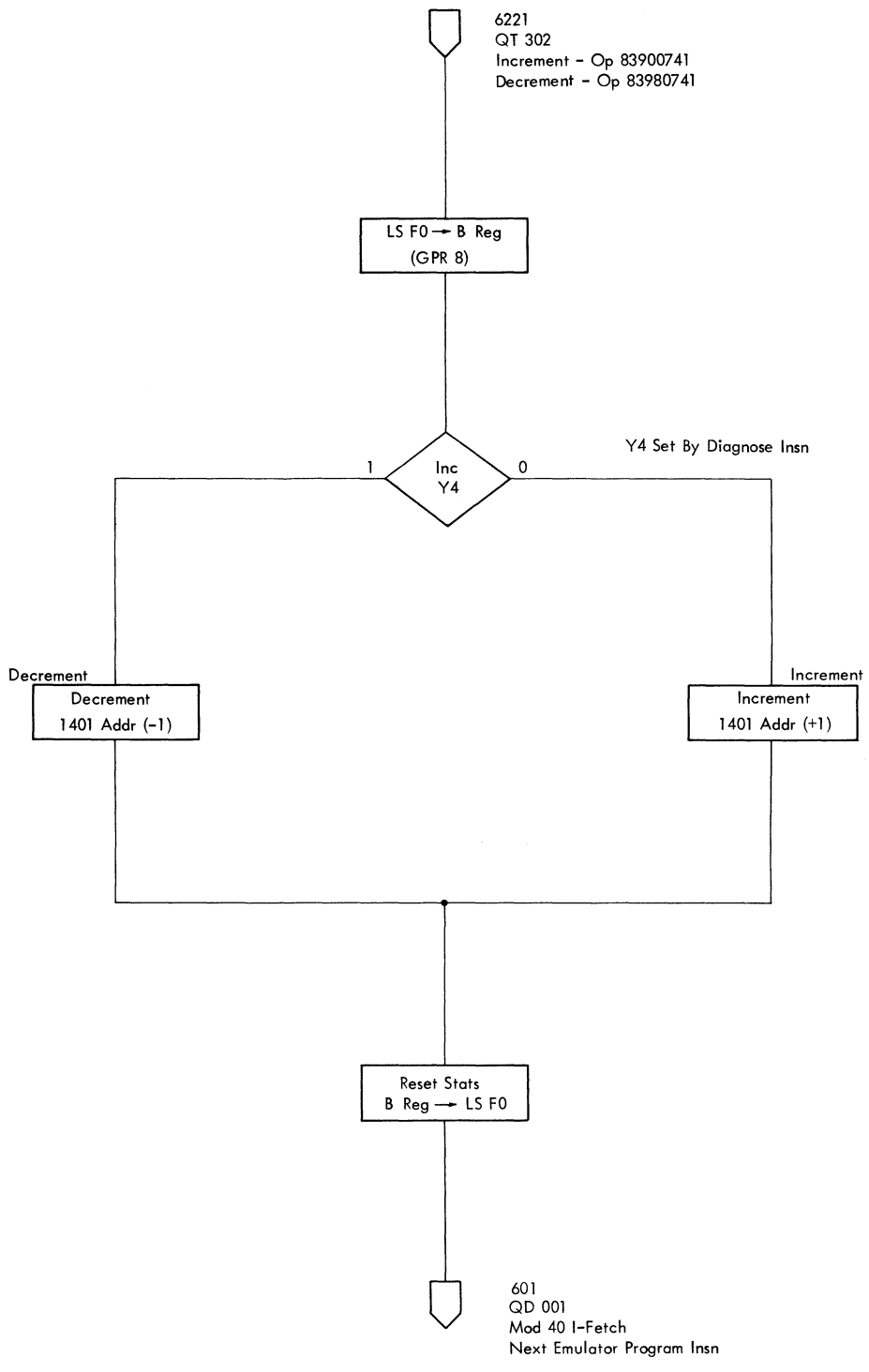


FIGURE 6206. 1401 INCREMENT - DECREMENT

Scatter

Op 83 100741

GPR-8			Read Buffer Storage Address
GPR-9	Control Byte	Count	1401 Storage Addr

Control Byte:	
Bit	Means
0	Word Mark Control
4	Use BAR
7	Scatter Operation
All Others = 0	

Y0 Stat for Scatter Op
Y0 = 0 Use 1401 WM
Y0 = 1 Use EBCDIC - III

Scatter or Gather are entered via a Diagnose Instruction issued by the Emulator Program

6221
QT 302
Scatter/Gather

Read Out Control Byte, Count, and 1401 Address
Store Control Byte in Stats

Use GPR - 9

Read Out Emulator Storage Address

Use GPR - 8

Complement of Count to B_1 Reg Reset Y4

Test Y7

Gather — 0

A
6207A
Gather
QT 208

1 Scatter

QT 207

Fetch Character from Emulator Storage
Increment Emulator Storage Address (+1)

QT 208

Translate Chars from EBCDIC-III to EBCDIC-II
A EBCDIC-III Word Mark Sets Y5

Character Valid — No

Set Y4
Change Character to all Zeros

Yes

Translate Characters are Tested for Valid EBCDIC-II

00 or 80 Hex Stored for Invalid Characters

QT 241

Test Y0, Y5

01

Error
Set Y4

00 and 10

11

Set Bit 0 of Char Equal 1

Retain EBCDIC-III Words Marks in Translated Characters

$(B_1+1) \rightarrow B_1$
Effectively Decrement Count (-1)

Count = 0 — Yes

Set Y6

No

QT 208

Read Out 1401 Storage
Save Bit 0

Test Y0

Y0 = 0

Use Bit 0 of 1401 Character

Y0 = 1

Retain 1401 Word Marks

Store Translated Char in 1401 Storage
Increment 1401 Address (+1)

Count $\neq$ 0

Count = 0
Y6 = 1

Count = 0
Last Char

Test Y4

Y4 = 1
Error

Set D_1 = 1F

A
6220
QT 300
Emulator Program
Branch Table Entry 31

Y4 = 0

601
QD 001
Mod 40 I-Fetch
Emulator Program
Next Instruction

QT 209

FIGURE 6207. SCATTER-GATHER

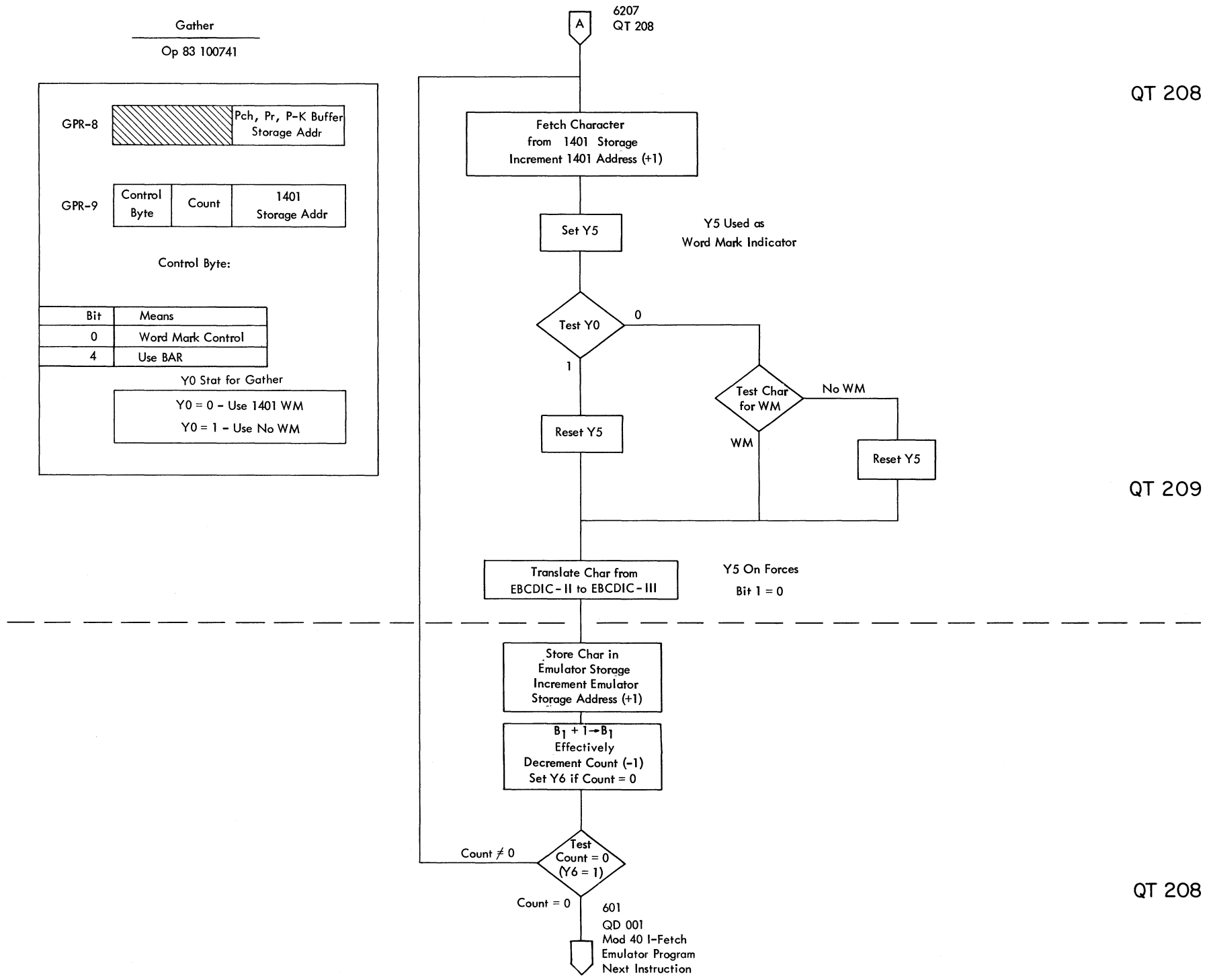


FIGURE 6207A. SCATTER-GATHER

Stat	Meaning (Scatter)
Y0	0 = Move 1 = Load
Y2	Error
Y3	LTM
	WMGM
	Cond Code
Y4	0 = Update GPR 9 1 = Update BAR
	0 = 1401 Addr Even 1 = 1401 Addr Odd
Y5	WM
Y6	0 = Mod 40 Addr Even 1 = Mod 40 Addr Odd
Y7	0 = Gather 1 = Scatter
	LTM

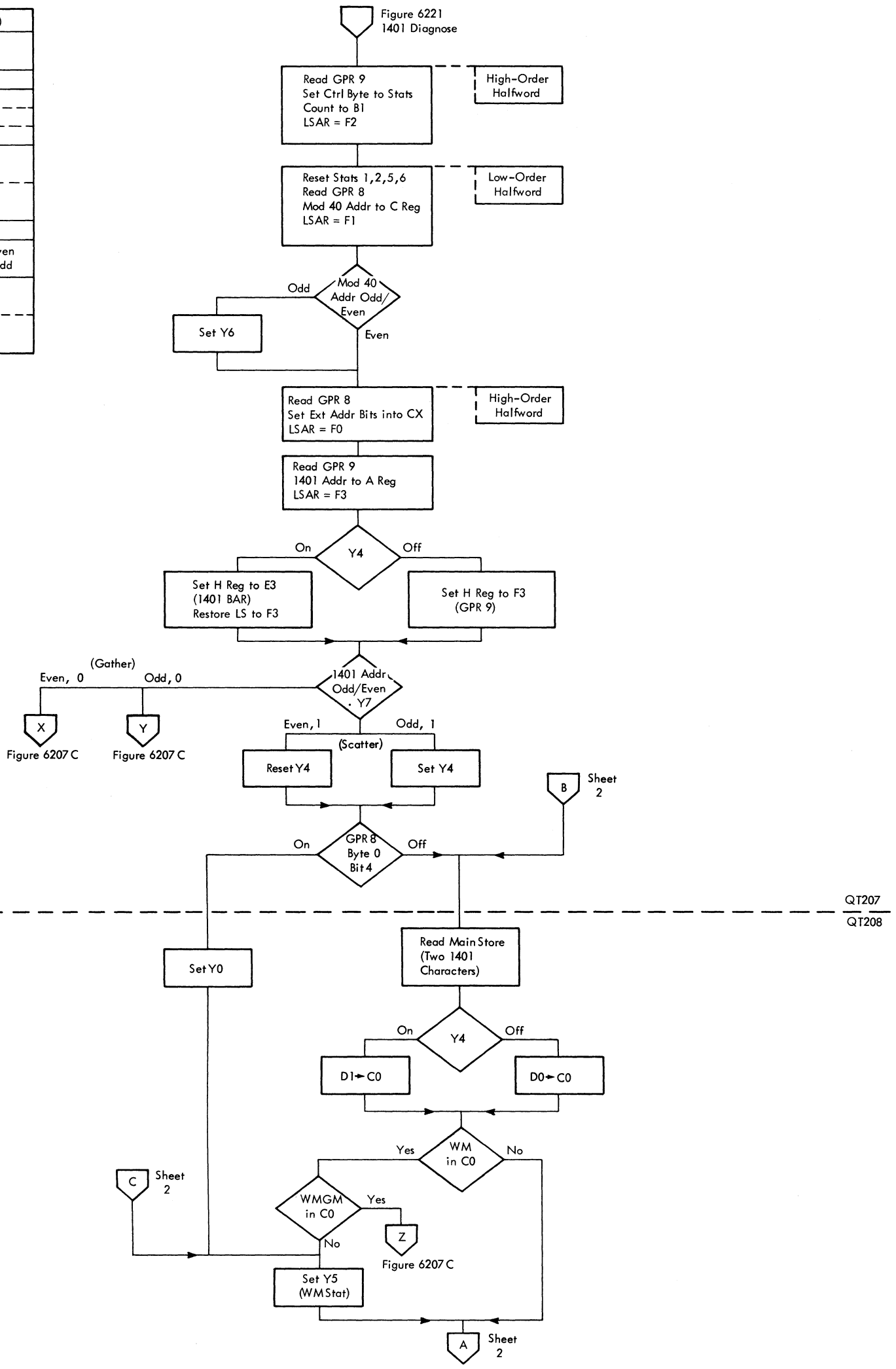


FIGURE 6207B. SCATTER (DOS COMPATIBILITY FEATURE) (PART 1 OF 2)

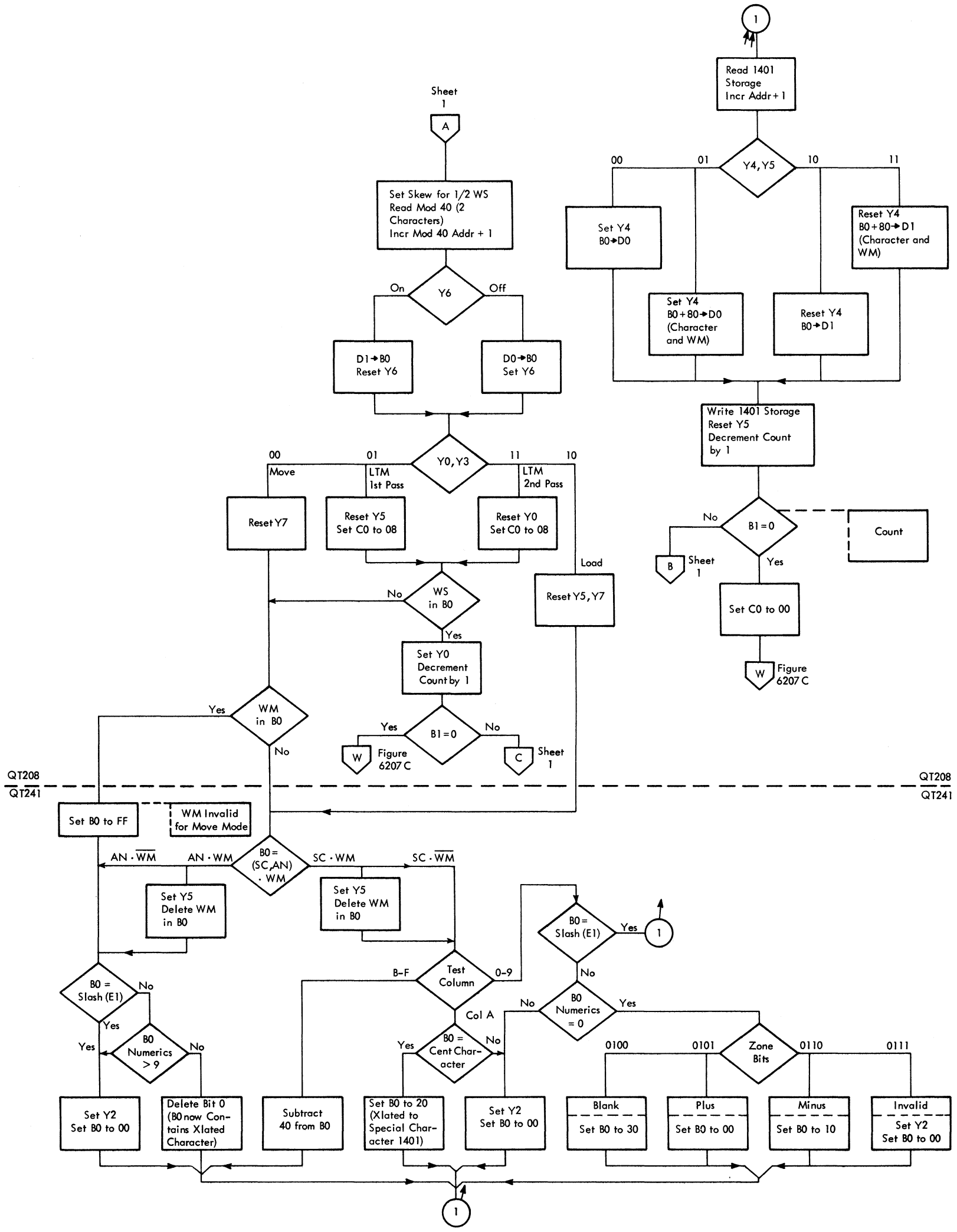


FIGURE 6207B. SCATTER (DOS COMPATIBILITY FEATURE) (PART 2 OF 2)

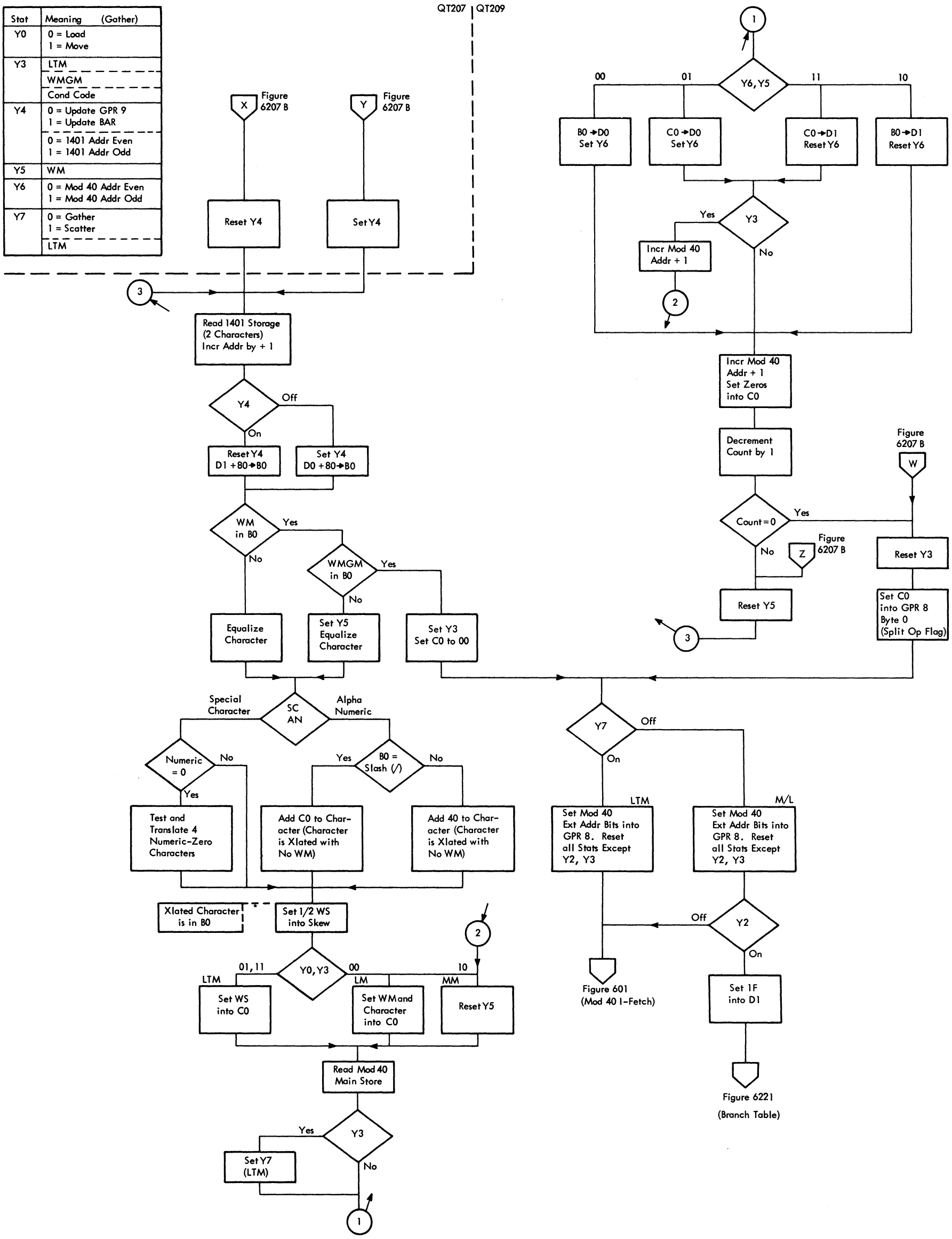


FIGURE 6207C. GATHER (DOS COMPATIBILITY FEATURE)

B0 Byte	
Bit	Means
0	Divide
2	Remainder Sign
3	A-WM
4	Minus Sign
5	RC MQ
6	B-WM (Mult)
7	IZT Zero (Div)

Stats	
0	Carry
2	First Scan
3	BAR Odd
4	AAR Odd
5	Comp Add
6	X or MQ
7	X or Y

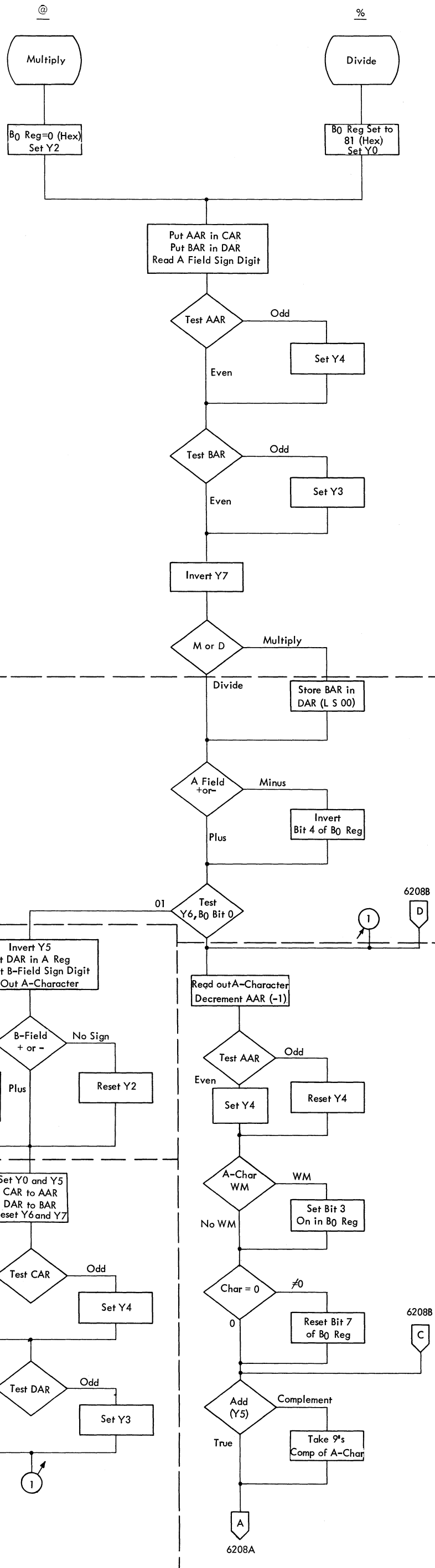
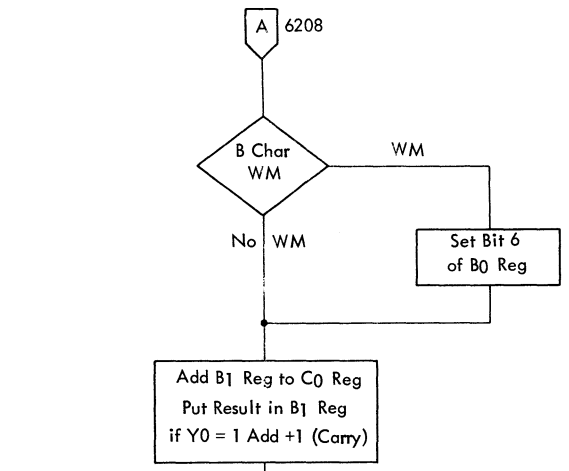


FIGURE 6208. 1401 MULTIPLY AND DIVIDE

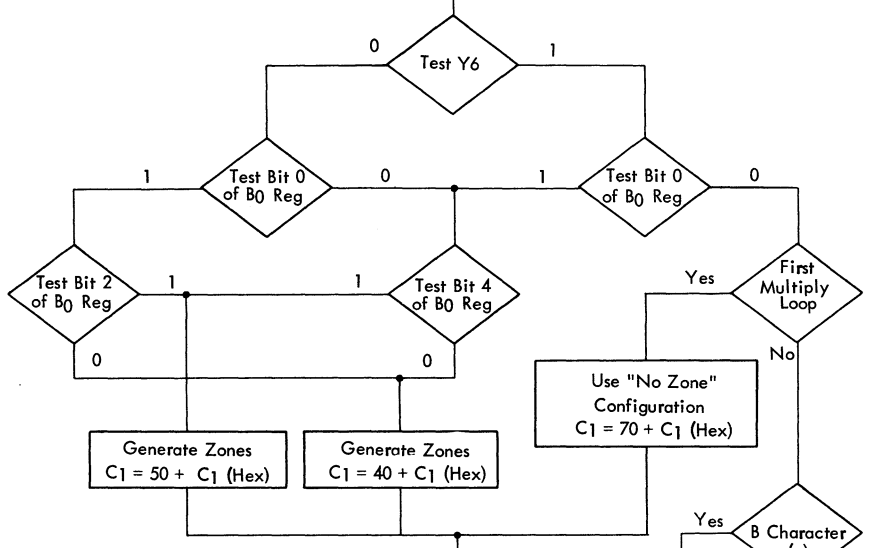
B0 Byte	
Bit	Means
0	Divide
2	Remainder Sign
3	A-WM
4	Minus Sign
5	R C MQ
6	B-WM (Multiply)
7	IZT Zero (Divide)

Y	Stats
0	Carry
2	First Scan
3	BAR Odd
4	AAR Odd
5	Comp Add
6	X or MQ
7	X or Y

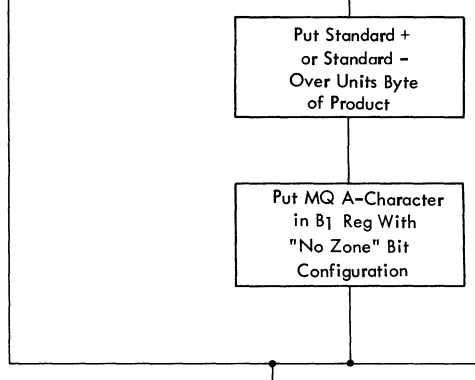
QT 214



QT 212



QT 210



QT 211

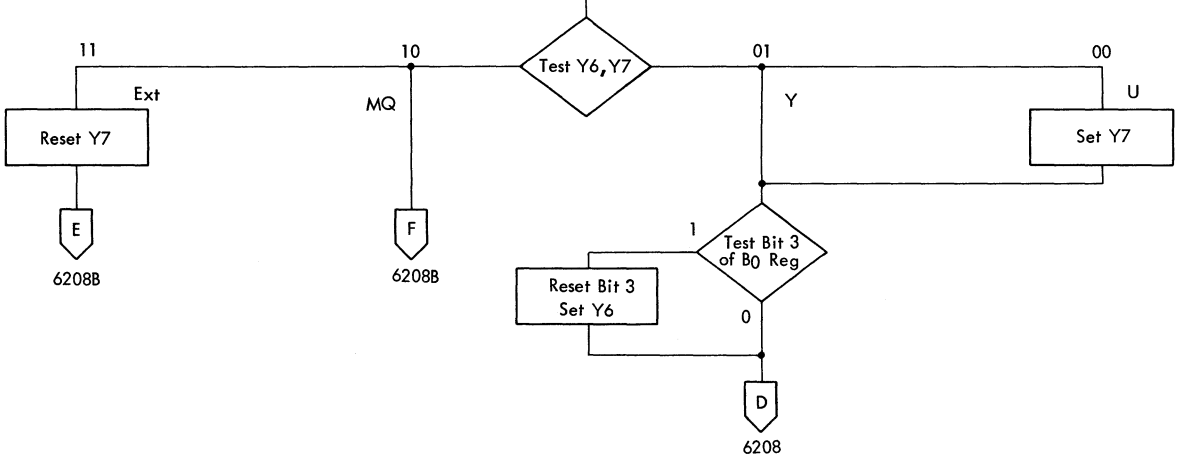
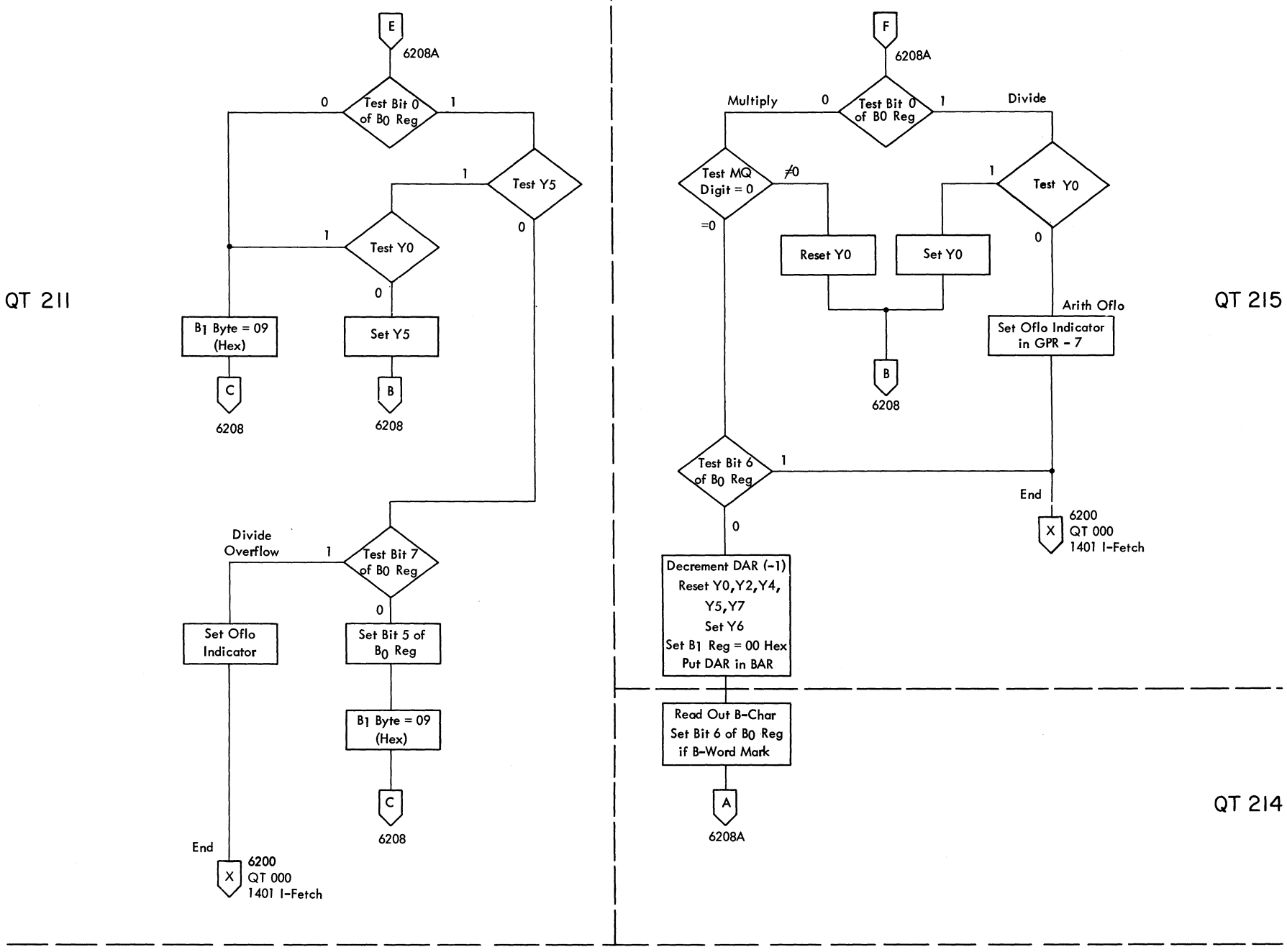


FIGURE 6208A. 1401 MULTIPLY AND DIVIDE



B0 Byte

Bit	Means
0	Divide
2	Remainder Sign
3	A WM
4	Minus Sign
5	RC MQ
6	B WM (Multiply)
7	IZT Zero (Divide)

Y	Stats
0	Carry
2	First Scan
3	BAR Odd
4	AAR Odd
5	Comp Add
6	X or MQ
7	X or Y

FIGURE 6208B. 1401 MULTIPLY AND DIVIDE

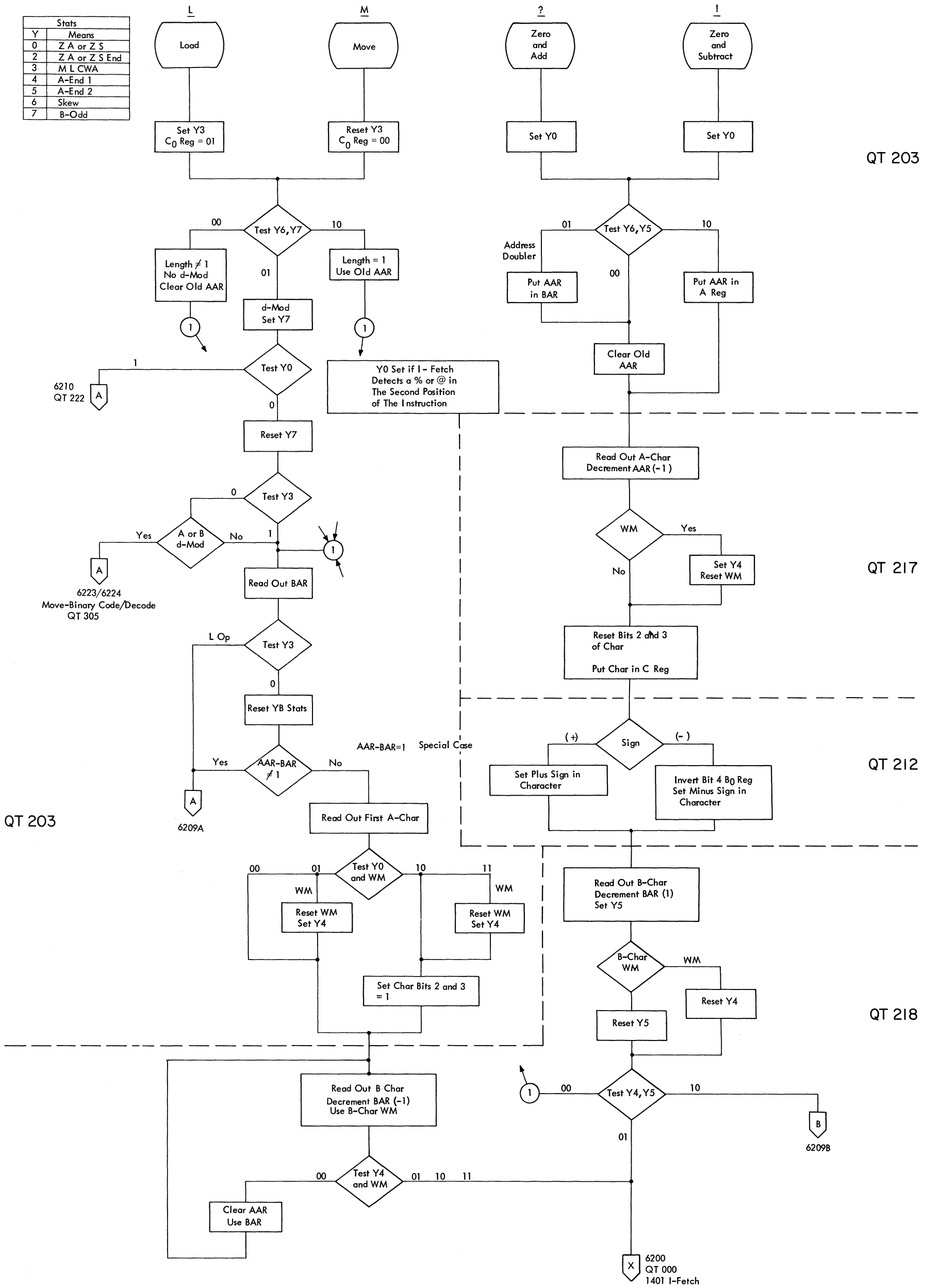
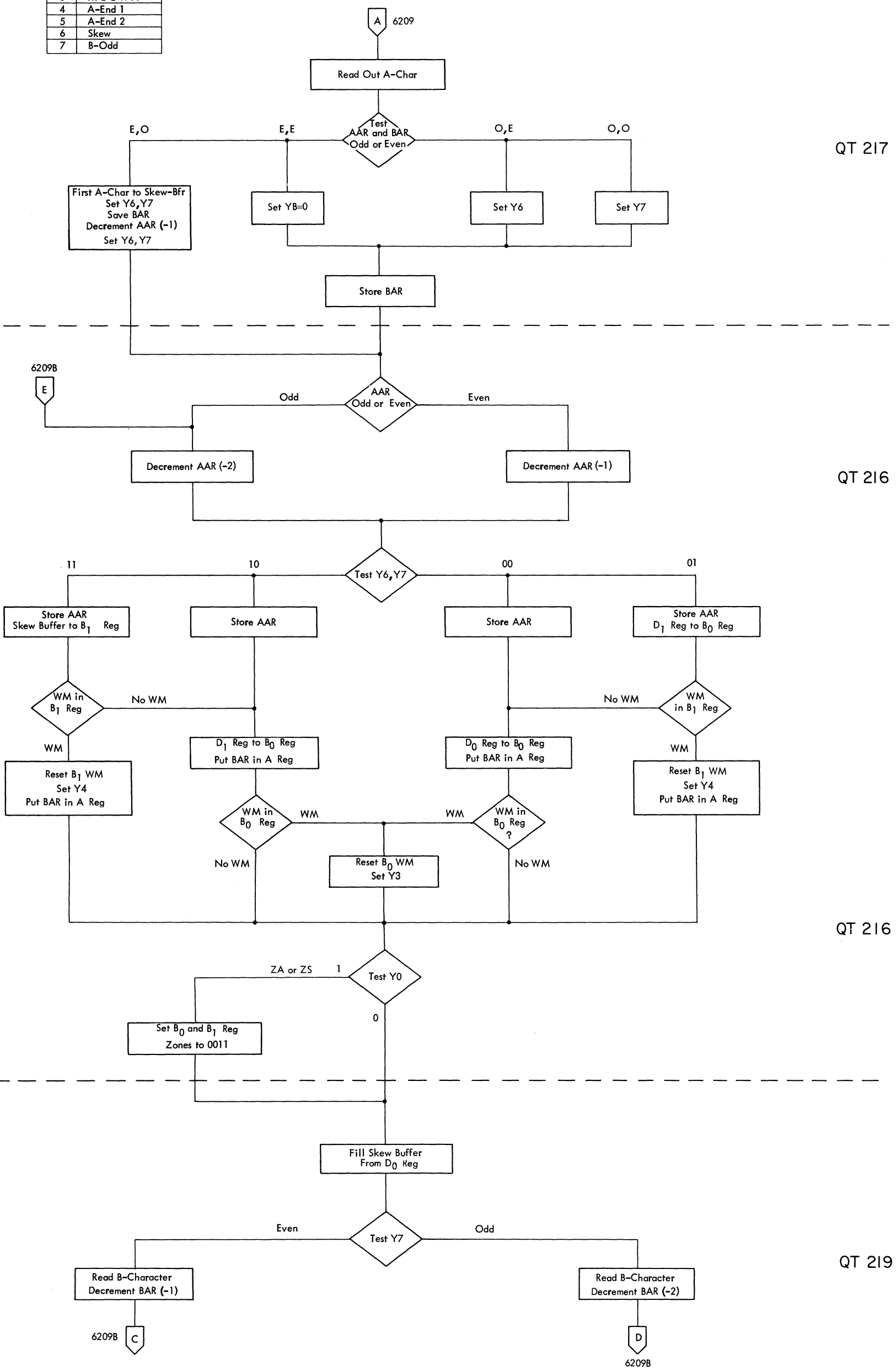


FIGURE 6209. 1401 MOVE, LOAD, ZERO AND ADD, ZERO AND SUBTRACT

Stats	
Y	Means
0	Z A or Z S
2	Z A or Z S End
3	M L C W A
4	A-End 1
5	A-End 2
6	Skew
7	B-Odd



QT 217

QT 216

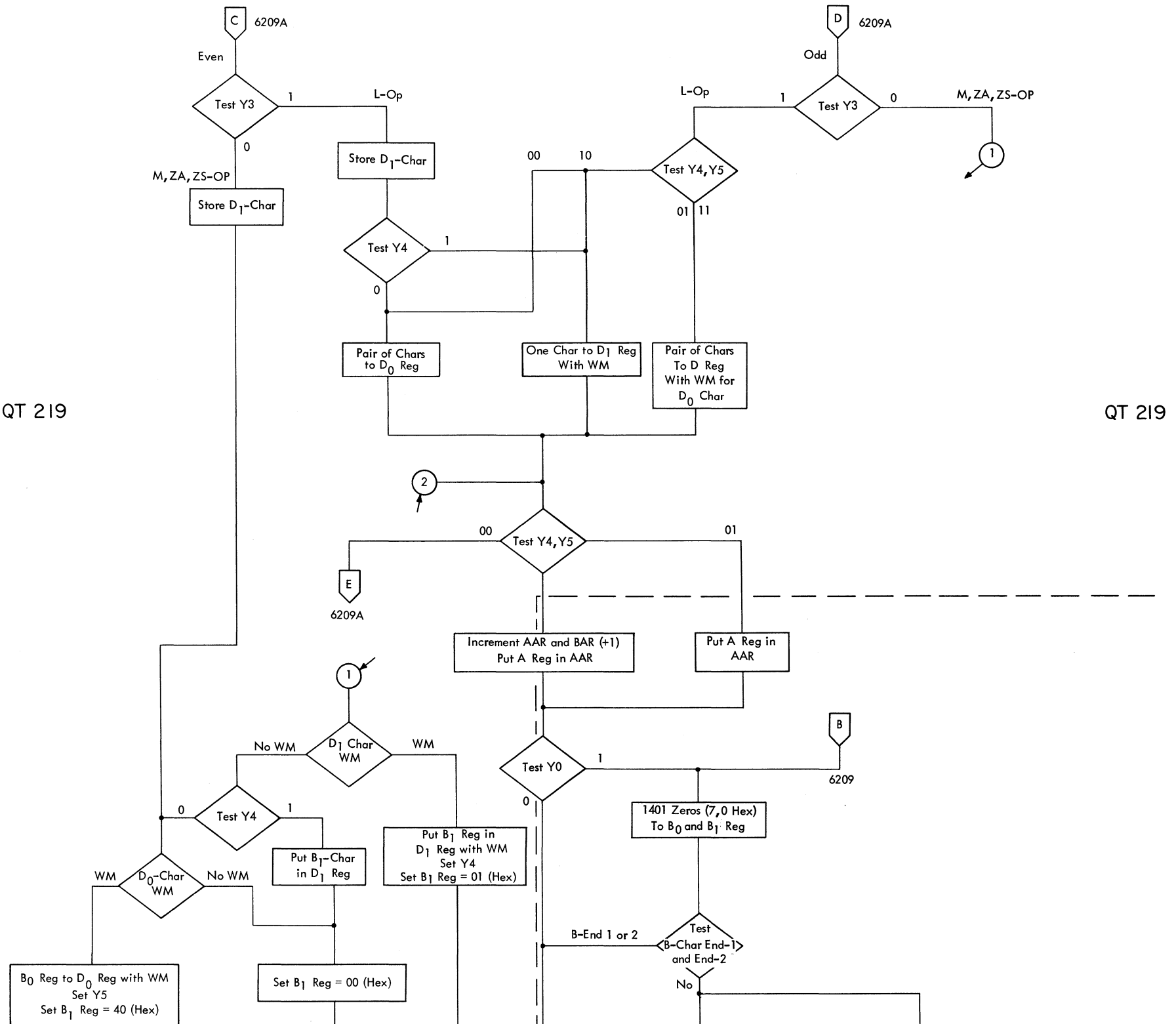
QT 216

QT 219

FIGURE 6209A. 1401 MOVE, LOAD, ZERO AND ADD, ZERO AND SUBTRACT

QT 219

QT 219



QT 220

Stats

Y	Means
0	Z A or Z S
2	Z A or Z S End
3	M L C W A
4	A-End 1 or B-End 1
5	A-End 2 or B-End 2
6	Skew
7	B-Odd

FIGURE 6209B. 1401 MOVE, LOAD, ZERO AND ADD, ZERO AND SUBTRACT

QT 222

QT 226

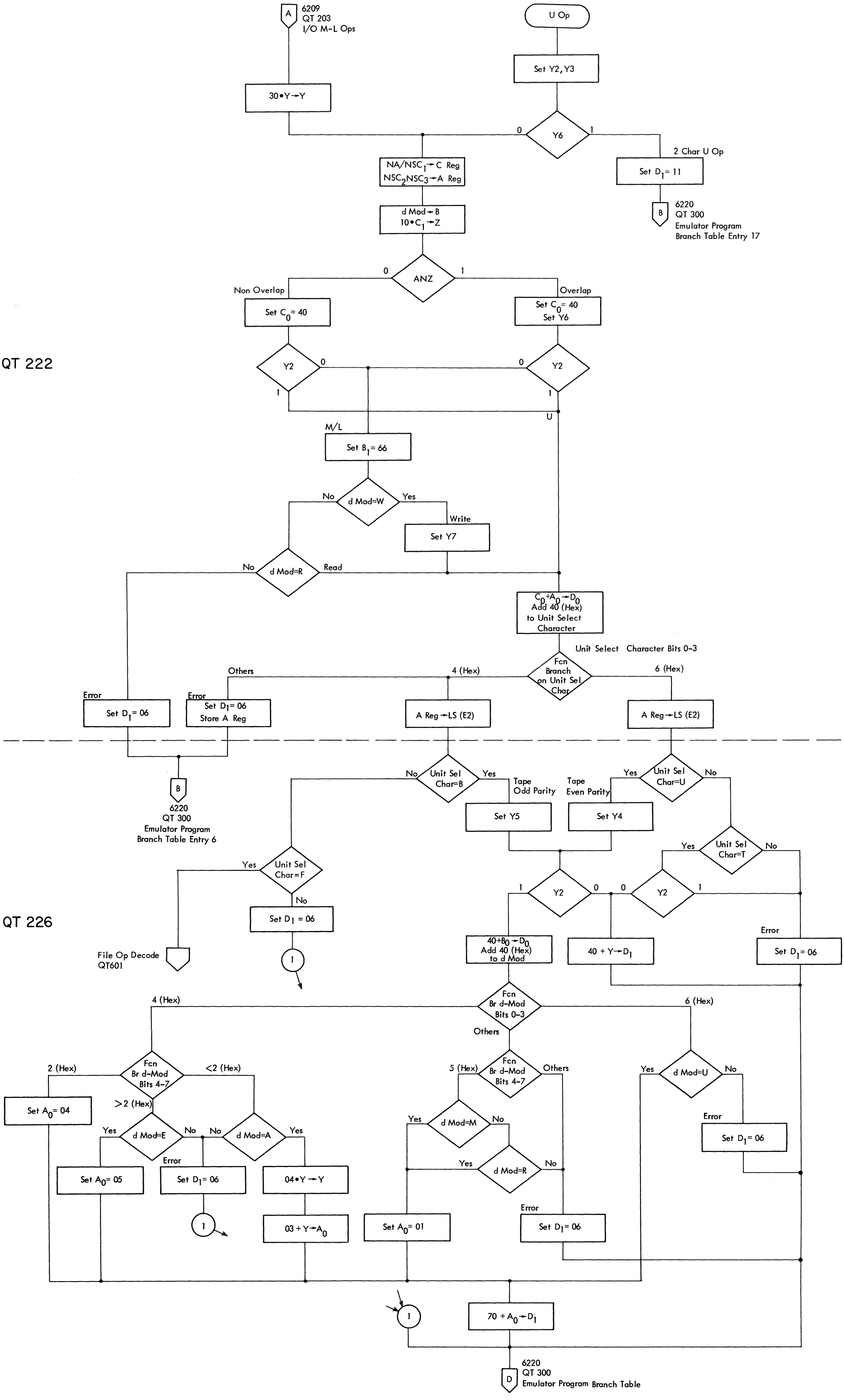
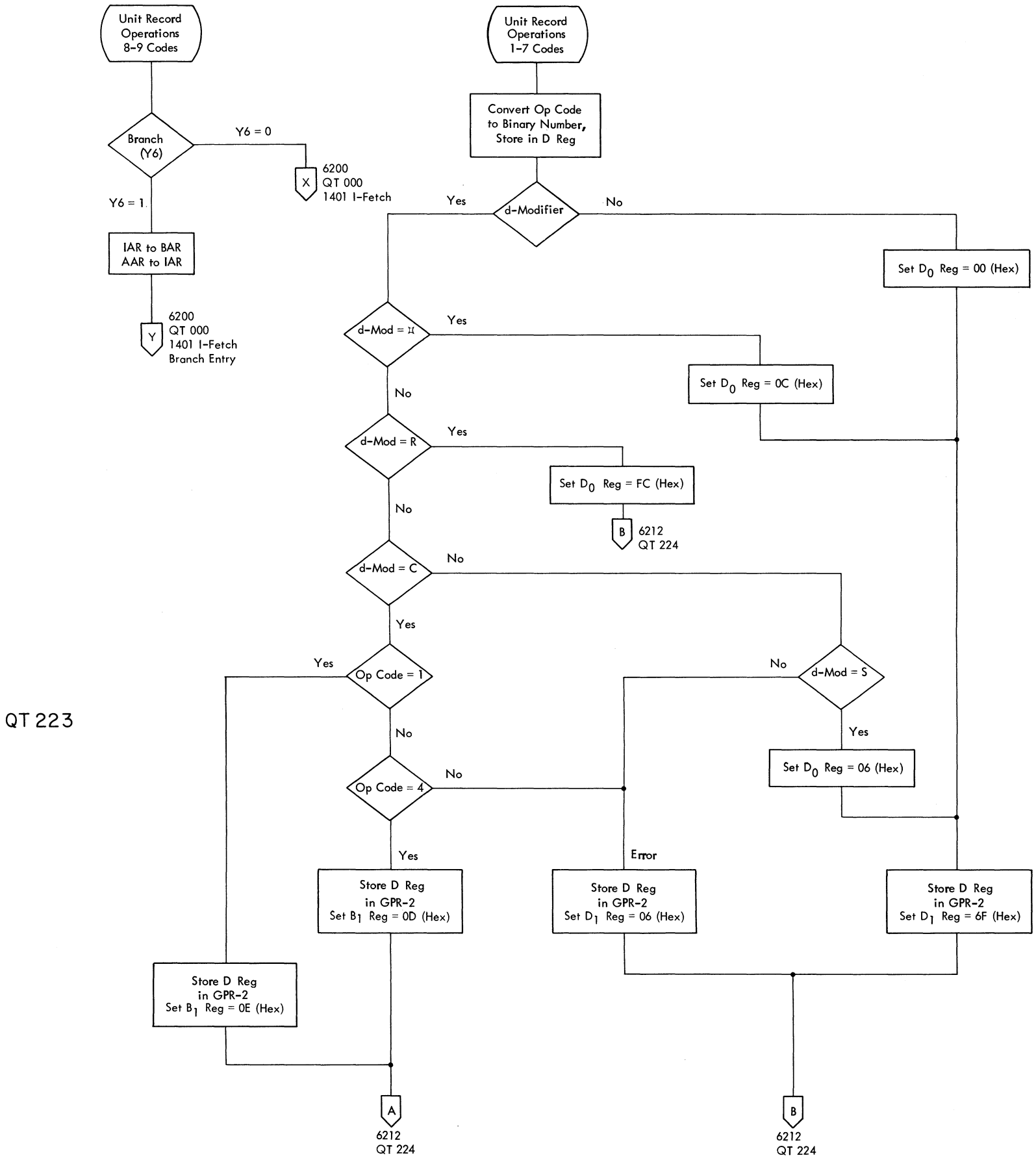


FIGURE 6210. 1401 I/O M, L, U OPERATIONS



QT 223

FIGURE 6211. 1401 UNIT RECORD OPERATIONS

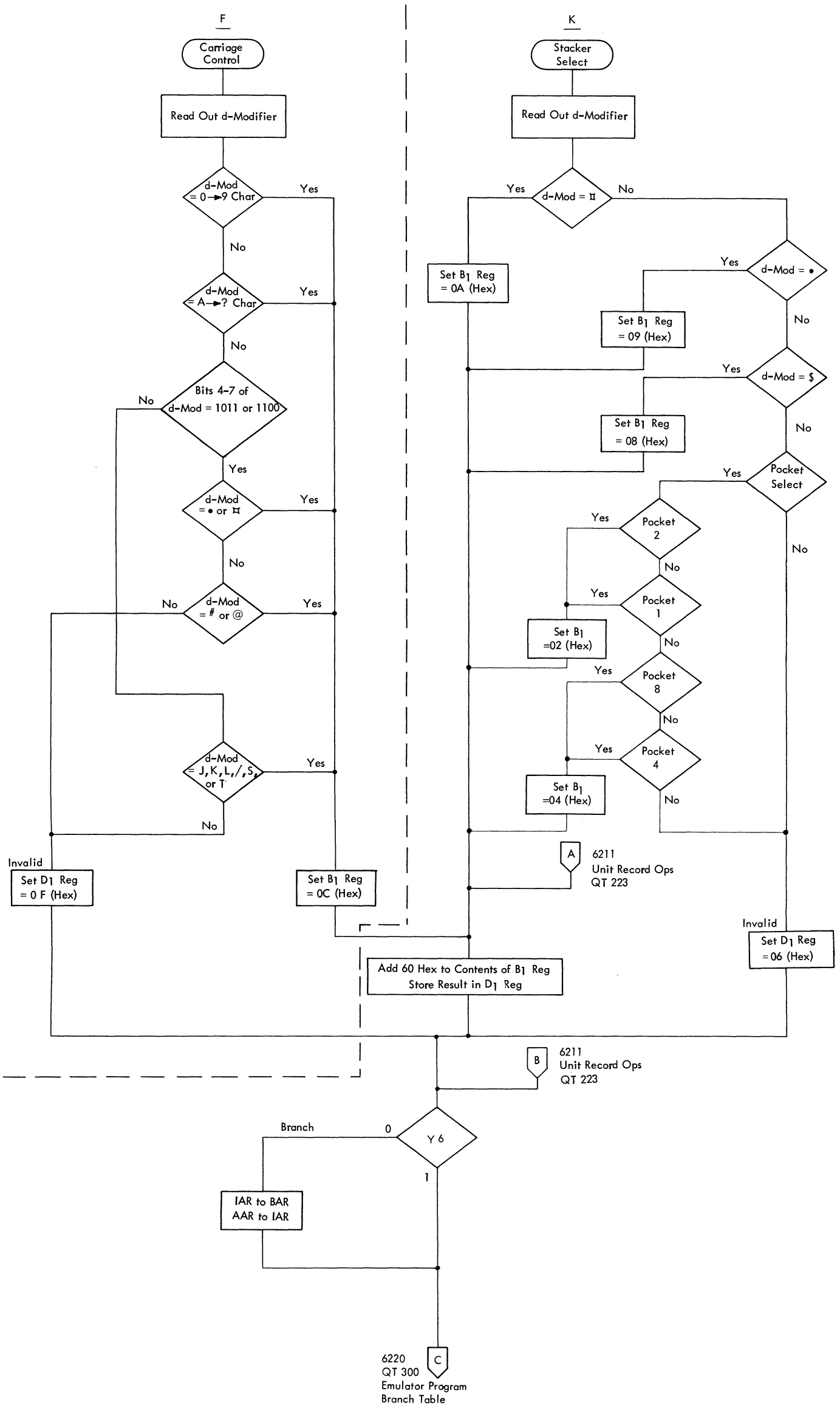
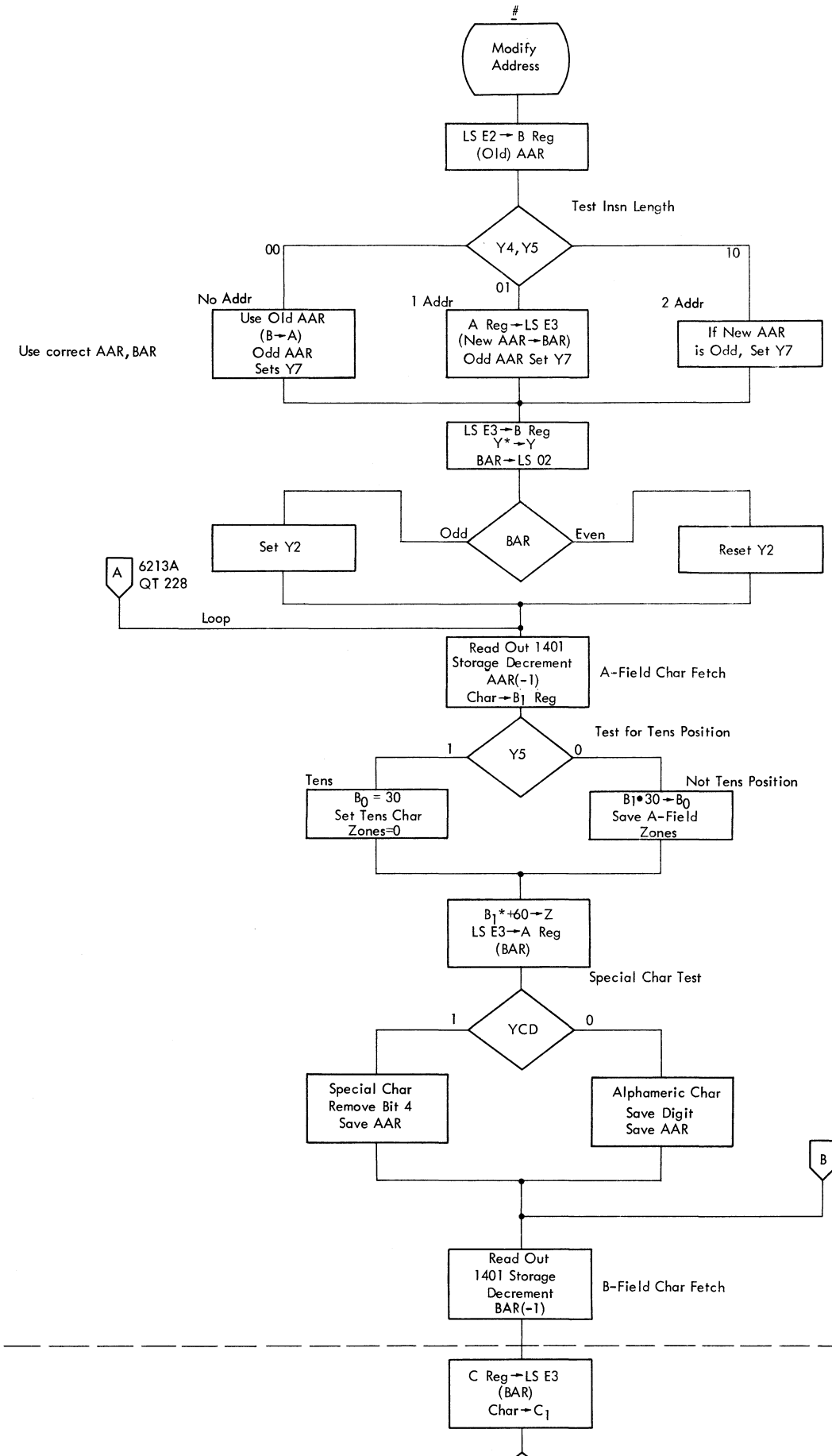


FIGURE 6212. 1401 CARRIAGE CONTROL AND STACKER SELECT

QT 227

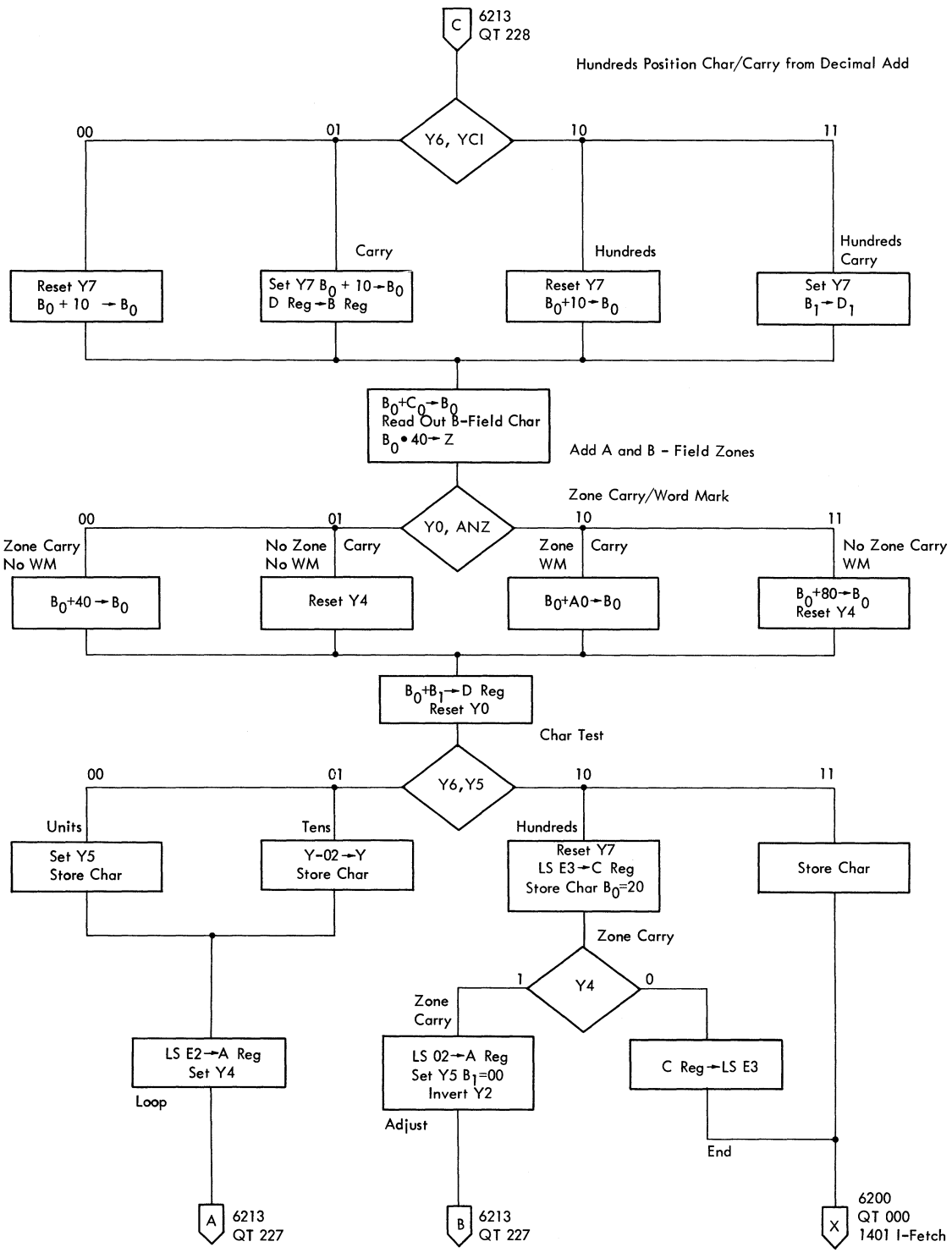


Y	Means
0	B-Field WM
2	BAR (Mod 2)
3	AAR (Mod 2)
4	Zone Carry
5	Tens
6	Hundreds
7	Numeric Carry

FIGURE 6213. 1401 ADDRESS MODIFY

QT 228

Y	Means
0	B-Field WM
2	Bar (Mod 2)
3	AAR (Mod 2)
4	Zone Carry
5	Tens
6	Hundreds
7	Numeric Carry



QT 228

FIGURE 6213A. 1401 ADDRESS MODIFY

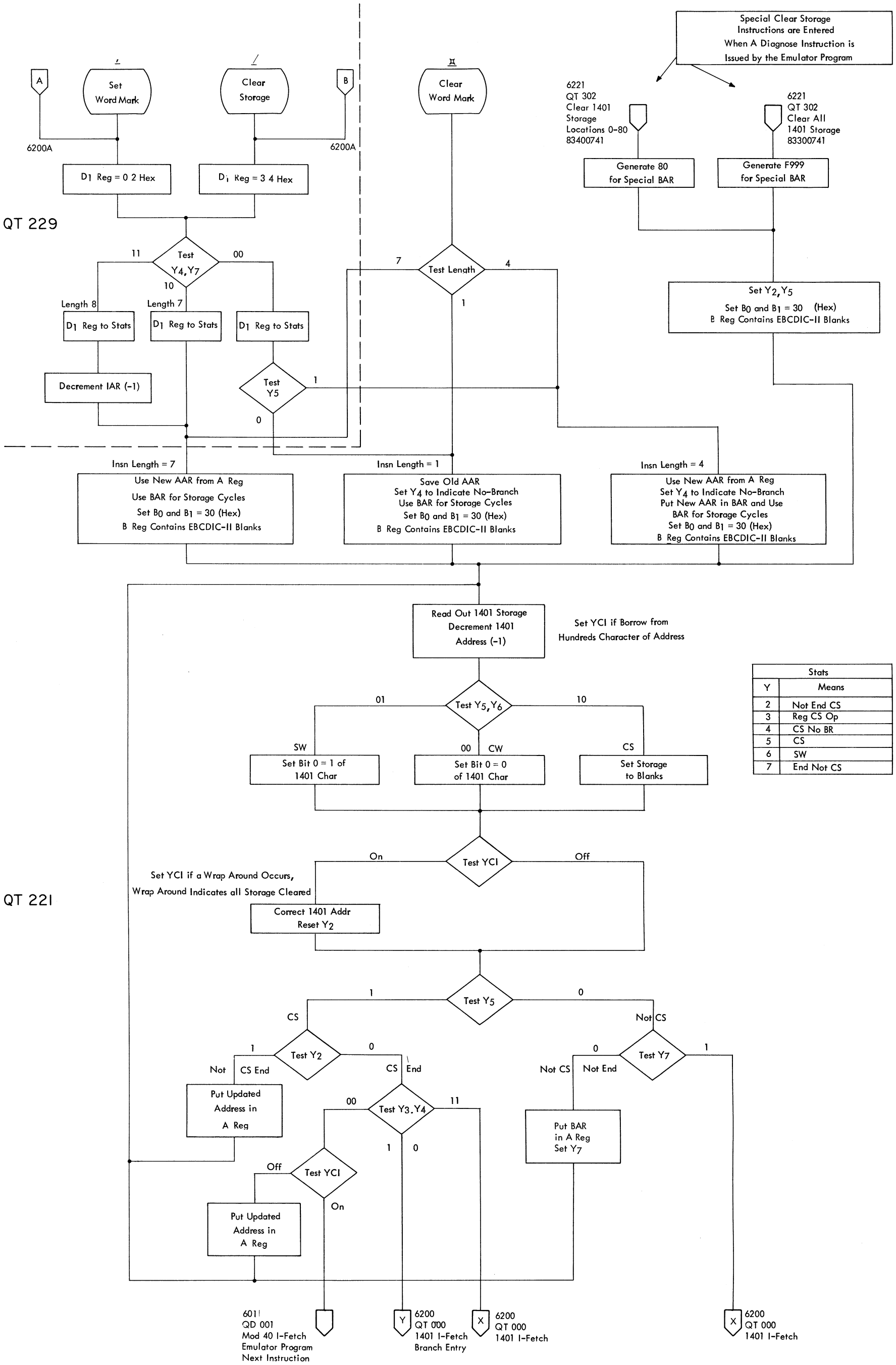


FIGURE 6214. 1401 SET WORD MARK, CLEAR WORD MARK, CLEAR STORAGE, AND SPECIAL CLEARS

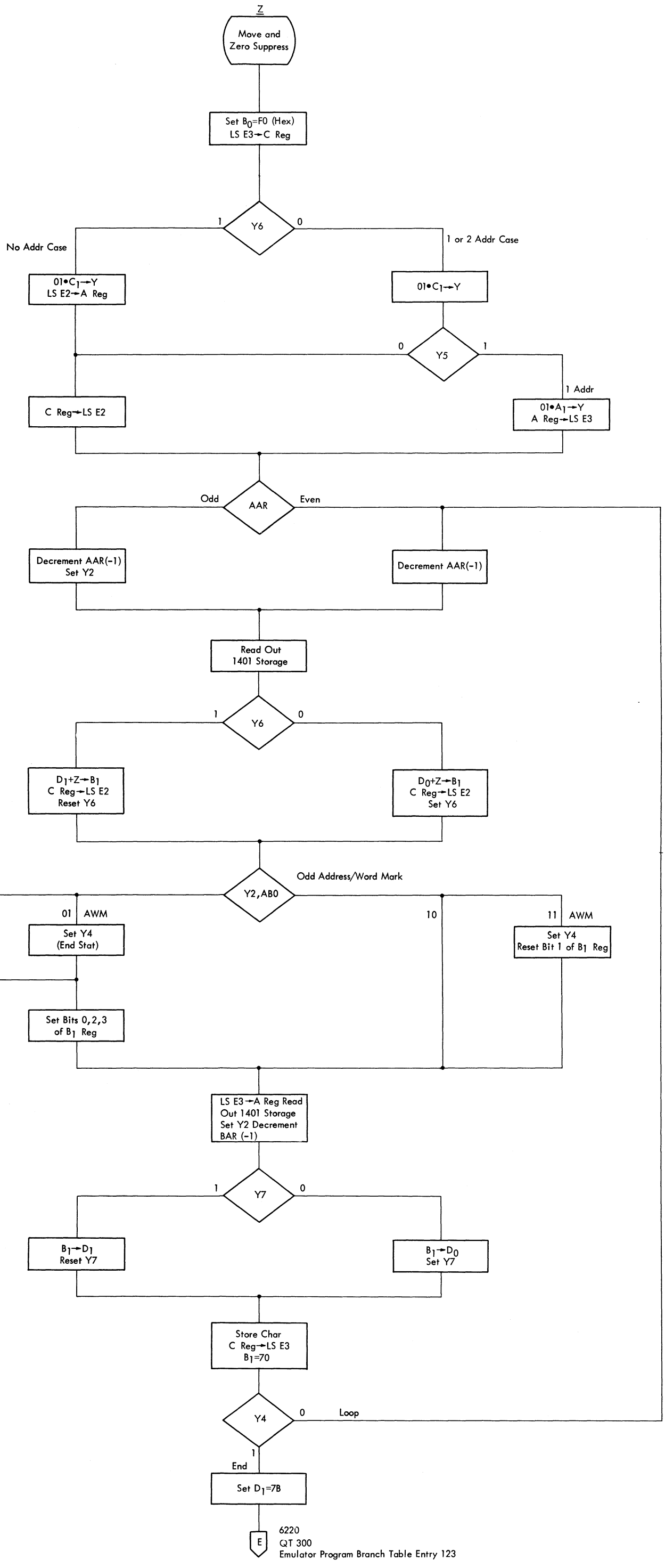
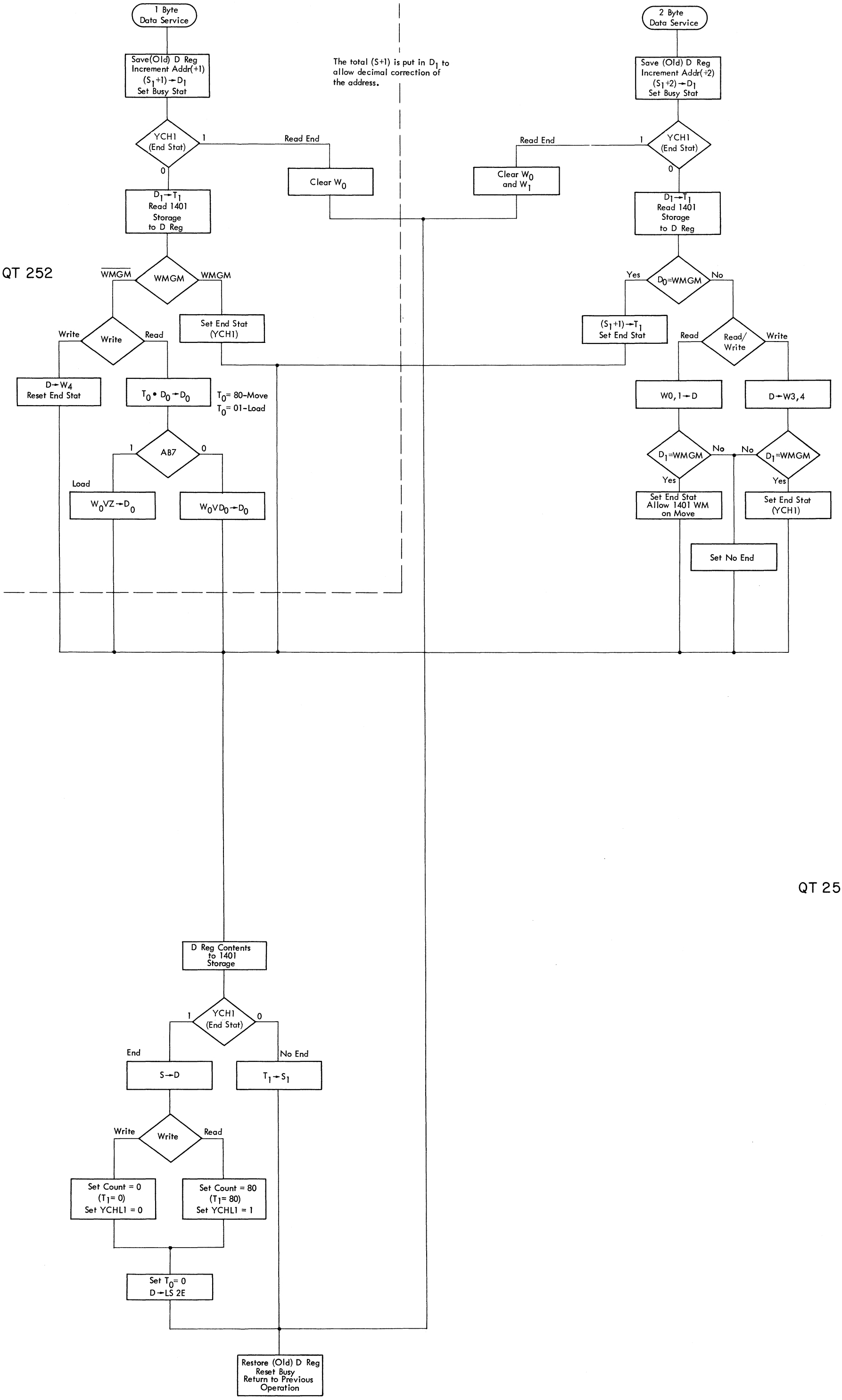


FIGURE 6215. 1401 MOVE CHARACTERS AND SUPPRESS ZEROS



QT 252

QT 251

FIGURE 6216. 1401--1 AND 2 BYTE DATA SERVICE

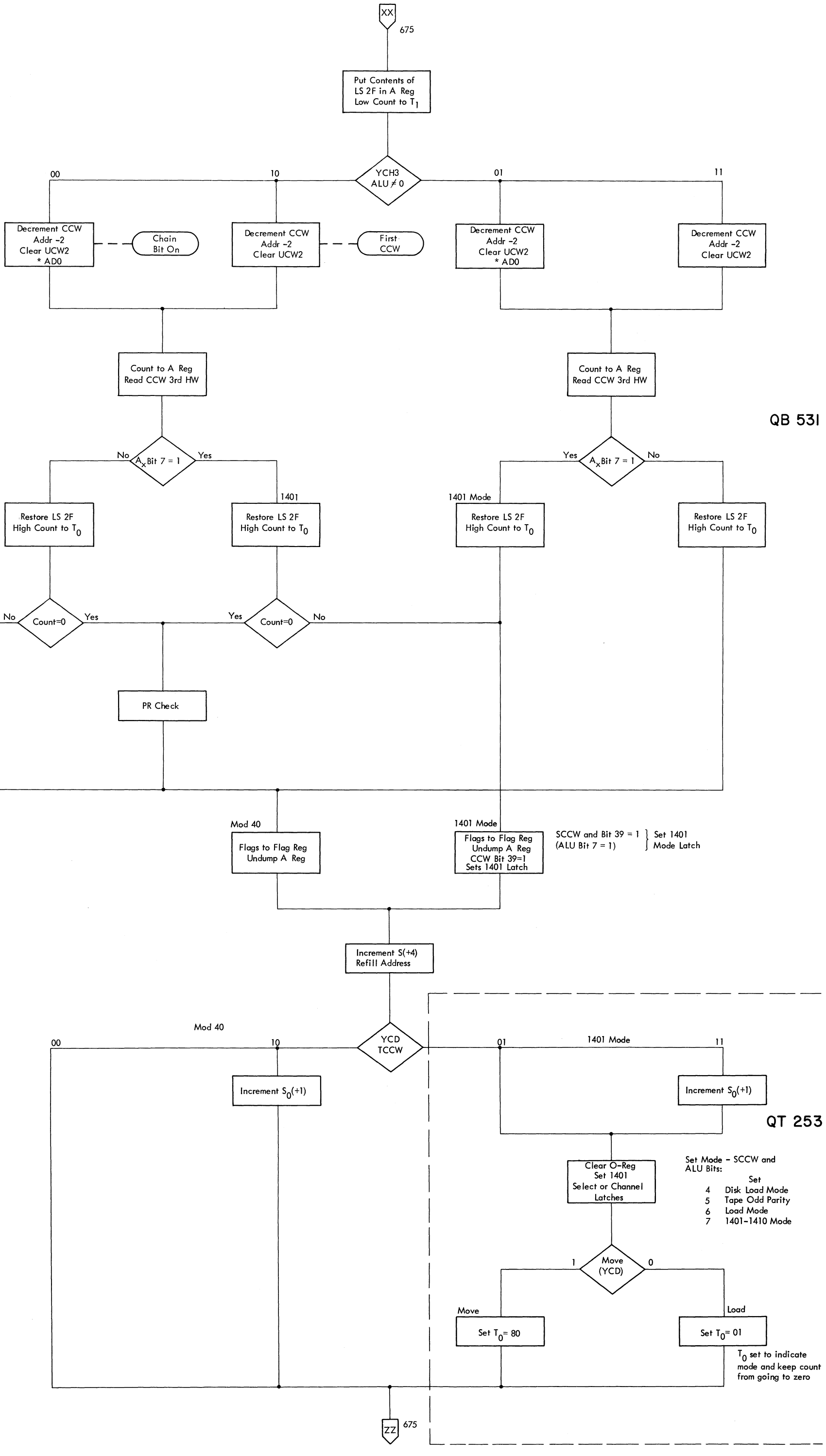


FIGURE 6217. SET SELECTOR CHANNEL TO 1401 MODE

QT 256

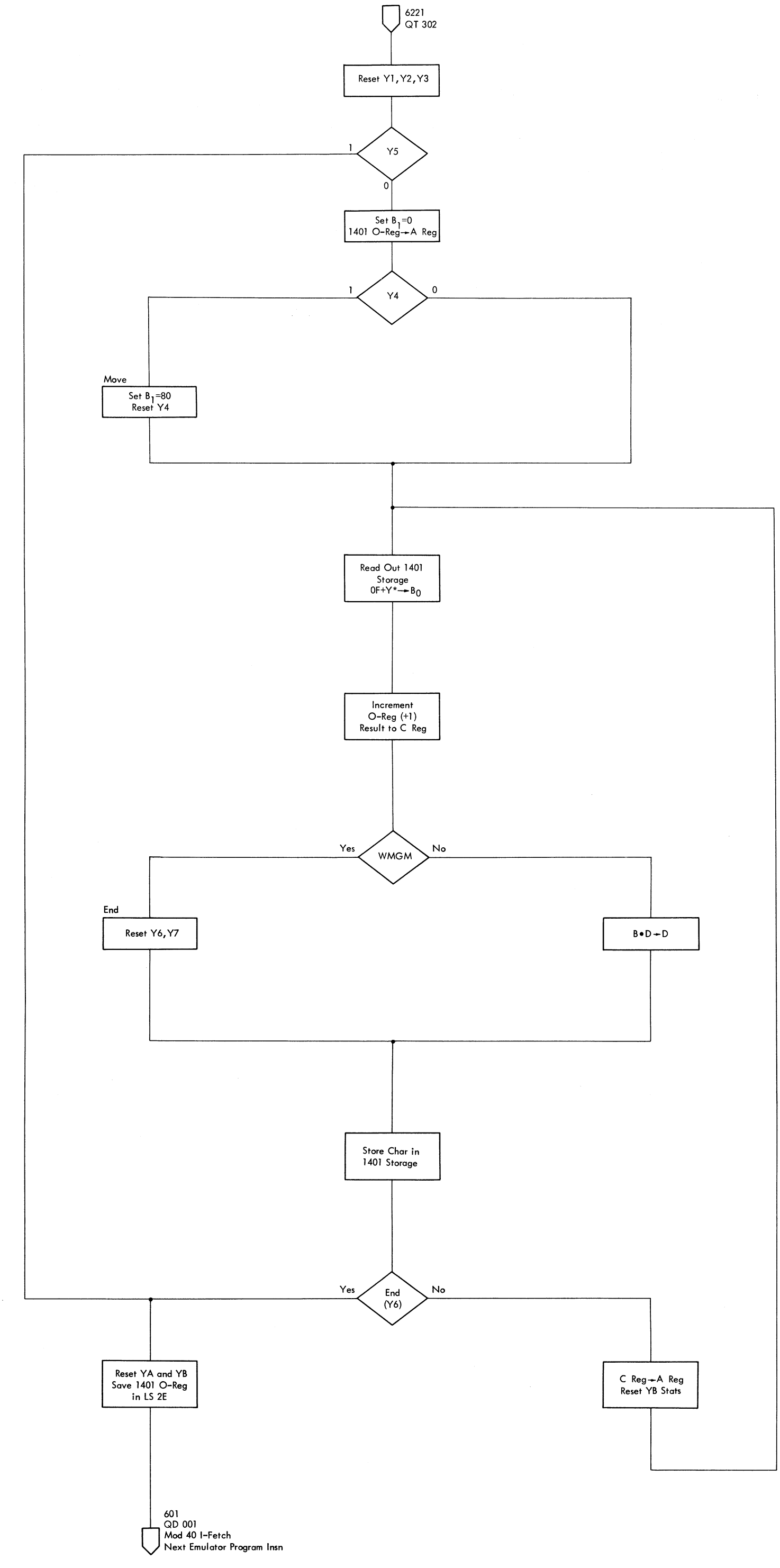


FIGURE 6218. 1401 TAPE OPERATIONS

V (I) (B) d

d	Condition
1	Word Mark
2	No Zone
3	Word Mark or No Zone
B	12 Zone
K	11 Zone
S	0 Zone
C	Word Mark or 12 Zone
L	Word Mark or 11 Zone
T	Word Mark or 0 Zone

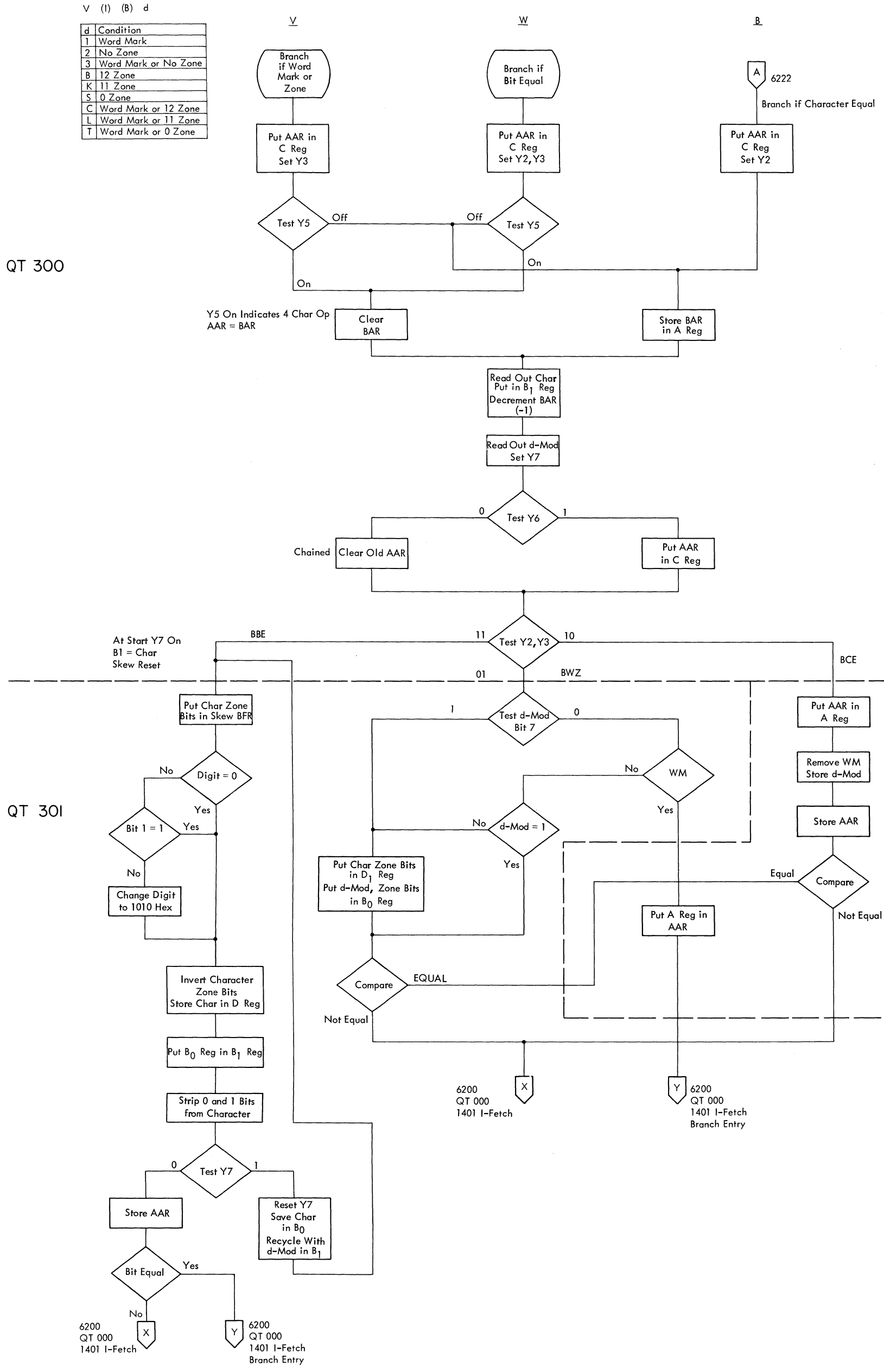


FIGURE 6219. 1401 BRANCH IF: WORD MARK OR ZONE, BIT EQUAL, OR CHARACTER EQUAL

QT 300

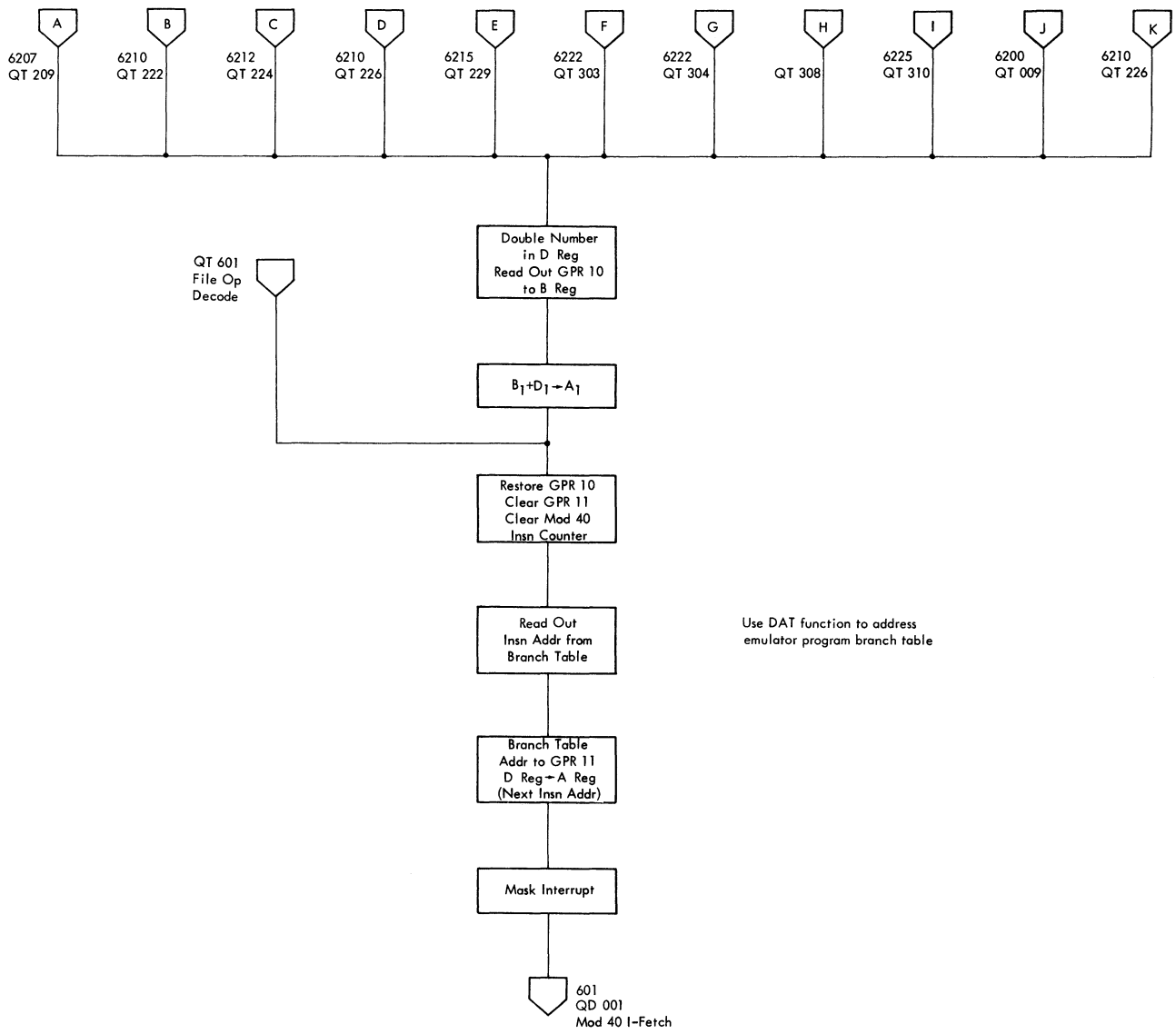
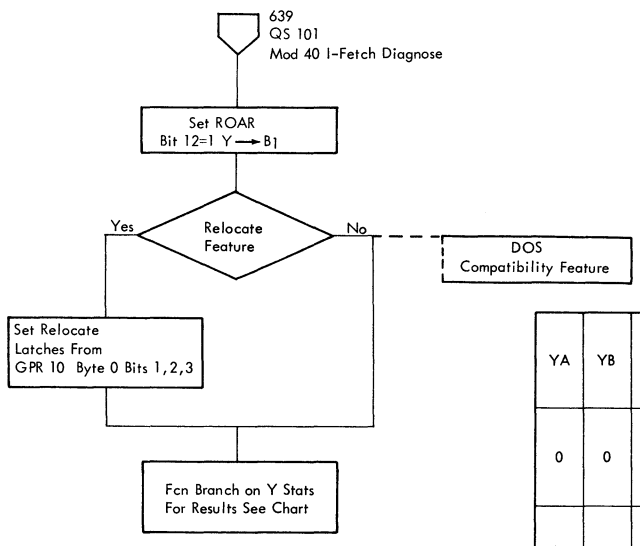


FIGURE 6220. 1401 EMULATOR PROGRAM ENTRY

QT 302



YA	YB	Microprogram	Diagram CAS Page	YA	YB	Microprogram	Diagram CAS Page
0	0	Multiway Branch	6205 QT 204	C	0	Transfer O-Reg to BAR	QT 304
1	0	Scatter-Gather	6207 QT 207	D	0	To 1401 I-Fetch	6200 QT 302
3	0	Clear 1401 Storage	6214 QT 221	E	0	Branch Satisfied Return to 1401 I-Fetch	6200 QT 302
4	0	Clear 1401 Locations 0-80	6214 QT 221	F	0	Tape Load Mode Insert GM in 1401 Storage	6218 QT 256
9	0	Increment Address	6206 QT 207	F	3	Tape Load Mode Insert TM and GM in 1401 Storage	6218 QT 256
9	8	Decrement Address	6206 QT 207	F	8	Tape Move Mode Insert GM in 1401 Storage	6218 QT 256
A	0	Read Column Binary	6225 QT 309	F	B	Tape Move Mode Insert TM and GM in 1401 Storage	6218 QT 256
B	0	Punch Column Binary	6225 QT 309				

●FIGURE 6221. 1401 DIAGNOSE

Character at d for B(I) d Branch			
d	Branch On	d	Branch On
b1	Unconditional	*	Inquiry Clear
9	Carr Chan #9	T	Low Compare B<A
A	"Last Card" Switch	U	High Compare B>A
B	Sense Switch B	Z	Overflow
C	Sense Switch C	?	Reader Error
D	Sense Switch D		
E	Sense Switch E	‡	Printer Error
F	Sense Switch F		
G	Sense Switch G	@	Carr Chan #12
H	Reader Busy	%	Processing Check with Process Check Switch Off
I	Punch Busy		
S	Equal Compare B = A*	/	Unequal Compare B ≠ A
P	Printer Busy	L	Tape Error
!	Punch Error	Q	Inquiry Request
J	Tape or I/O Busy	R	Printer Carriage Busy
K	End of Reel		

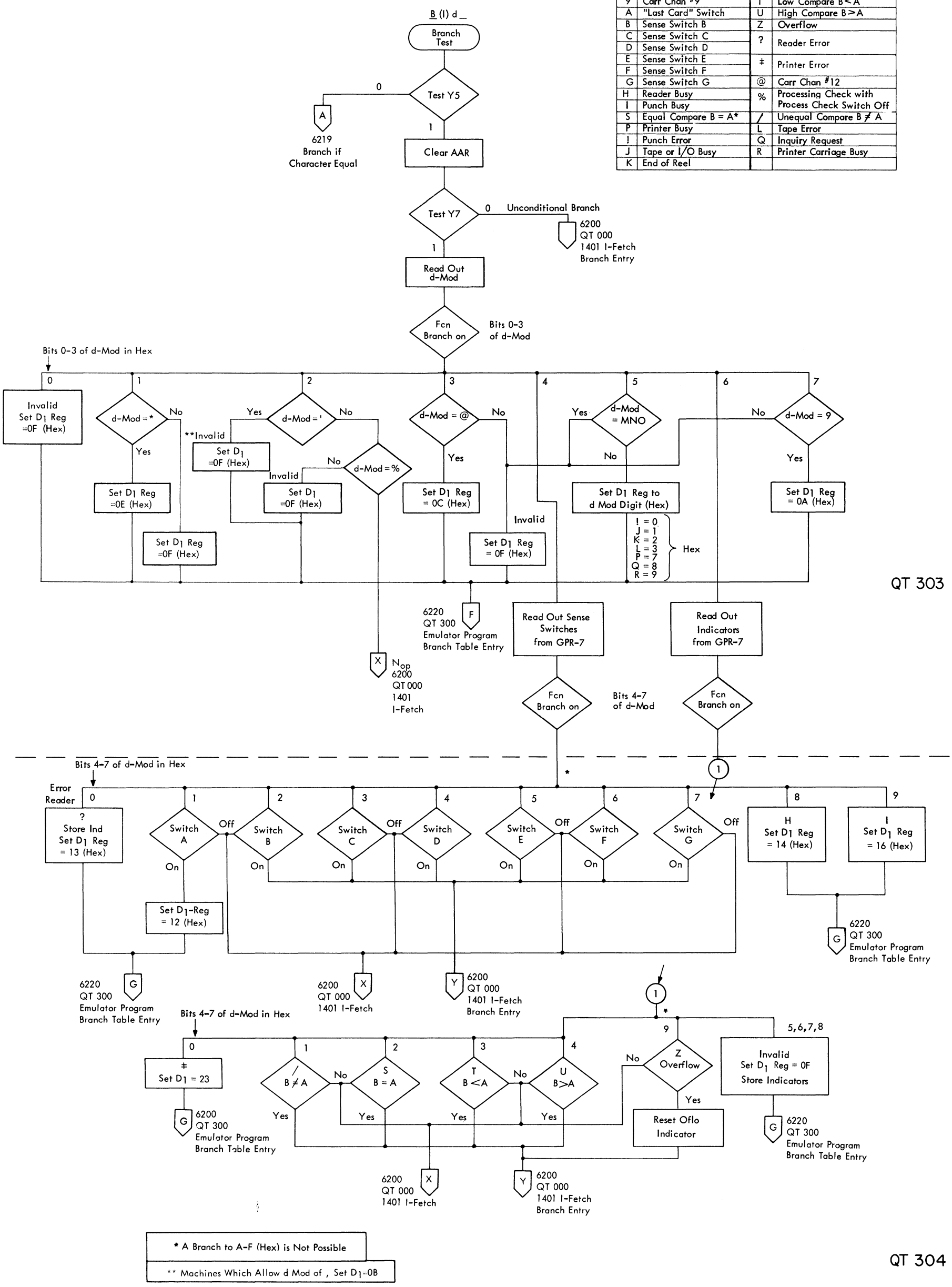


FIGURE 6222. 1401 BRANCH TESTS

QT 305

QT 306

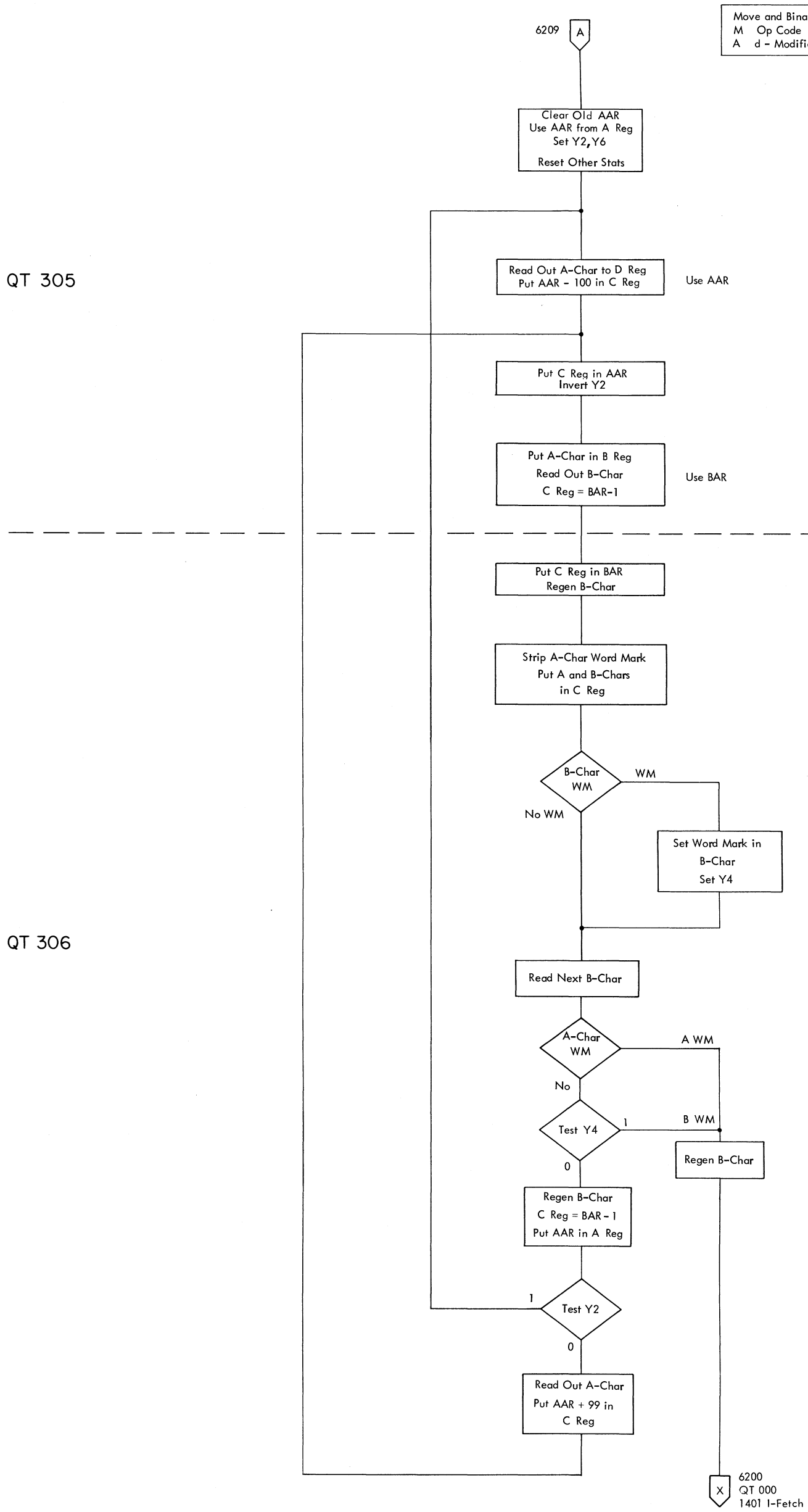


FIGURE 6223. 1401 MOVE AND BINARY DECODE

QT 305

QT 306

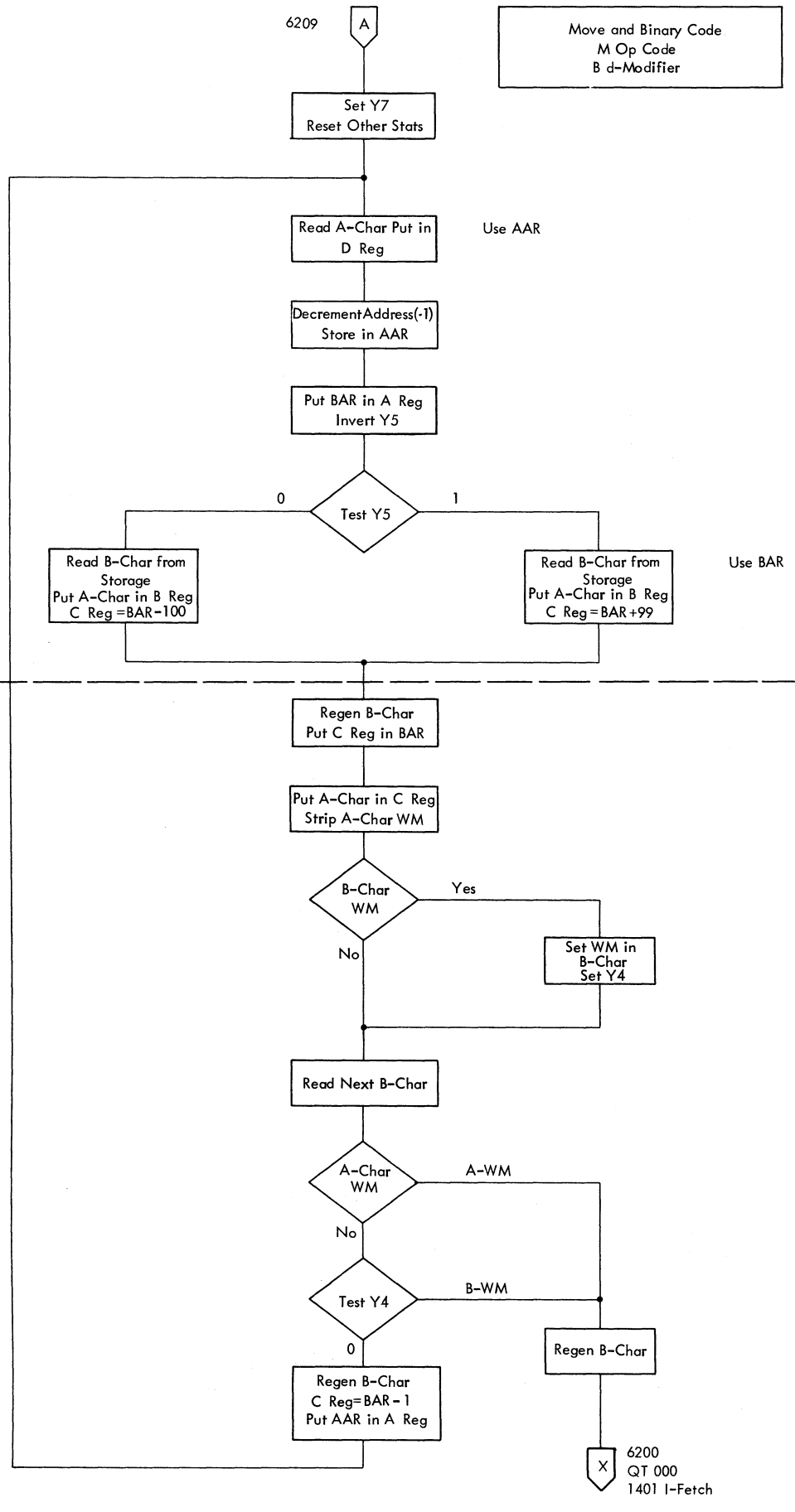


FIGURE 6224. 1401 MOVE AND BINARY CODE

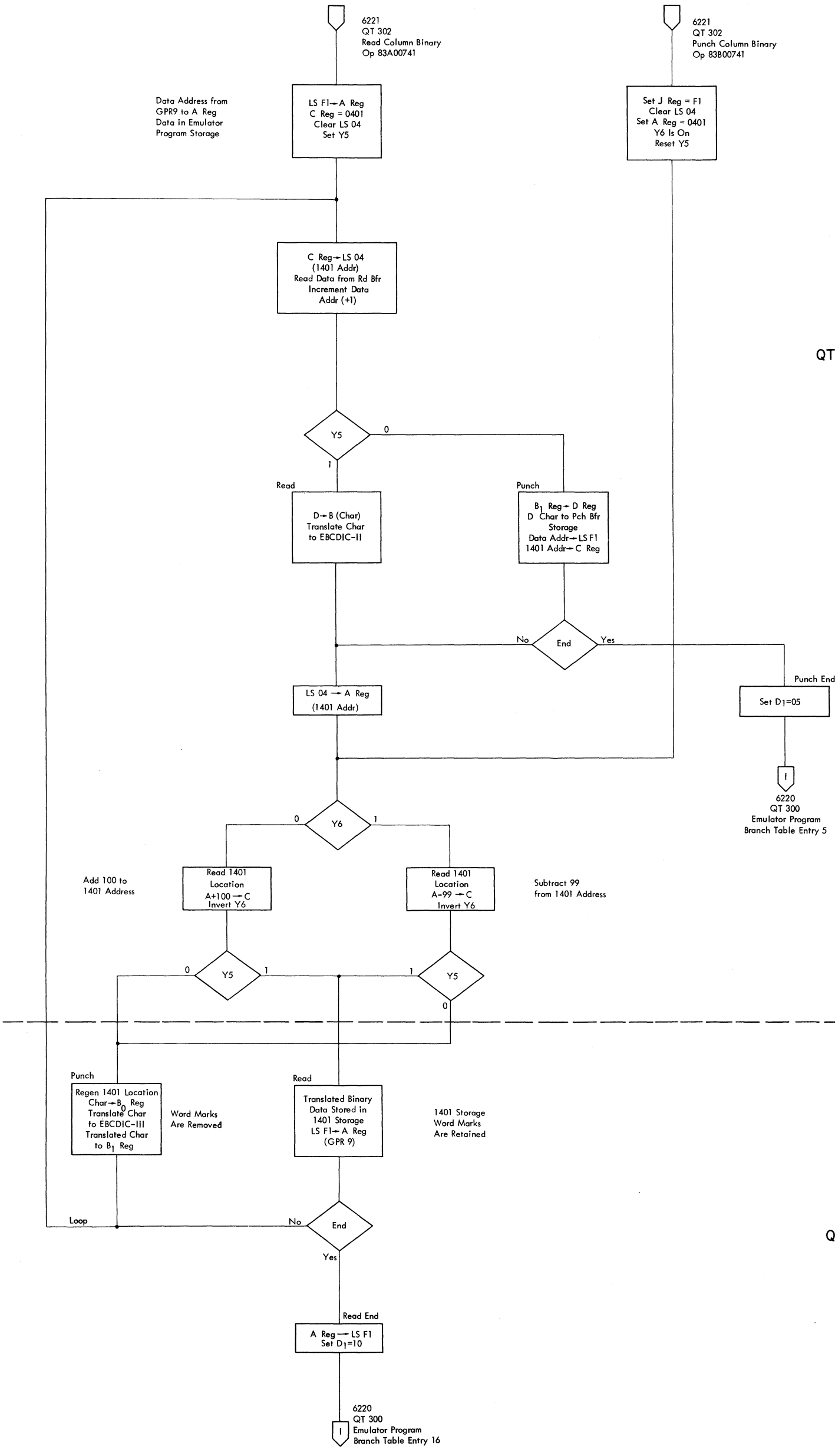
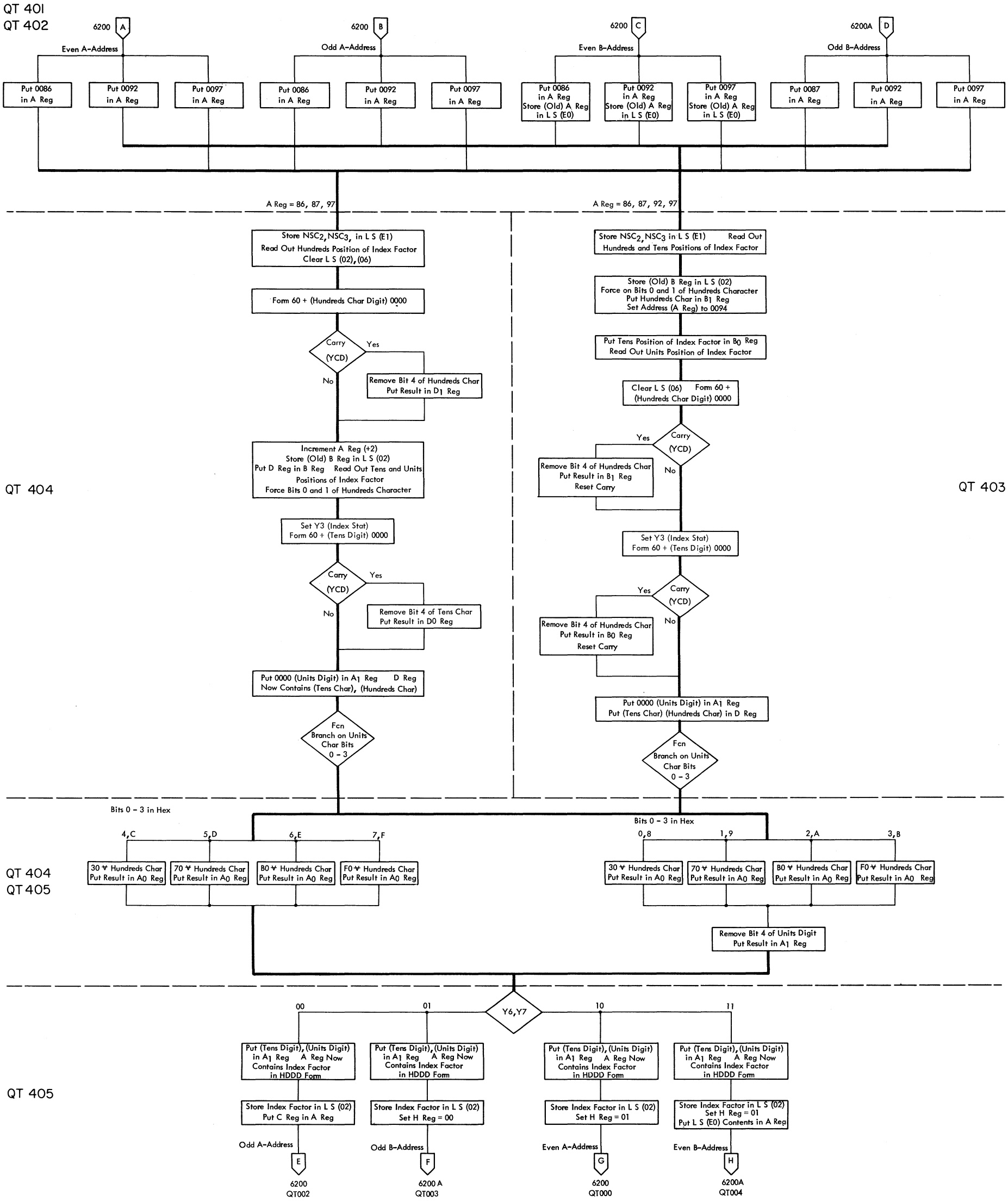


FIGURE 6225. 1401 READ AND PUNCH COLUMN BINARY

QT 401
QT 402



QT 404

QT 403

QT 404
QT 405

QT 405

FIGURE 6226. 1401 INDEX FACTOR FETCH

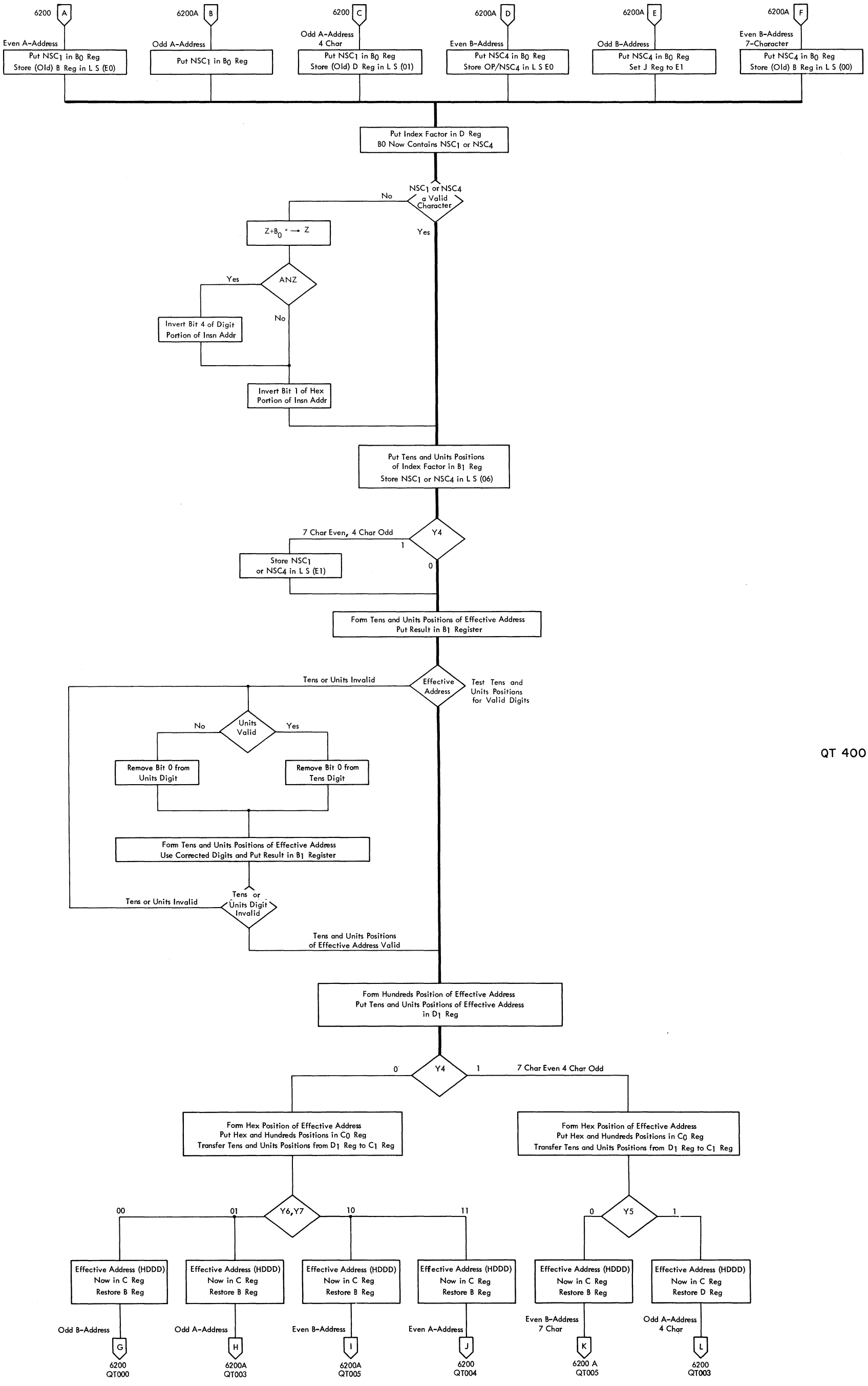


FIGURE 6227. 1401 INDEX ADD

Bits 0-3 * ↓ / Bits 4-7 →	XXXX0000	XXXX0001	XXXX0010	XXXX0011	XXXX0100	XXXX0101	XXXX0110	XXXX0111	XXXX1000	XXXX1001	XXXX1010	XXXX1011	XXXX1100	XXXX1101	XXXX1110	XXXX1111
1000XXXX												Halt QU506	¤ Clear WM QU271 6314			
1001XXXX												$ Store and Restore Status QU261 6313				
1010XXXX												, Set WM QU271 6314	% Divide QU311 6316/6322			
1011XXXX													@ Multiply QU311 6316/6320			
1100XXXX	? Zero and Add QU401** 6324***	A Add QU311 6316	B Branch if Char Equal QU221 6311	C Compare QU401 6331	D Data Move QU401 6324	E Move Char and Edit QU508 6341	F Forms Control QU507 6339	G Store Addr Reg QU251 6312								
1101XXXX	! Zero and Subtract QU401 6324	J Branch if Int Ind On QU211 6310	K Stkr Select and Feed QU507 6339	L Rd or Wr with WM QU501 6334	M Rd or Wr without WM QU501 6334	N No Op QU506 6306	Invalid Operation Codes QU506			R Branch if I/O Ind On QU201 6309						
1110XXXX	Invalid Op QU506	/ Clear Storage QU281 6314	S Subtract QU311 6316	T Table Lookup QU291 6315	U Unit Control QU501 6334	V Branch on WM/Zone QU221 6311	W Branch if Bit Equal QU221 6311	X Branch if I/O Ind On QU201 6309	Y Priority Test QU211 6310	Z Move Char and Sup Zero QU508						
1111XXXX																

Blank positions in matrix are invalid op codes; branch to QU506

* Bits 0-3 = 0000-0111 are invalid

** CLD page for start of microprogram

*** Figure number of maintenance diagram

FIGURE 6300. 1410/7010 OPERATION CODES IN EBCDIC II

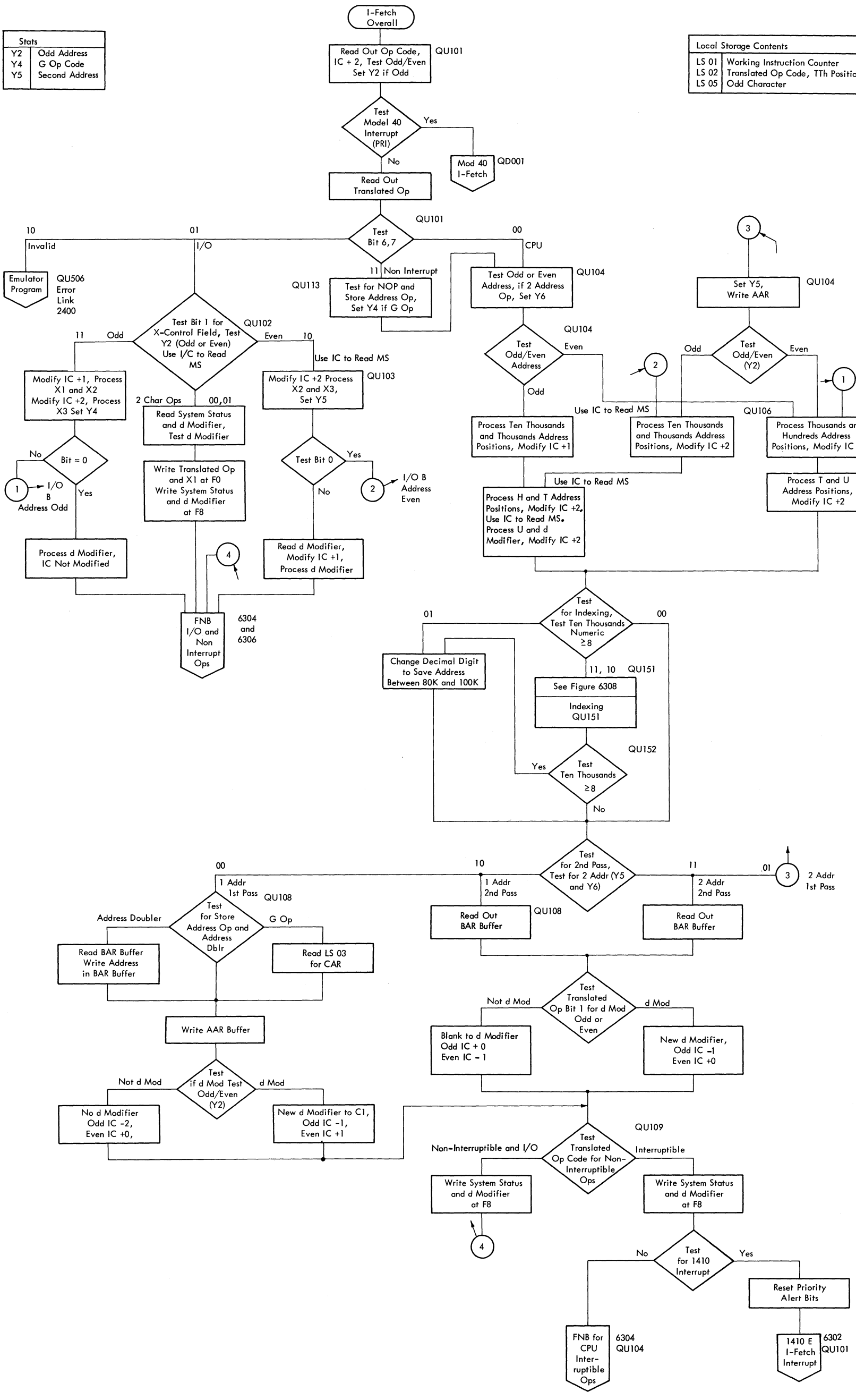


FIGURE 6301. 1410 E INSTRUCTION-FETCH OVERALL

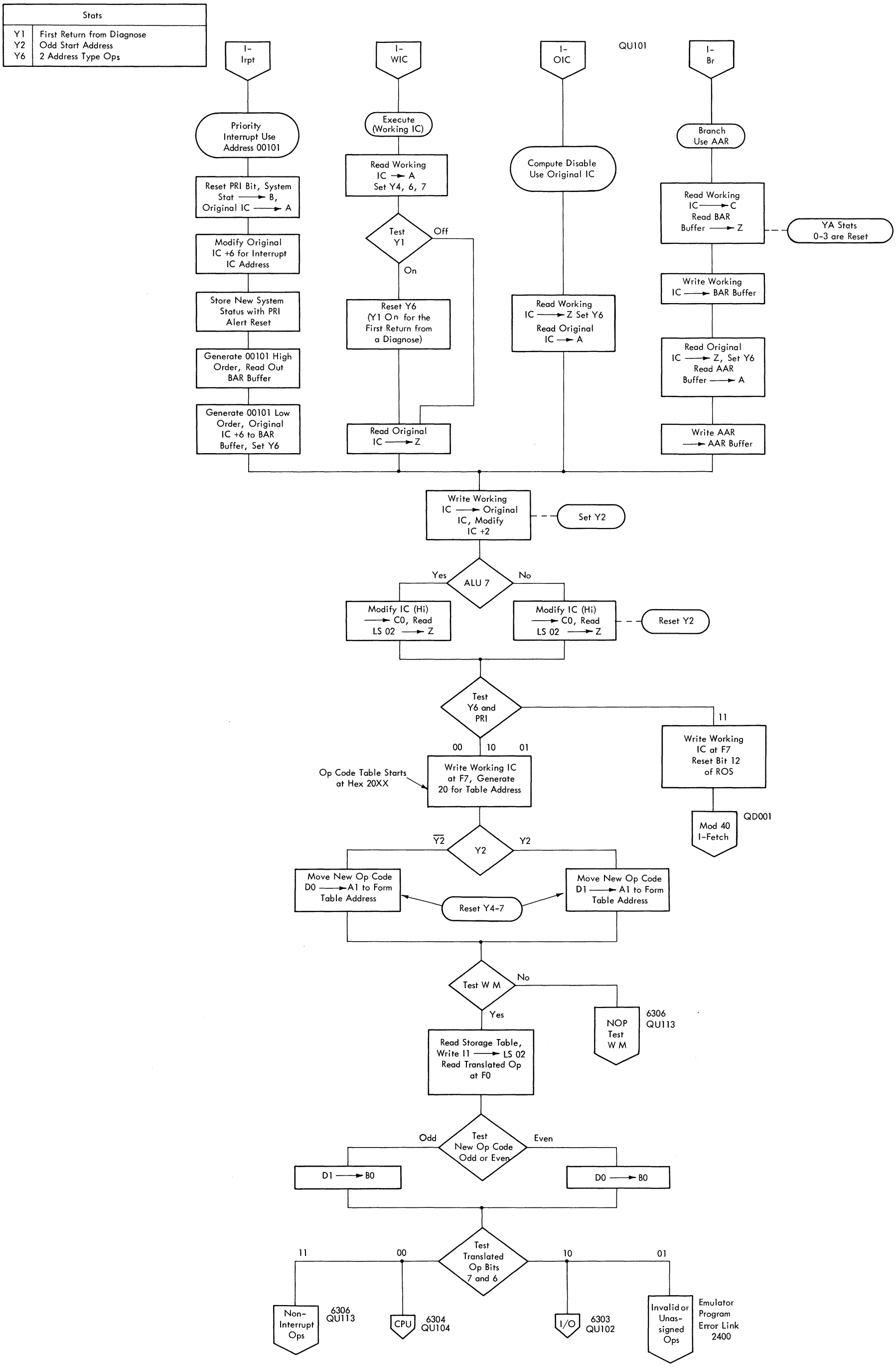


FIGURE 6302. 1410 E INSTRUCTION-FETCH START

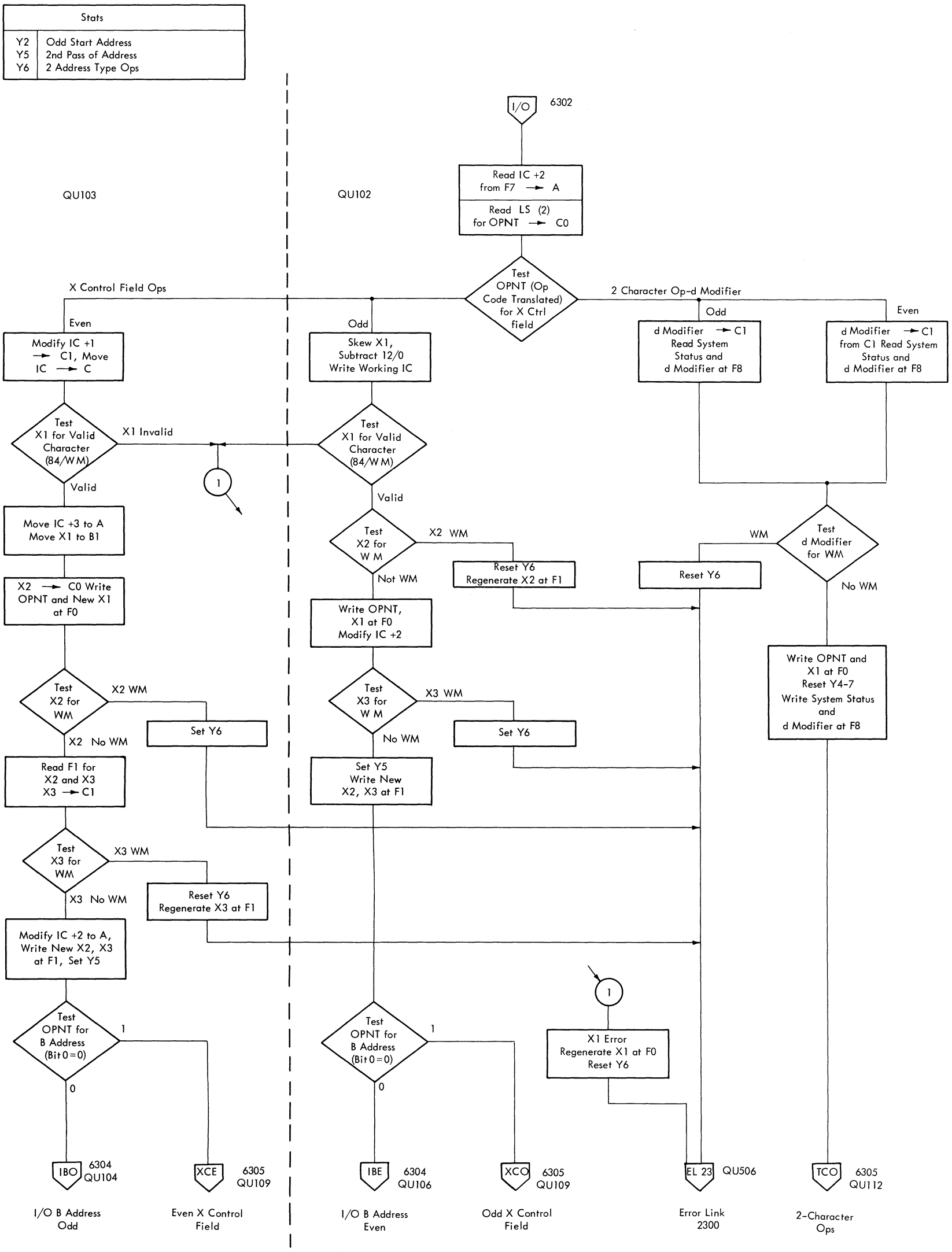
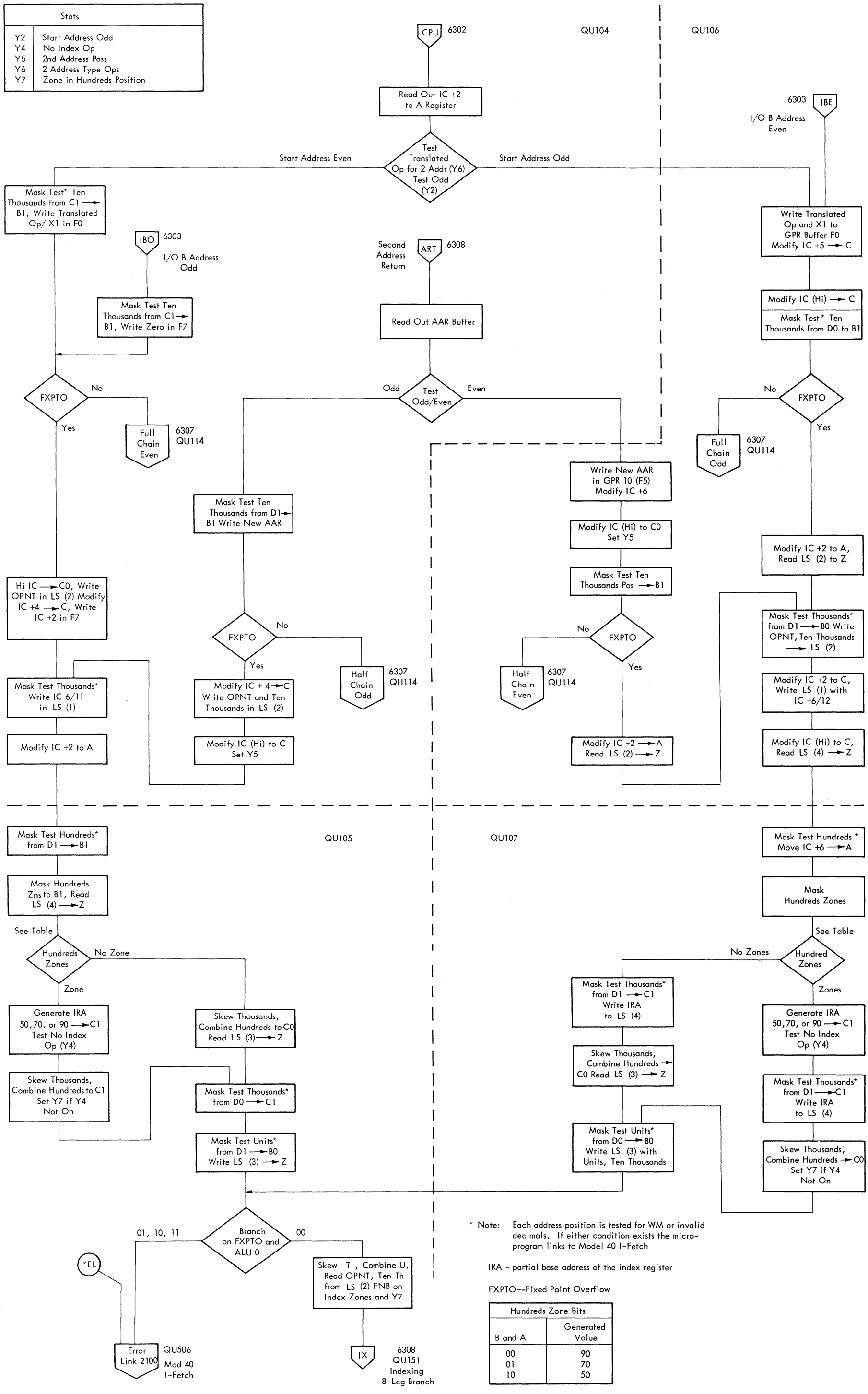


FIGURE 6303. 1410 E X CONTROL FIELD READOUT



* Note: Each address position is tested for WM or invalid decimals. If either condition exists the micro-program links to Model 40 I-Fetch

IRA - partial base address of the index register

FXPTO--Fixed Point Overflow

Hundreds Zone Bits	
B and A	Generated Value
00	90
01	70
10	50

FIGURE 6304. 1410 E ADDRESS READOUT

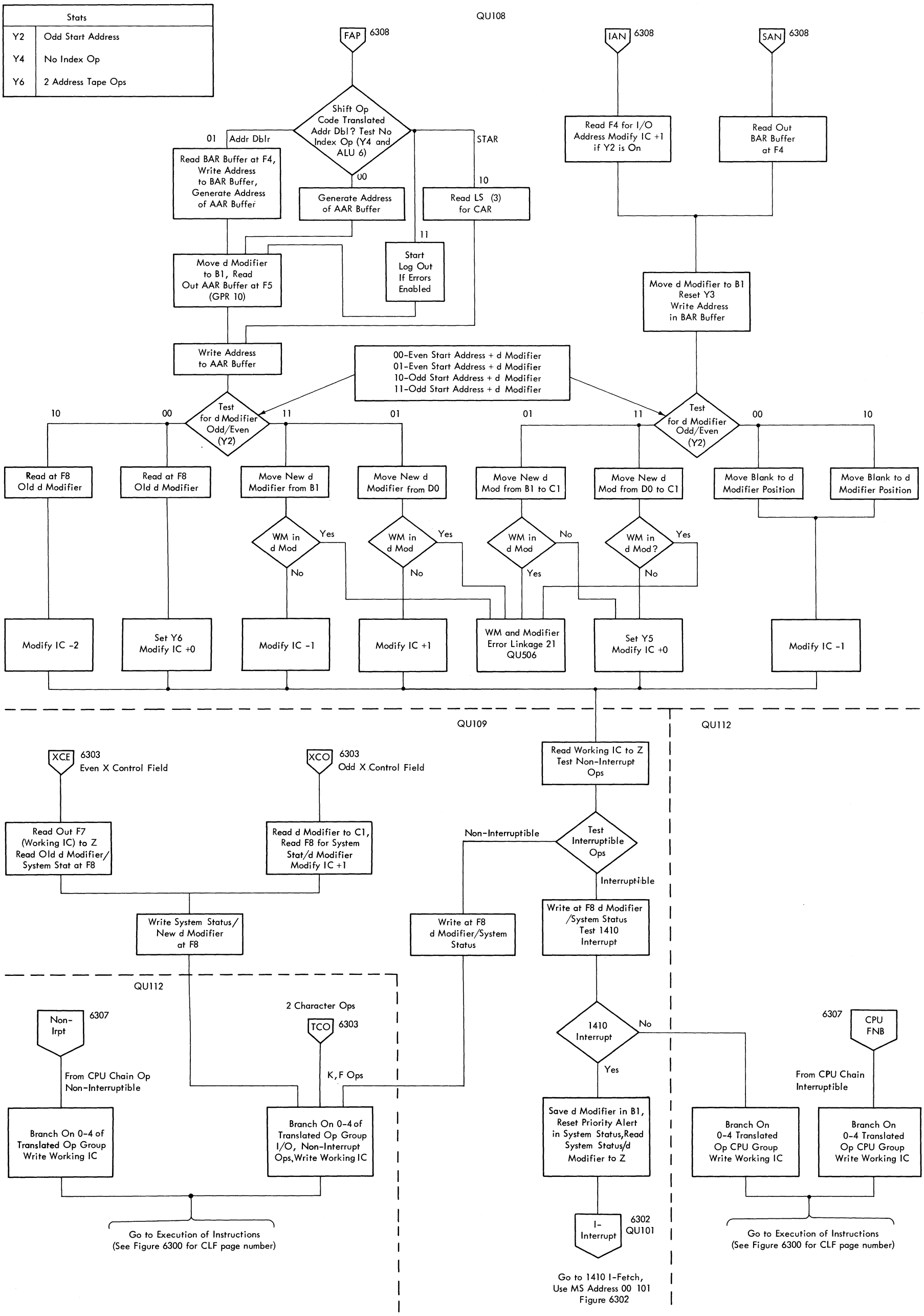


FIGURE 6305. 1410 E INSTRUCTION FETCH ENDING

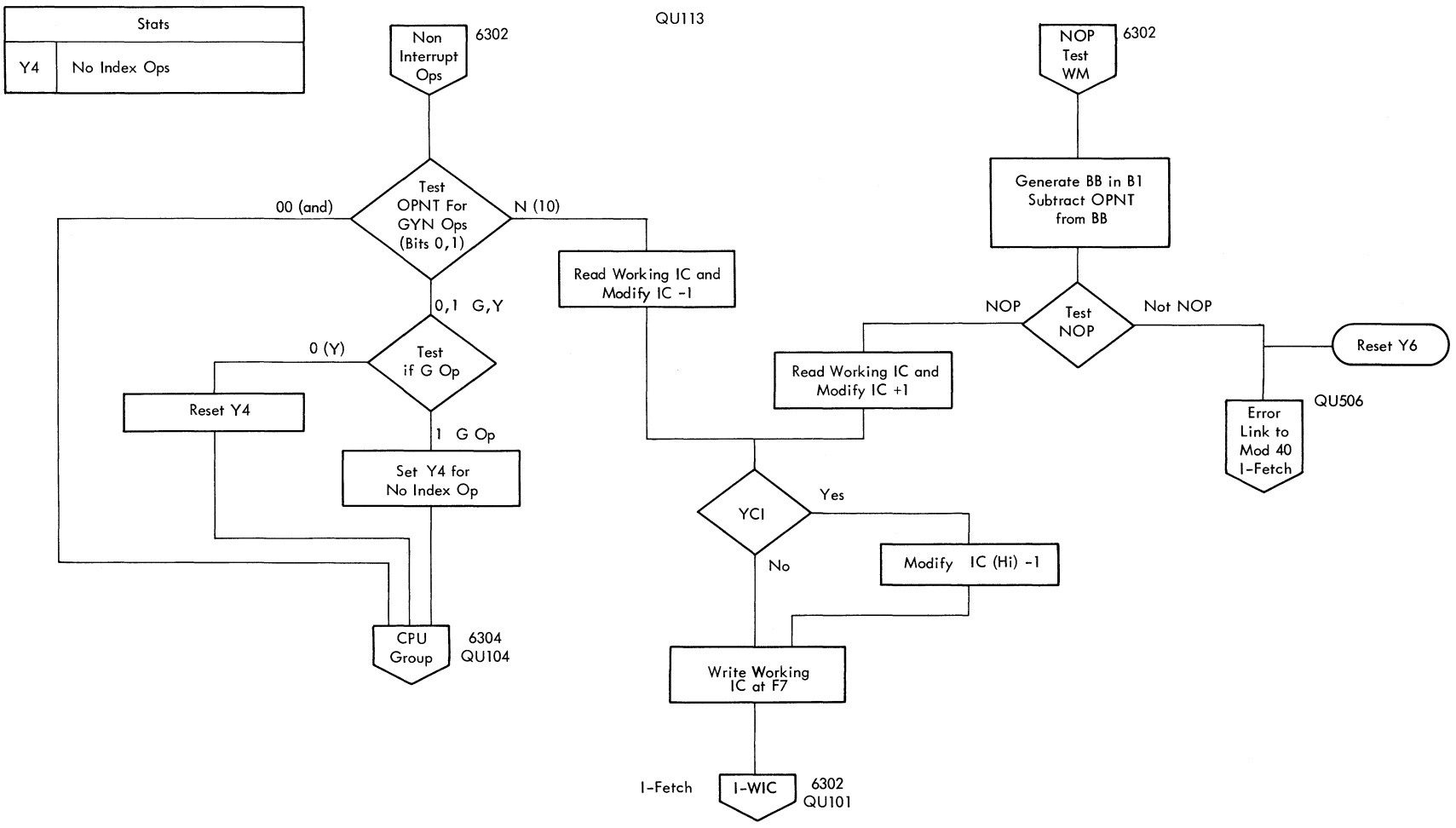


FIGURE 6306. 1410 E NOP AND NON-INTERRUPTIBLE OP CODES

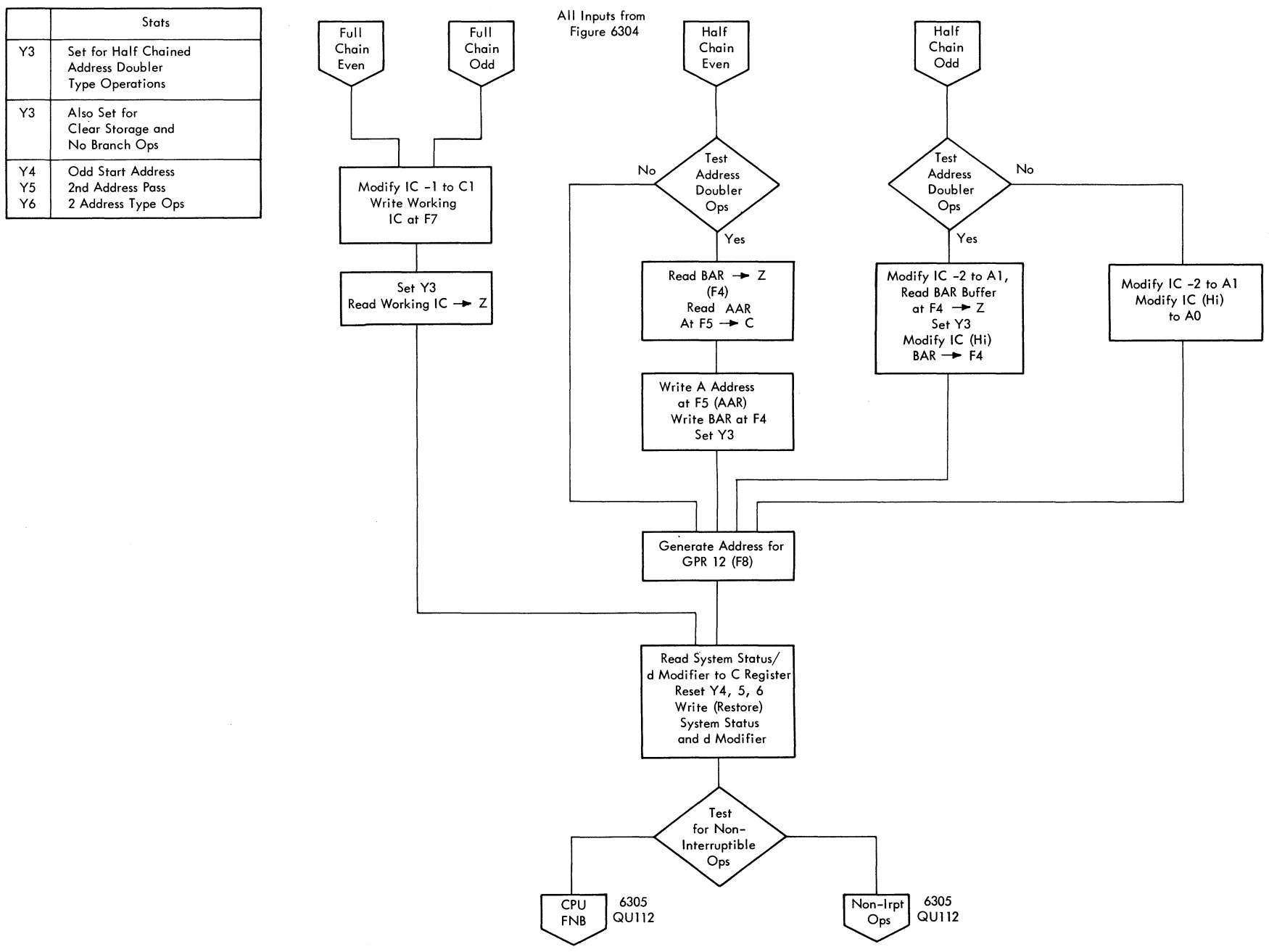


FIGURE 6307. 1410 E CHAINING

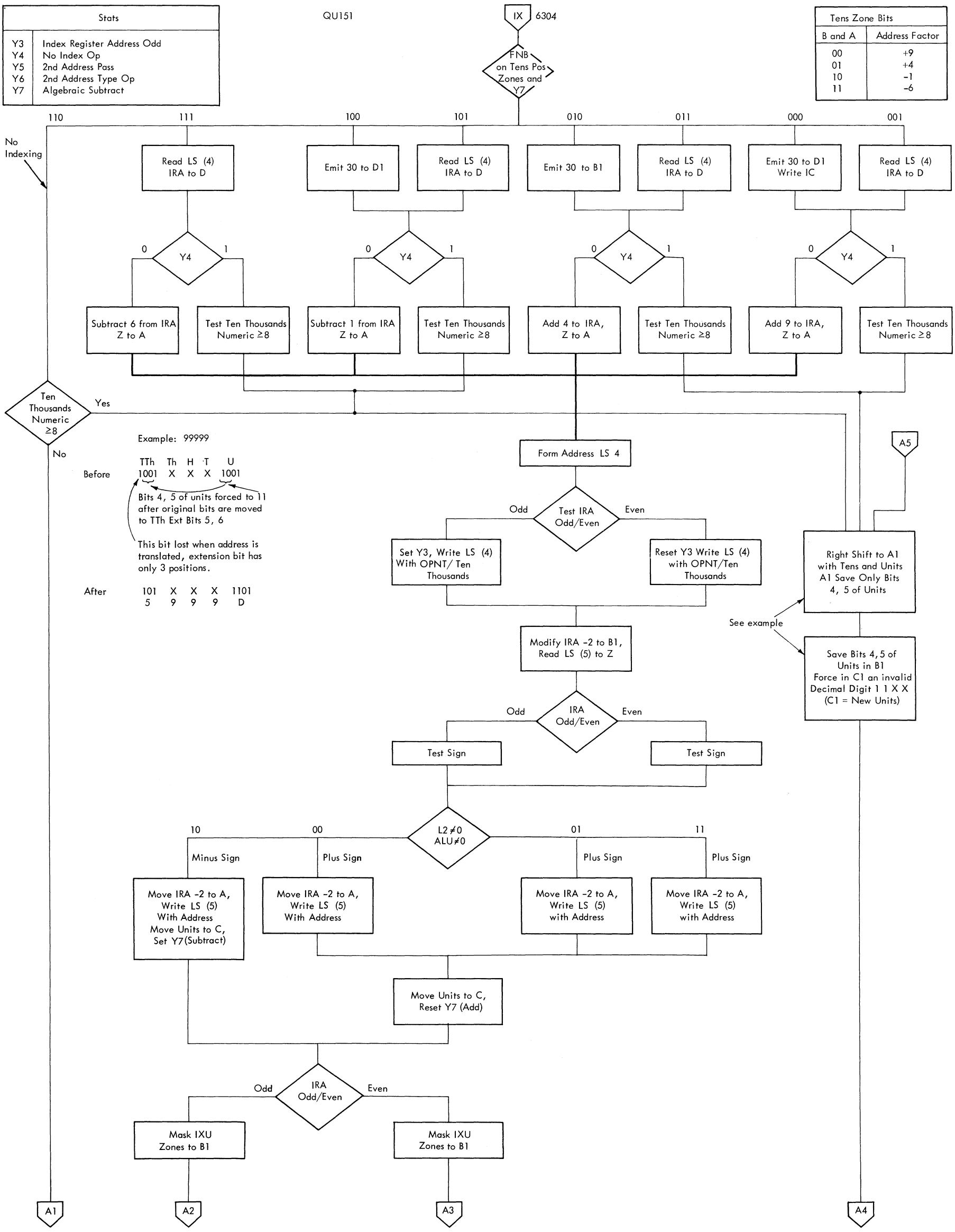


FIGURE 6308. 1410 E INDEXING (SHEET 1 OF 2)

Stats	
Y3	Index Register Address Odd
Y4	No Index Op
Y5	2nd Address Pass
Y6	2nd Address Op
Y7	Algebraic Subtract

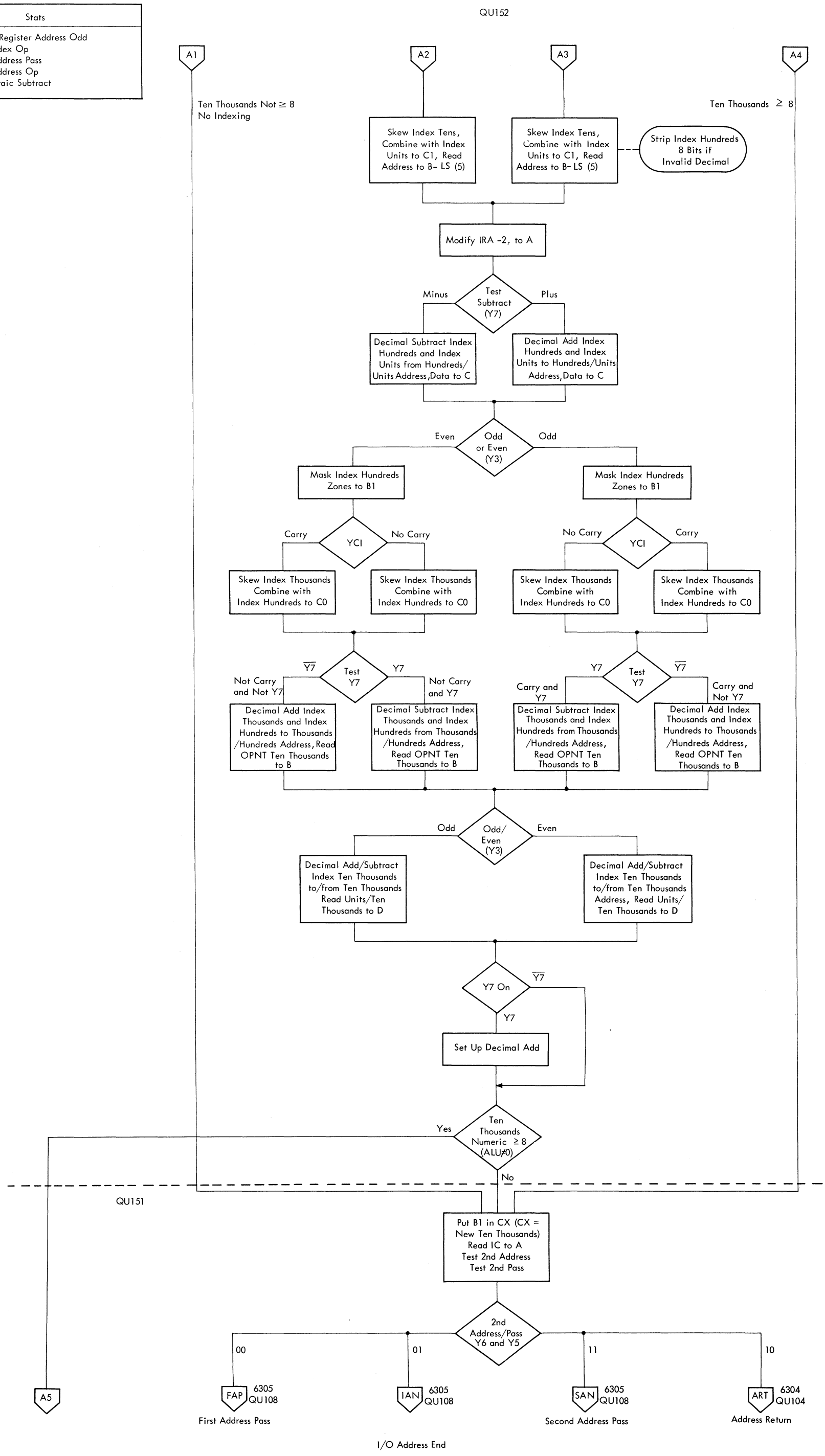


FIGURE 6308. 1410 E INDEXING (SHEET 2 OF 2)

Stats	
Y6	Branch Ch 2 - X Op

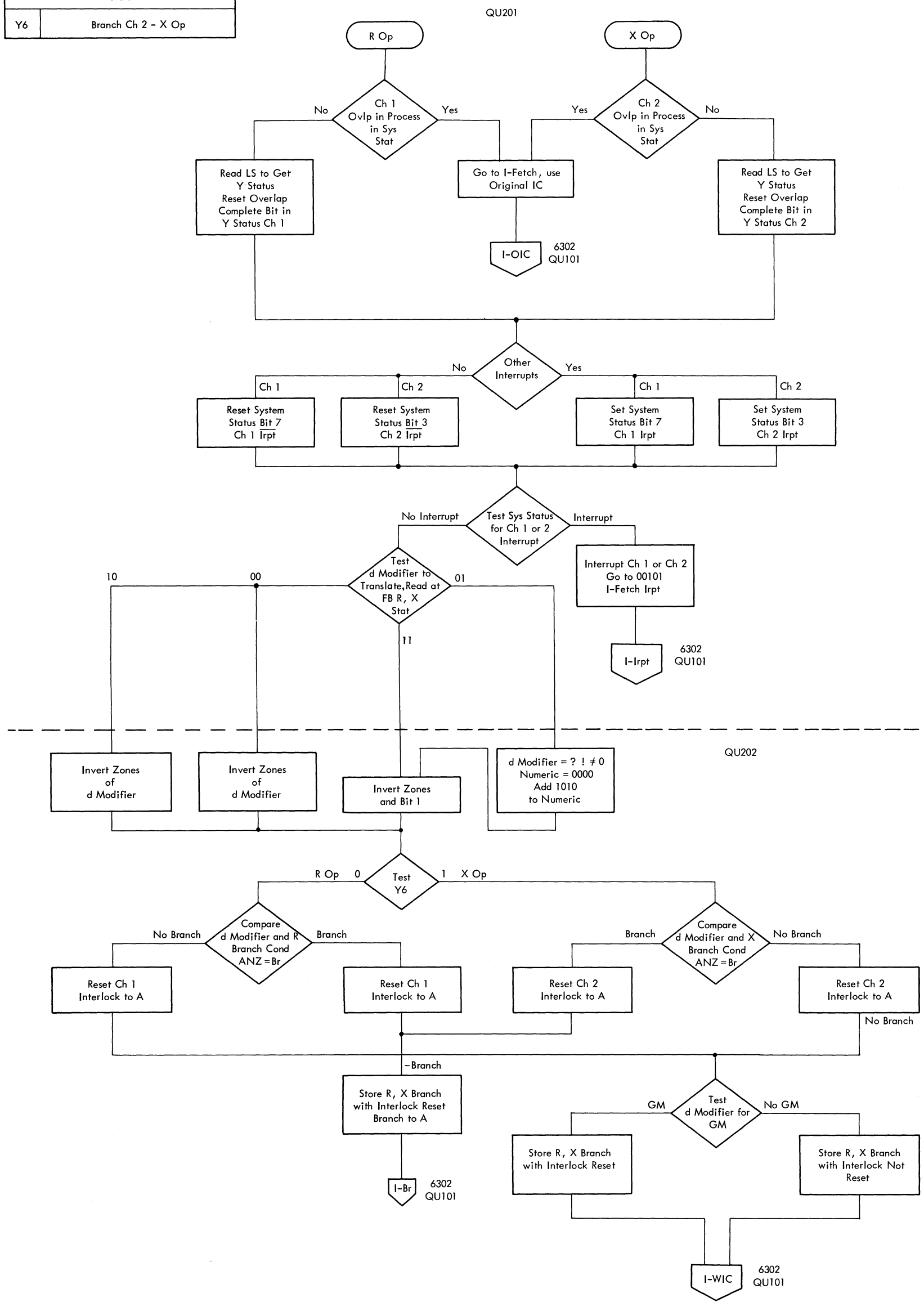


FIGURE 6309. 1410E BRANCH ON CHANNEL STATUS

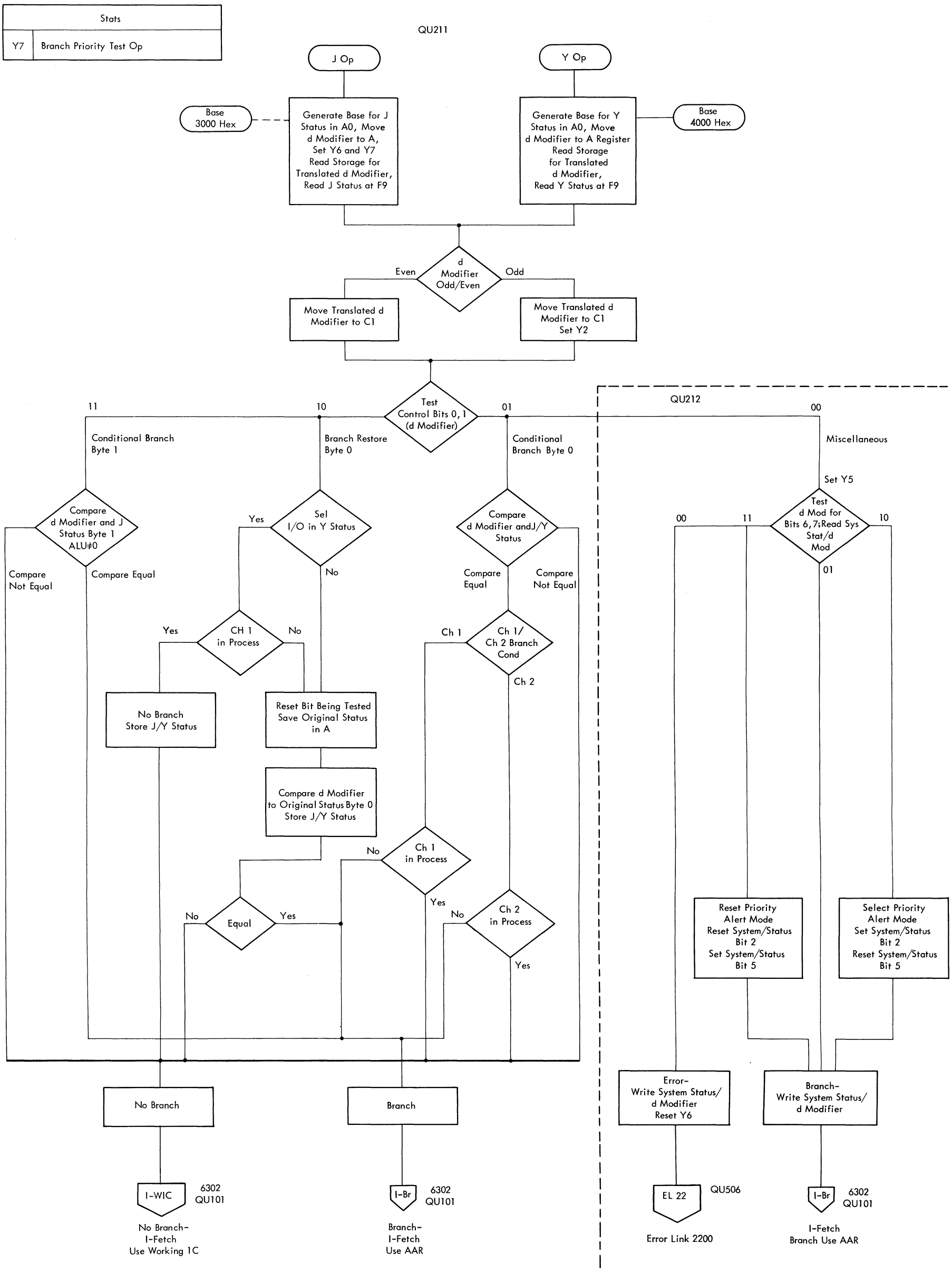


FIGURE 6310. 1410 E PRIORITY TEST AND BRANCH, BRANCH ON INTERNAL INDICATOR

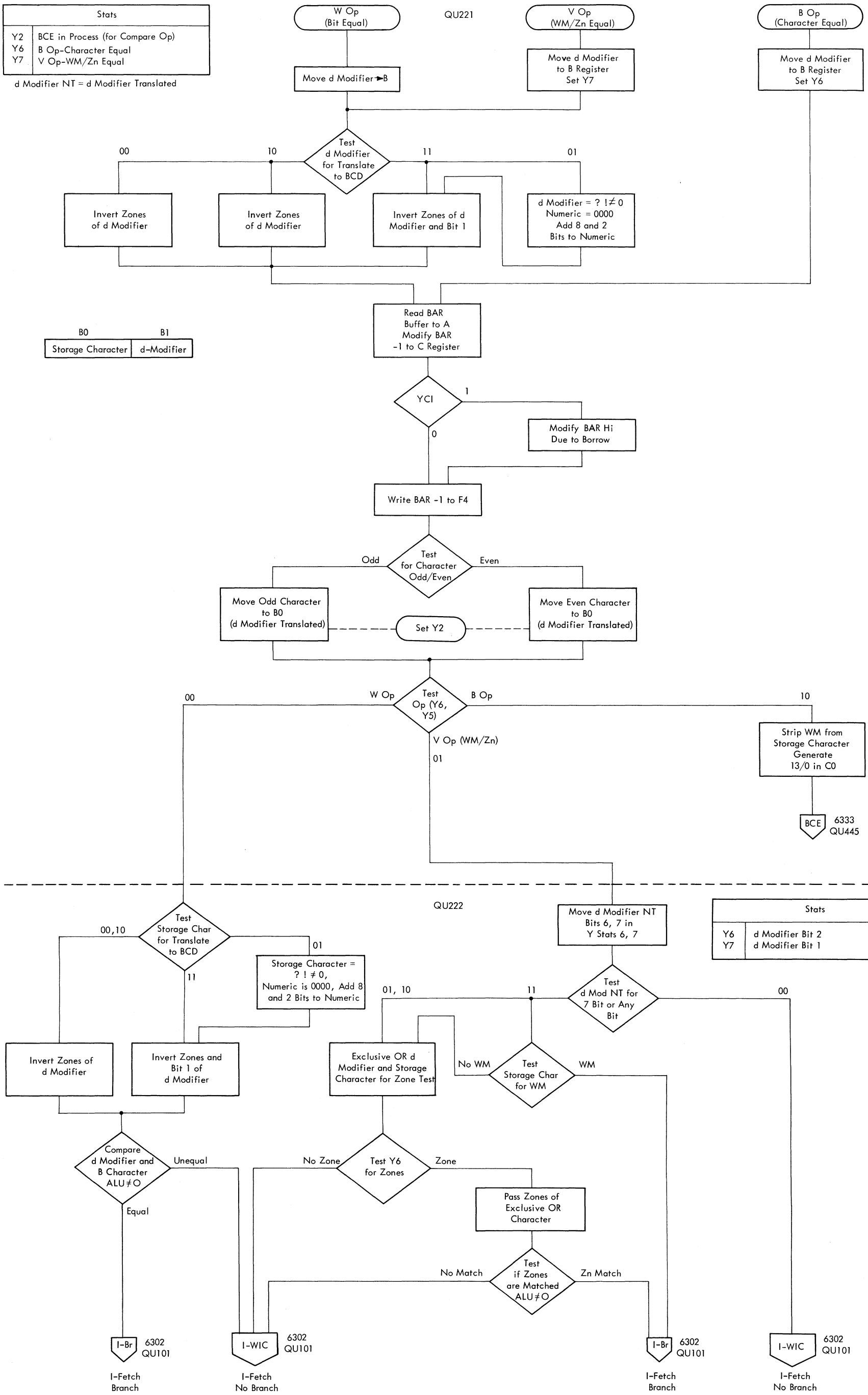


FIGURE 6311. 1410 E BRANCH IF: CHARACTER, WM/ZONE, OR BIT EQUAL

Stats	
Y6	Second Address in Loop
Y7	C Address Odd at Start
LS 03	C Address Register
O/H	Original Hundreds Position
O/T	Original Tens Position

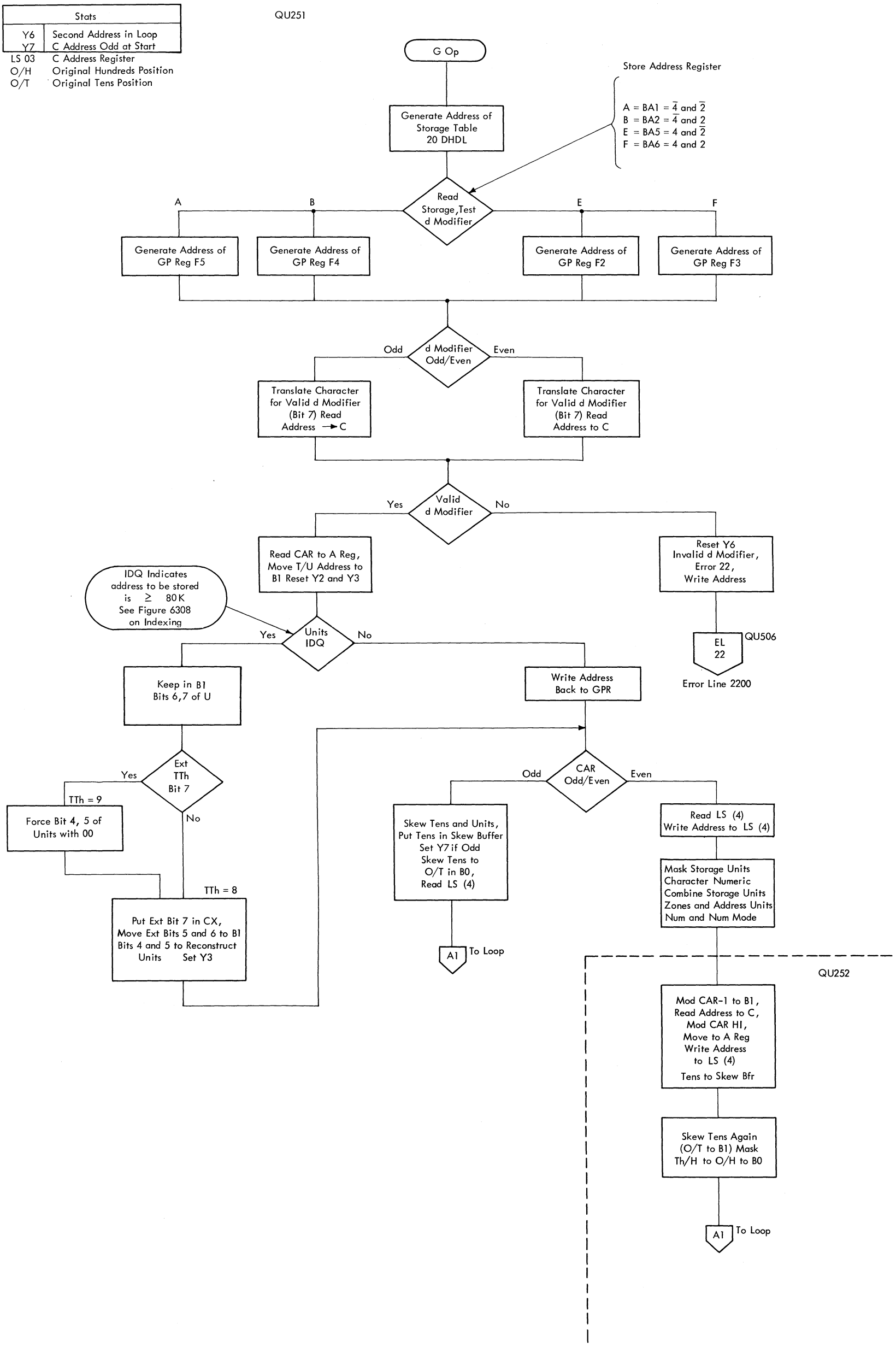


FIGURE 6312. 1410E STORE ADDRESS REGISTER (SHEET 1 OF 2)

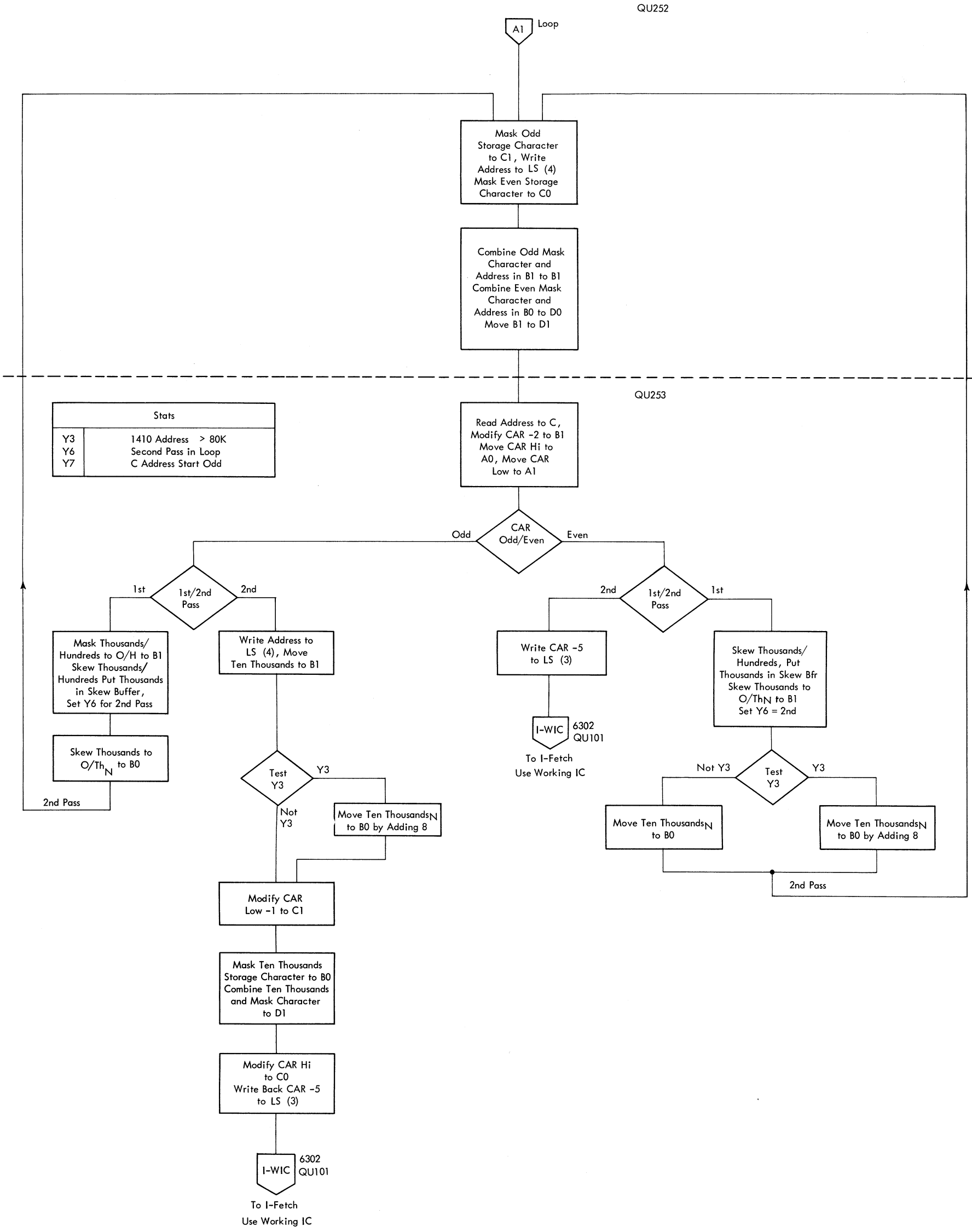


FIGURE 6312. 1410E STORE ADDRESS REGISTER (SHEET 2 OF 2)

Translated d Modifier - Stats	
Bit 4 Bit 5 Bit 6	Internal or Ch 2 - Set Y6 I/O Status - Set Y7 Store Status

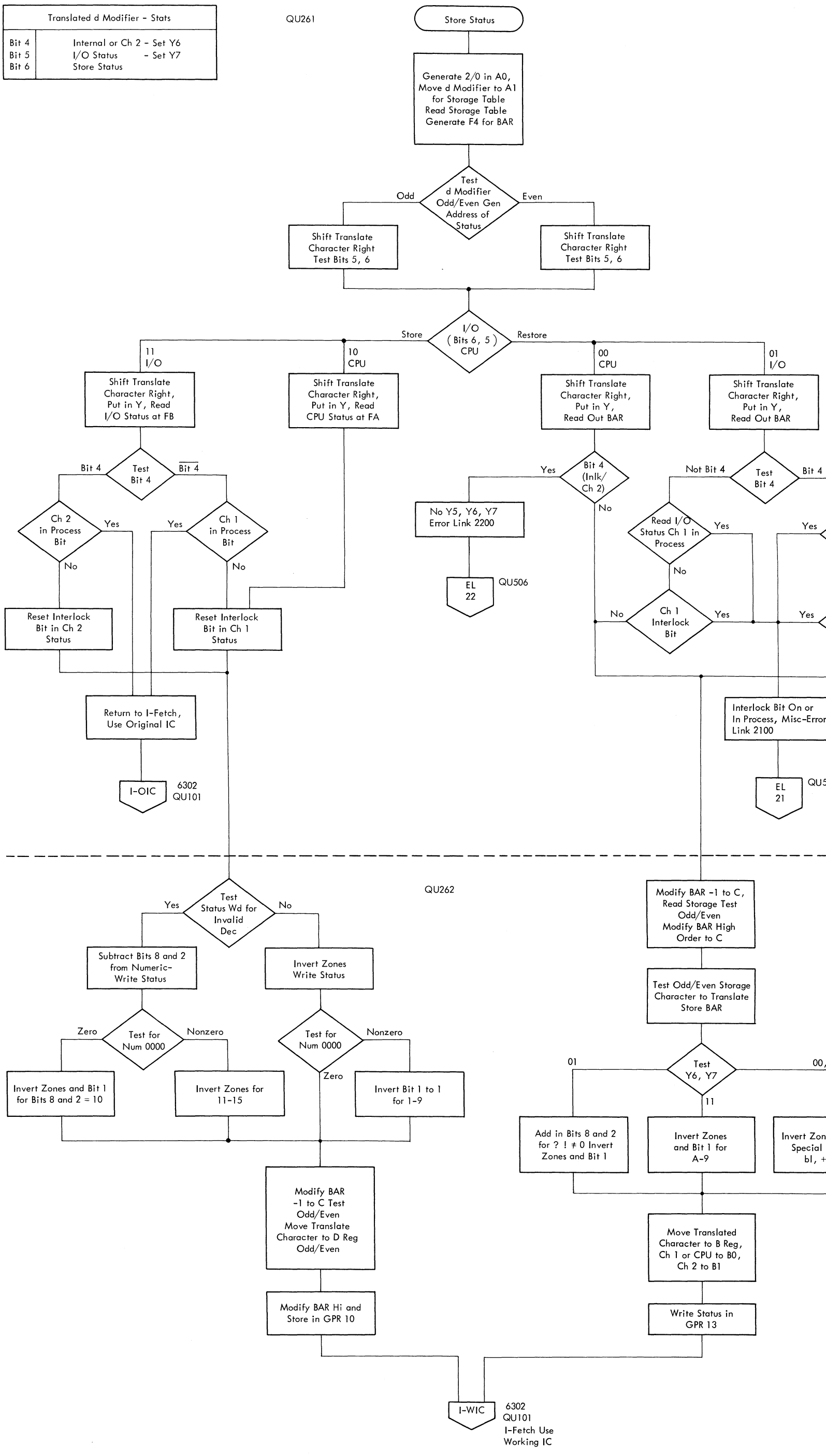
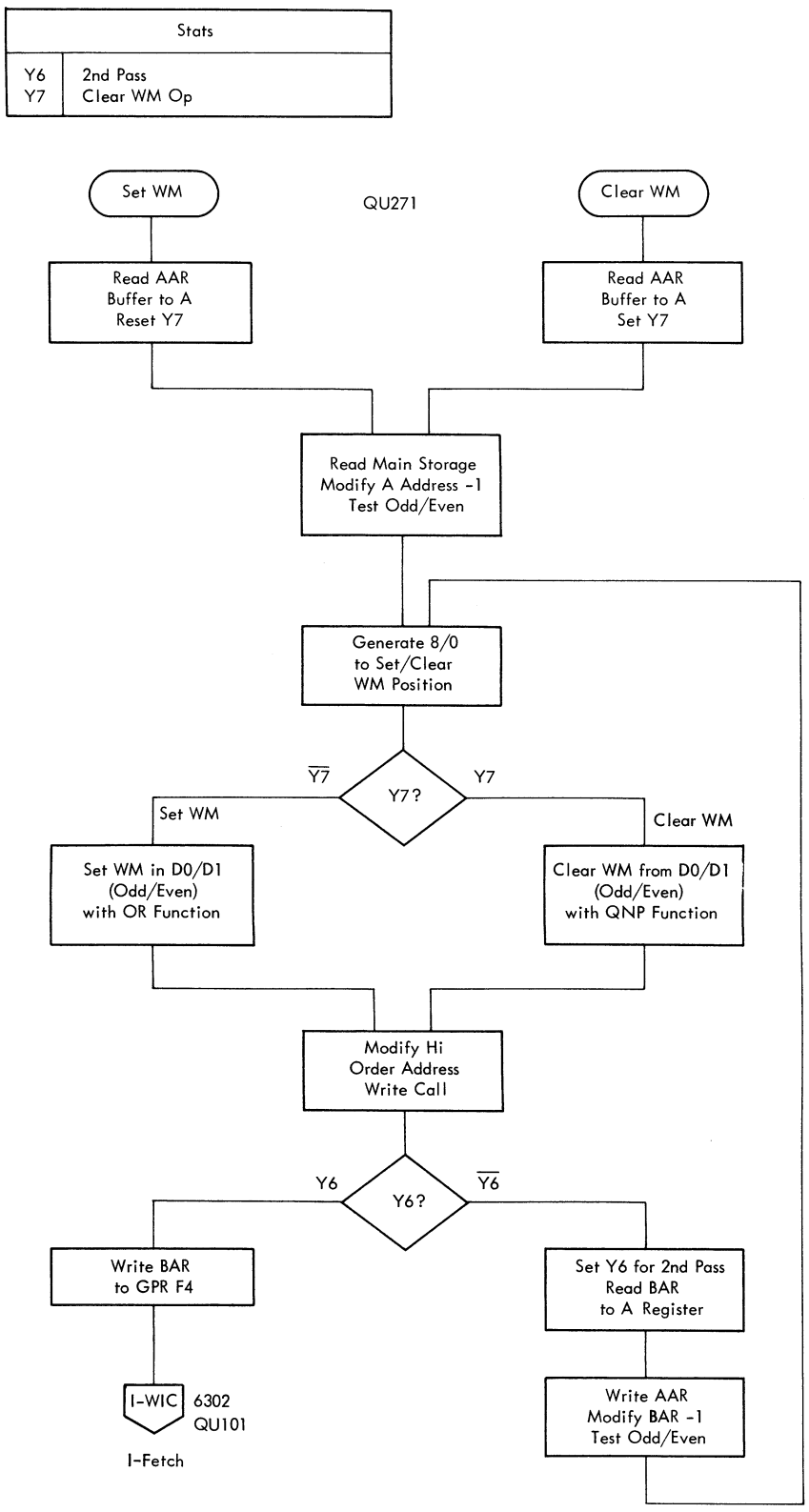
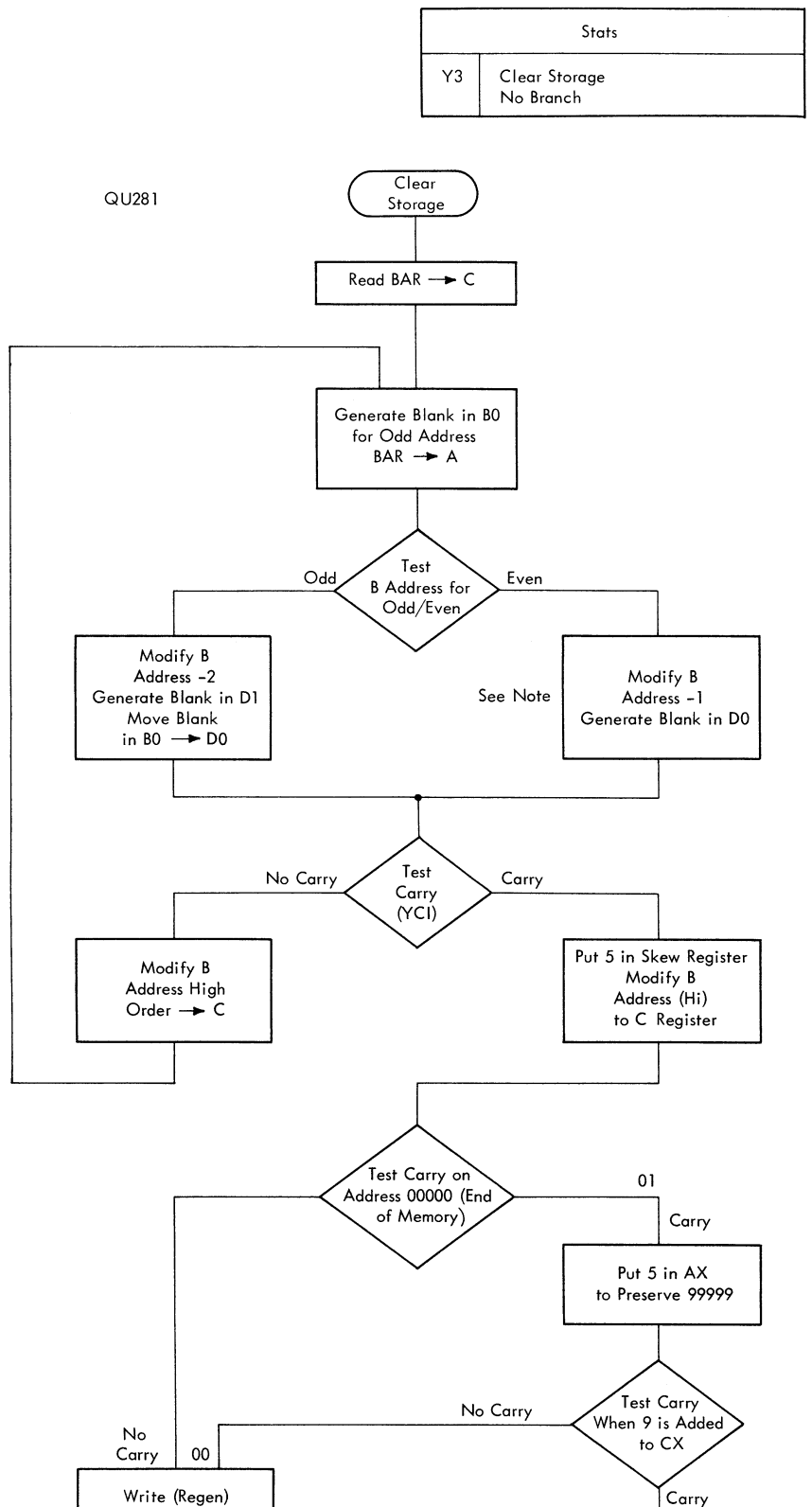


FIGURE 6313. 7010E STORE AND RESTORE STATUS





Note: Clearing storage across the 00000 storage boundary results in a 1410 address of 99,999. Because the 2040 data registers can emulate only those 1410 addresses ≦ 79,999, addresses ≧ 80,000 must be converted. See example on Figure 6308.

FIGURE 6314. 1410 E SET/CLEAR WM AND CLEAR STORAGE

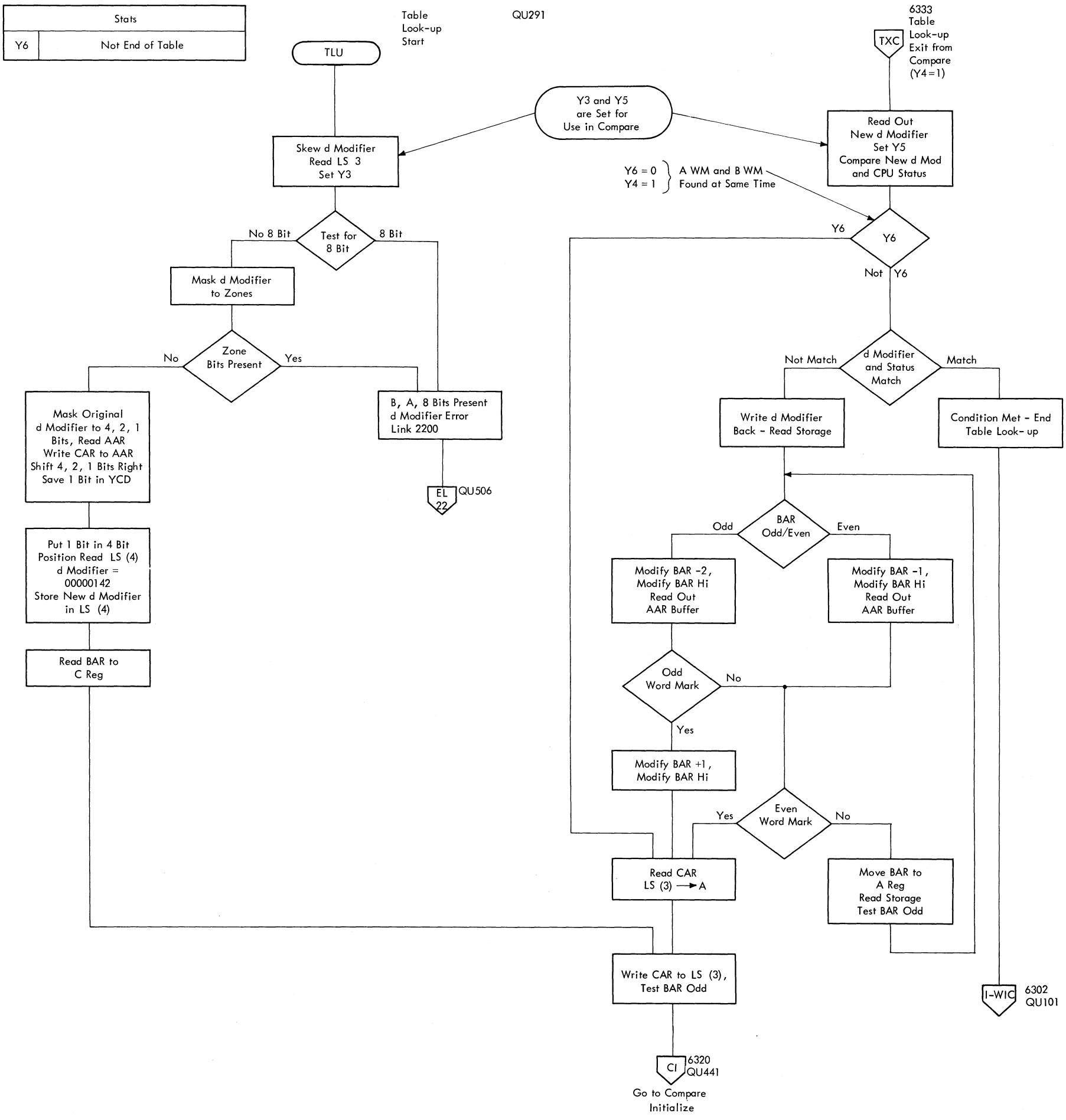


FIGURE 6315. 1410E TABLE LOOKUP

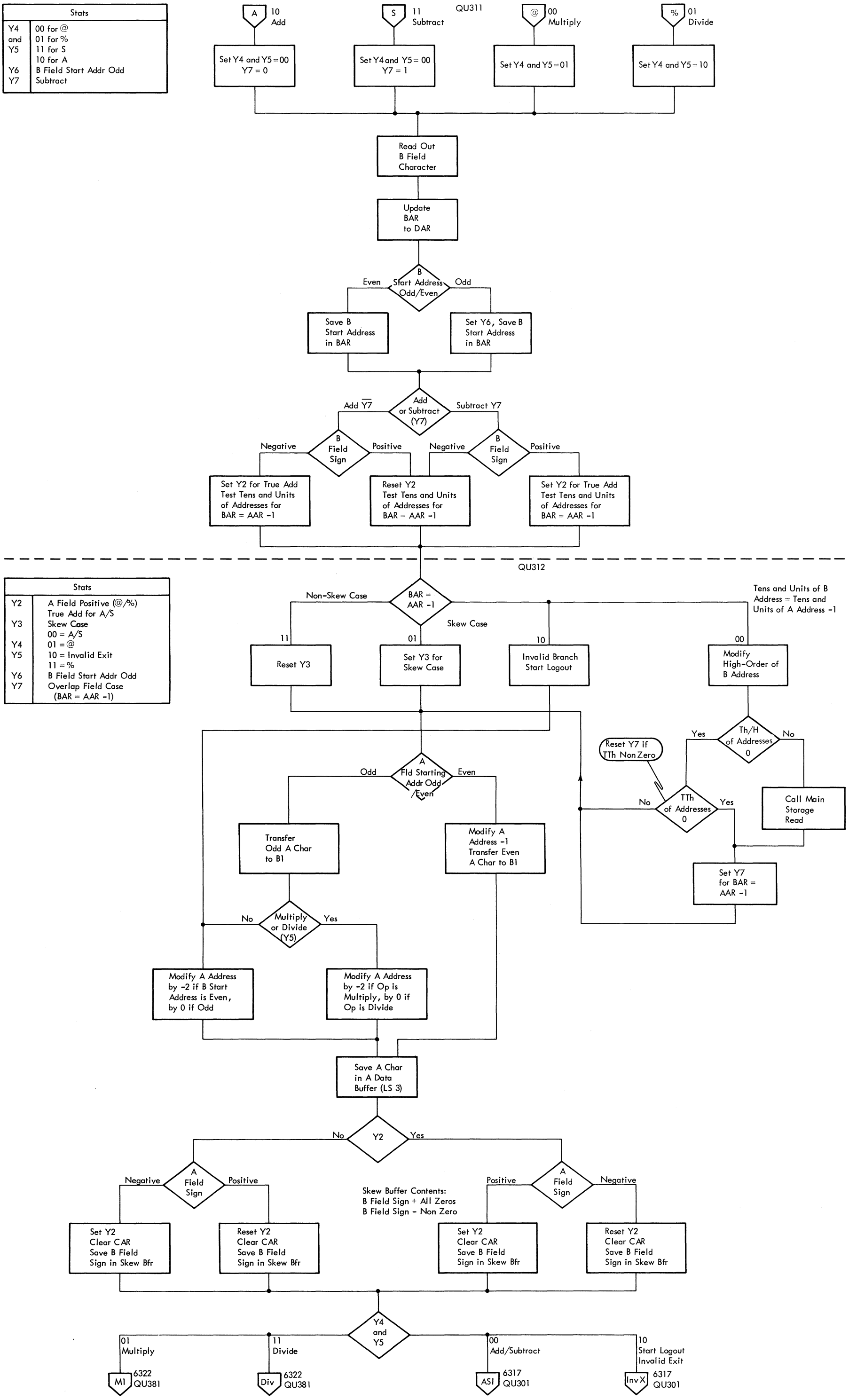


FIGURE 6316. 1410E ADD, SUBTRACT, MULTIPLY, DIVIDE

Stats	
Y2	True Add Case
Y3	Skew Case
Y5	Numeric Result Non-Zero
Y6	B Field Start Address Odd
Y7	BAR = AAR -1 Case (Ovlp)

Skew Case - A field starting address even, B field starting address odd or vice versa (AO/BE).

Overlap Case - Data fields overlap (BAR = AAR -1); requires special loop.

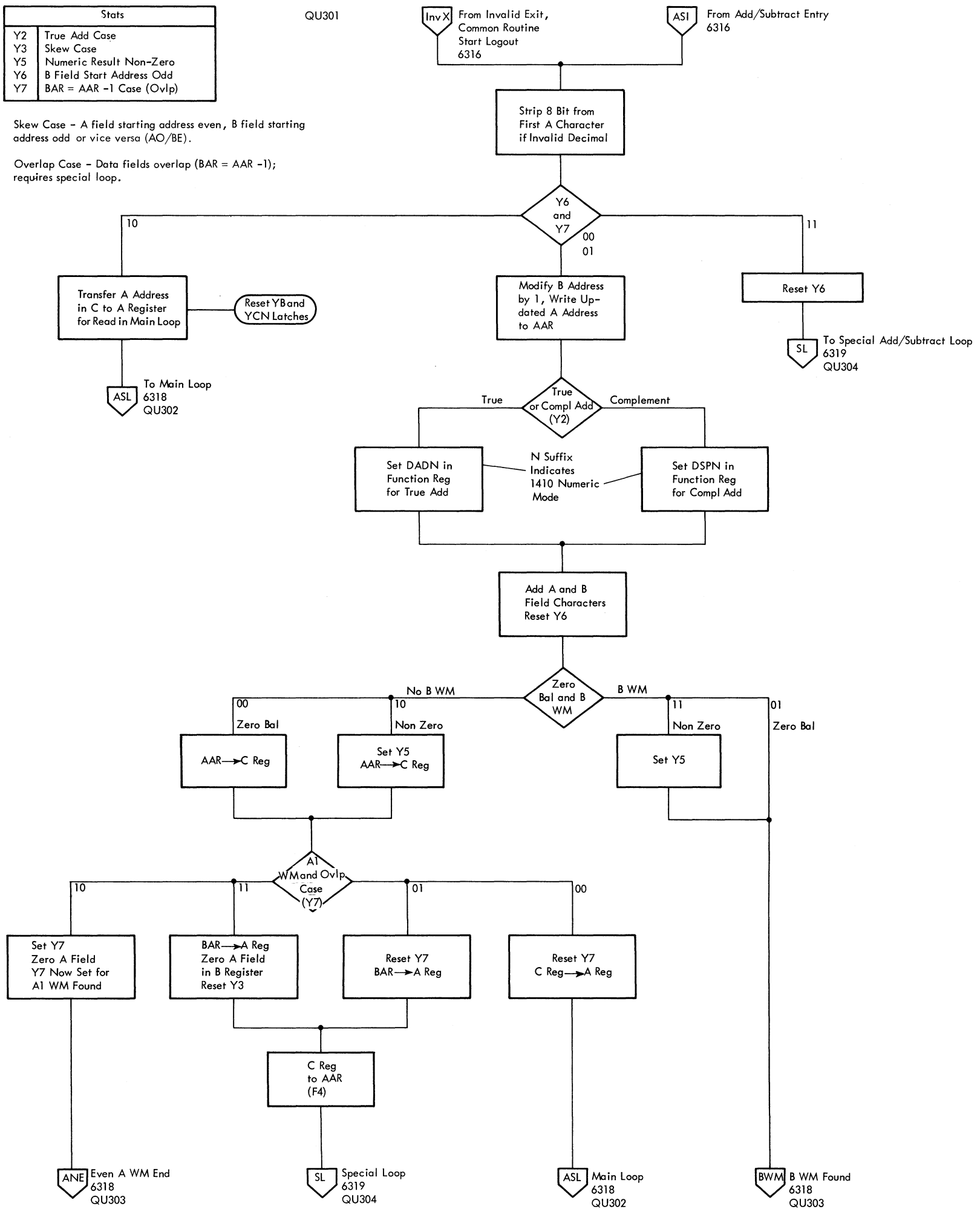


FIGURE 6317. 1410E ADD, SUBTRACT-INITIAL LOOP

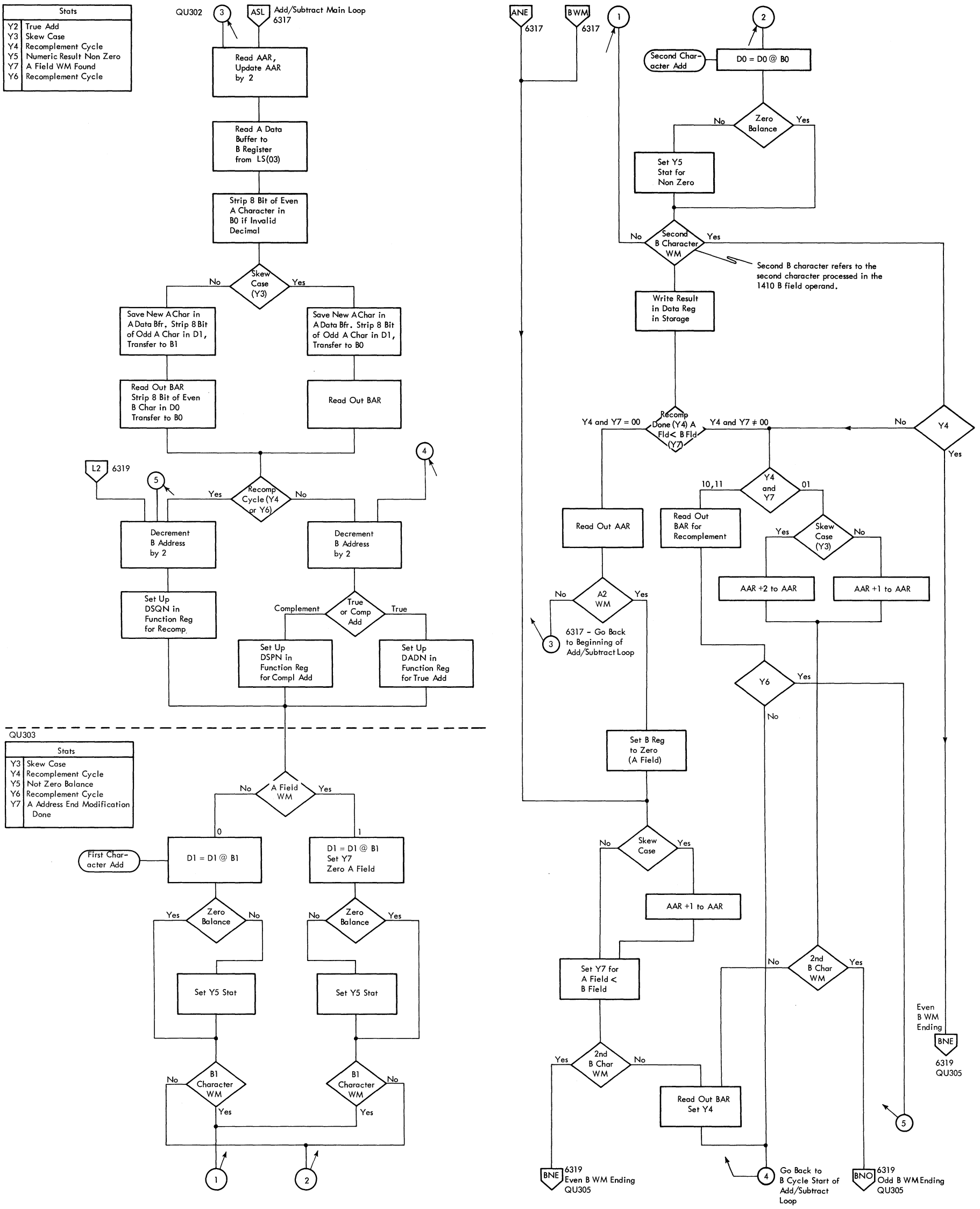


FIGURE 6318. 1410E ADD, SUBTRACT-MAIN LOOP

Stats	
Y2	True Add
Y3	No A WM before B WM
Y5	Non Zero Result

QU304

This loop handles overlapping fields. B cycles only are used.

SL 6317 Add/Subtract Special Loop

Reset Y6

True or Complement Add (Y2)

True: Set DADN in Function Register for True Add

Complement: Set DSPN in Function Register for Compl Add

Y3 = 0 Now Means No A WM before B WM

A Field WM (Y3)

No: D1 = B1@ D1

Add First Character

Yes: D1 = B1@ D1 Reset Y3

Zero Bal and Y3

A WM Found

Zero Balance

01: Transfer Result to B Register

Result Non Zero

11: Transfer Result to B Register Set Y5

A WM Not Found

Zero Balance

00: Set A Field to Zero (B=Z)

Result Non Zero

10: Set A Field to Zero (B = Z) Set Y5

1st B Char WM

No: Add Second Character

Zero Balance

No: Transfer Result to B Register Set Y5

Yes: Transfer Result to B Register

2nd B Char WM

No: Update B Address by (-2)

Yes: Update B Address by (-2)

Yes: Update B Address by (-1)

Ending B Address +1 to AAR Reset Y6

Stats	
Y2	True Add
Y3	Skew
Y4	Result has been recomplemented
Y5	Num Result Non Zero
Y6	Recomplement Done
Y7	A Field WM Found

MSD = Most Significant Digit
ZBI = Zero Balance Indicator

QU305

BNO 6318 Odd B WM Ending

Clear BAR for Ending Address, Modify BAR+1 to DAR

BNE 6318 Even B WM Ending

Clear BAR for Ending Address, Write

True or Complement Add (Y2)

QU306

Set CPU Indicator Routine

Carry Out of Most Signif Digit of Result

No: Read Out CPU Indicator Reset Y2

Y2 Stat Now Indicates Arithmetic Overflow

Yes: Read Out CPU Indicator

Zero Bal in Final Result (Y5)

Yes: Set CPU Indicator Bit 4 ZBI On

No: Reset CPU Indicator Bit 4 ZBI Off

Write Corrected Ending B Address to BAR

Arith Overflow (Y2)

Yes: Set CPU Indicator Bit 3

-WIC Return to I-Fetch 6302 QU101

Recomp or Carry Out of MSD

00: Recomplement Not Needed

Recomp Done: Read DAR to C Reg to Write Ending Address to BAR

10: Invalid Branch Start Logout

01 Recomplement Needed: Read Out DAR for Recomplement

Starting B Address Odd/Even

Even: Insert B Field Sign Reset Y7

Recomplement Even B Character D0 = Z @ D0

Sign Held in Skew Buffer from Common Routine

Odd: Insert B Field Sign Reset Y7

B Field WM

Yes

No: Set A Field to Zero (B = Z)

Set Y4 and Y6 Reset YCN Latch Write B Field to Main Storage

L2 6318 To Add/Subtract Main Loop, Compl Entry

FIGURE 6319. 1401E ADD, SUBTRACT-SPECIAL LOOP AND ENDING

Stats	
Y2	A Field Sign Negative
Y3	Skew Case
Y4	First Pass in First Scan Loop
Y6	B Starting Address Odd
Y7	Pointer to First Multiplier Digit
Y5	A Starting Address Odd

C Register Contents:
A field starting address if AO
A field starting address minus 1 if AE

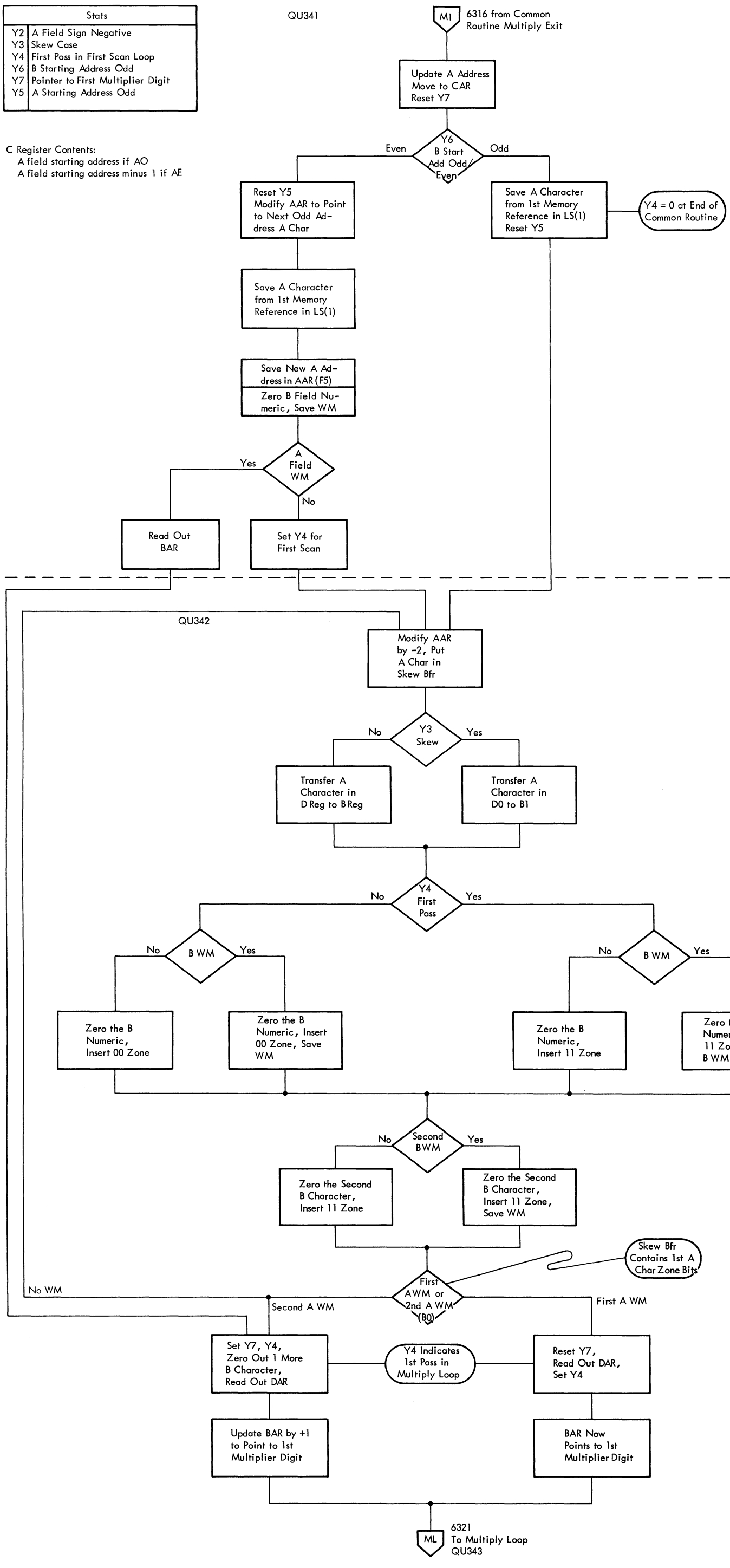


FIGURE 6320. 1410E MULTIPLY FIRST SCAN

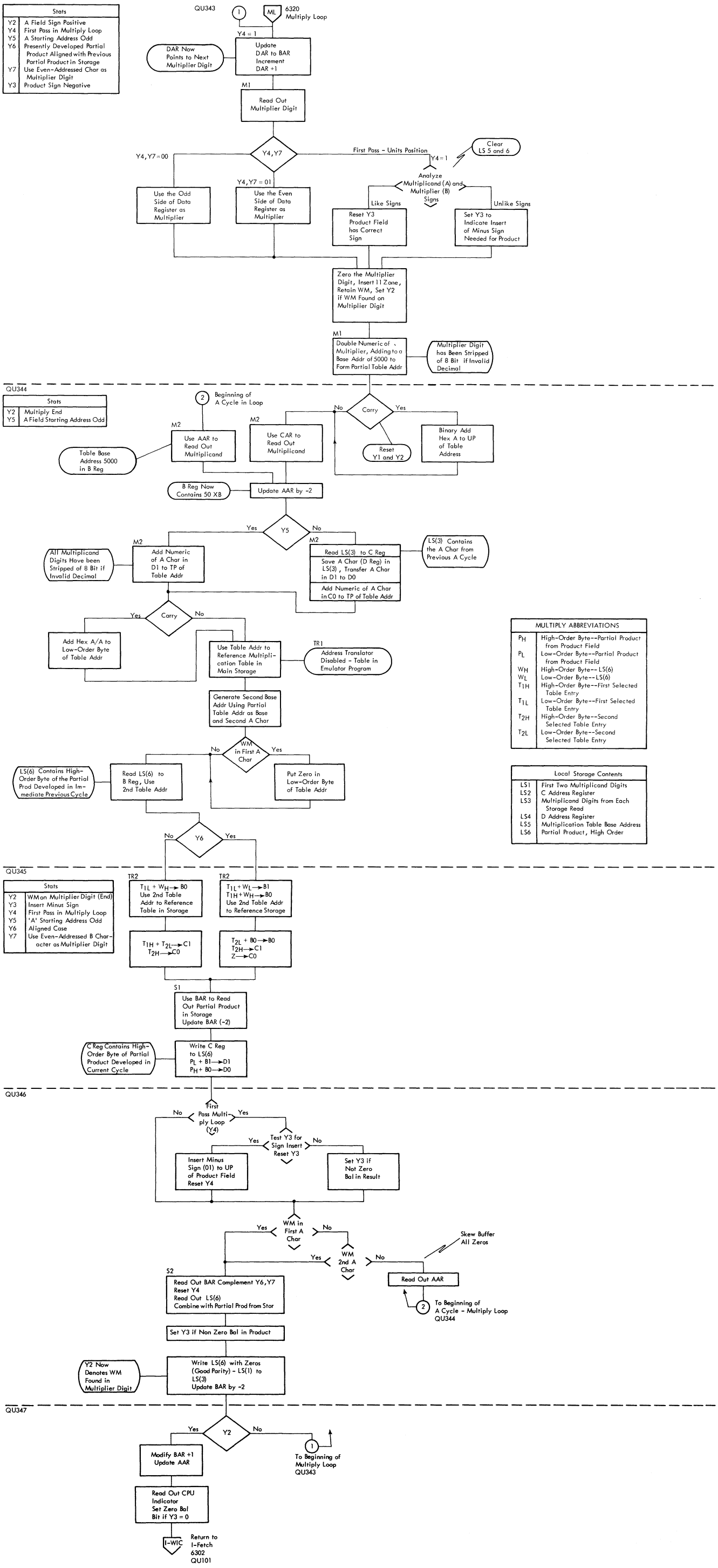


FIGURE 6321. MULTIPLY LOOP

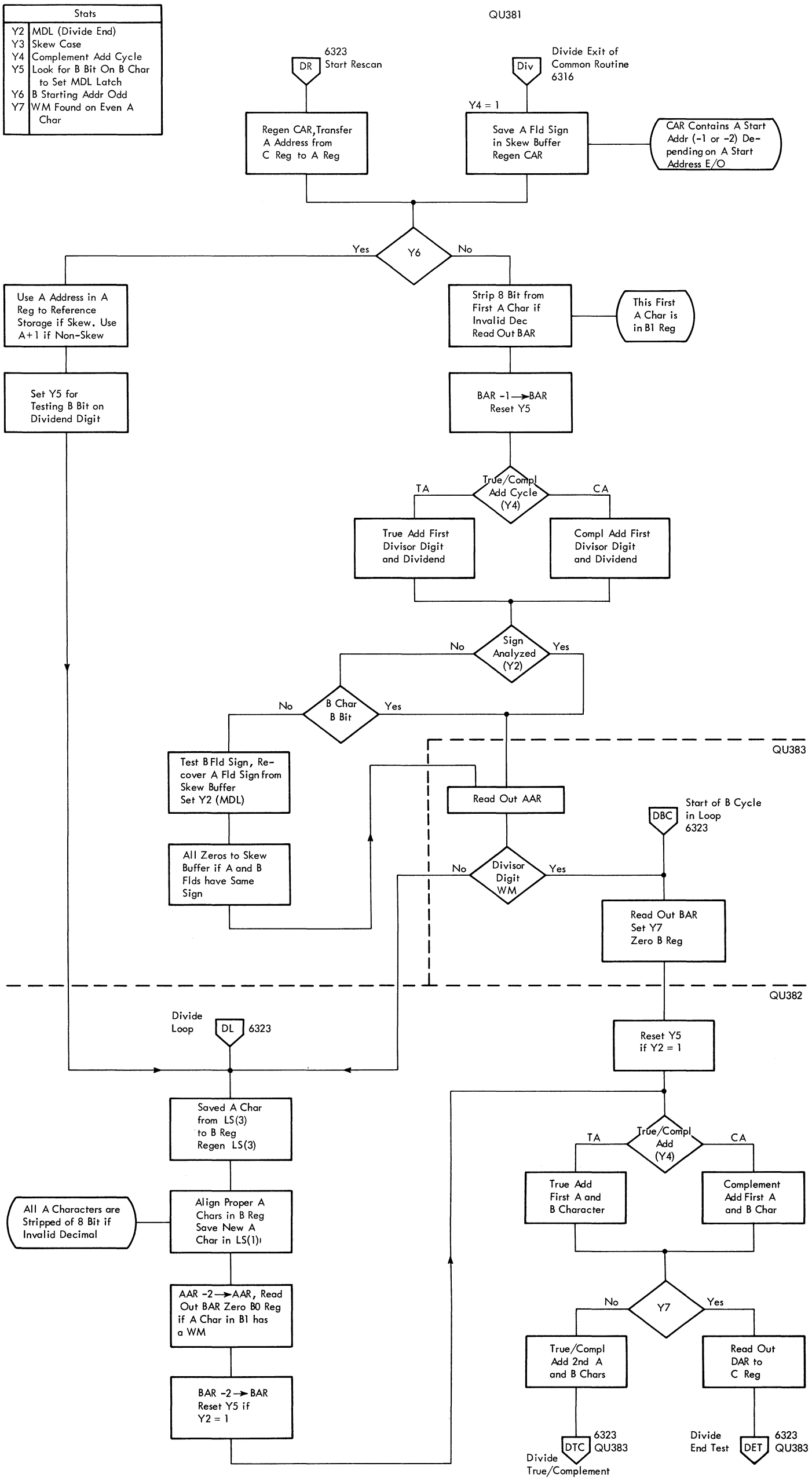


FIGURE 6322. 1410E DIVIDE INITIAL LOOP

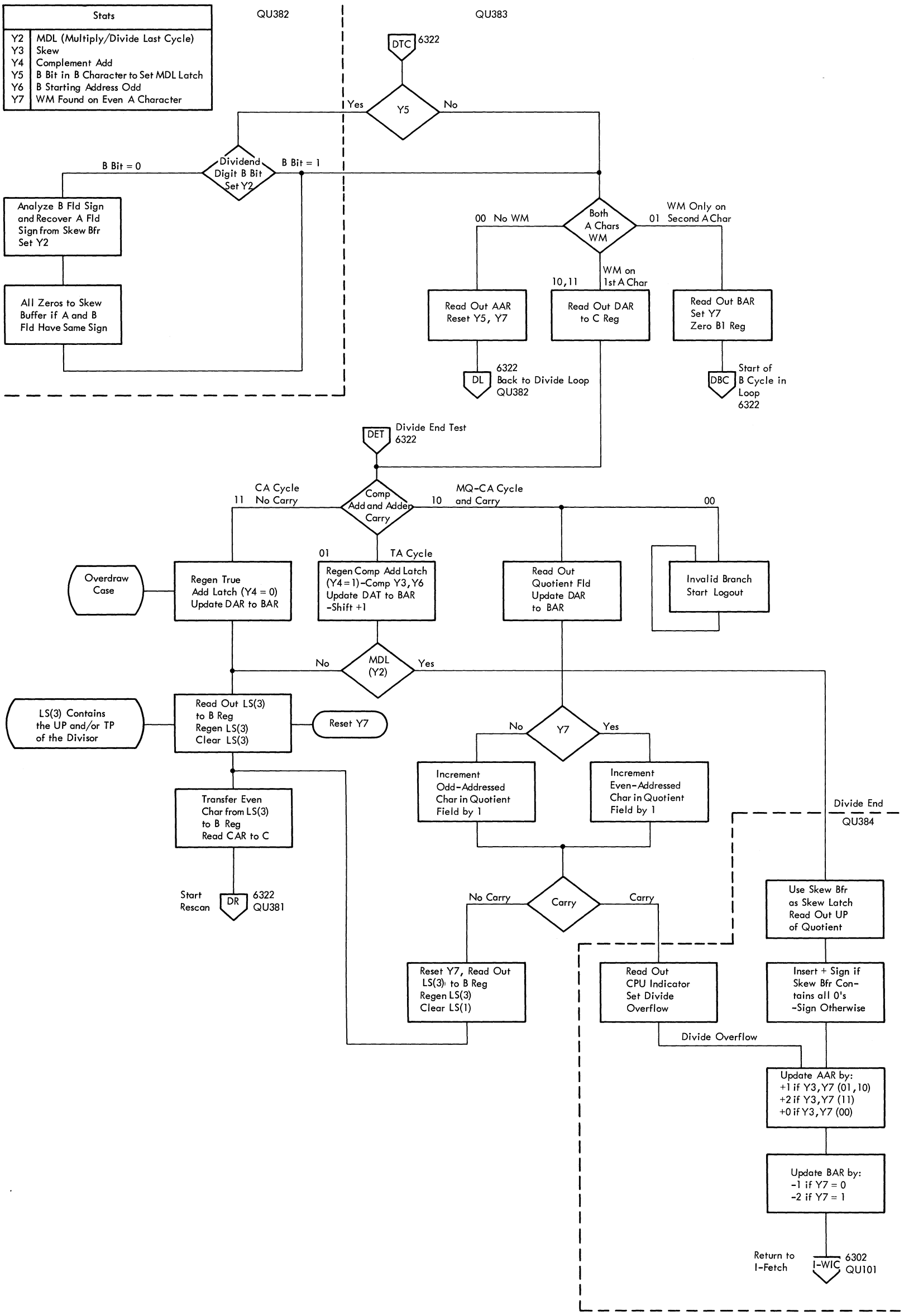


FIGURE 6323. 1410E DIVIDE LOOP AND ENDING

1410 d Modifier Numeric Bits

Bit and Status		Data Move Function
8	1	Plus Scan Data Move
	0	Minus Scan Data Move
4	1	Move WM Bit
2	1	Move Zone Bits
1	1	Move Numeric Bits

Stats	
Y2 and Y3	00 Zero Add 10 Zero Subtract 01 Compare 11 Data Move
Y4	Skew Case
Y5	Plus Scan Data Move
Y6	B Start Address Odd
Y7	(BAR = AAR 1)

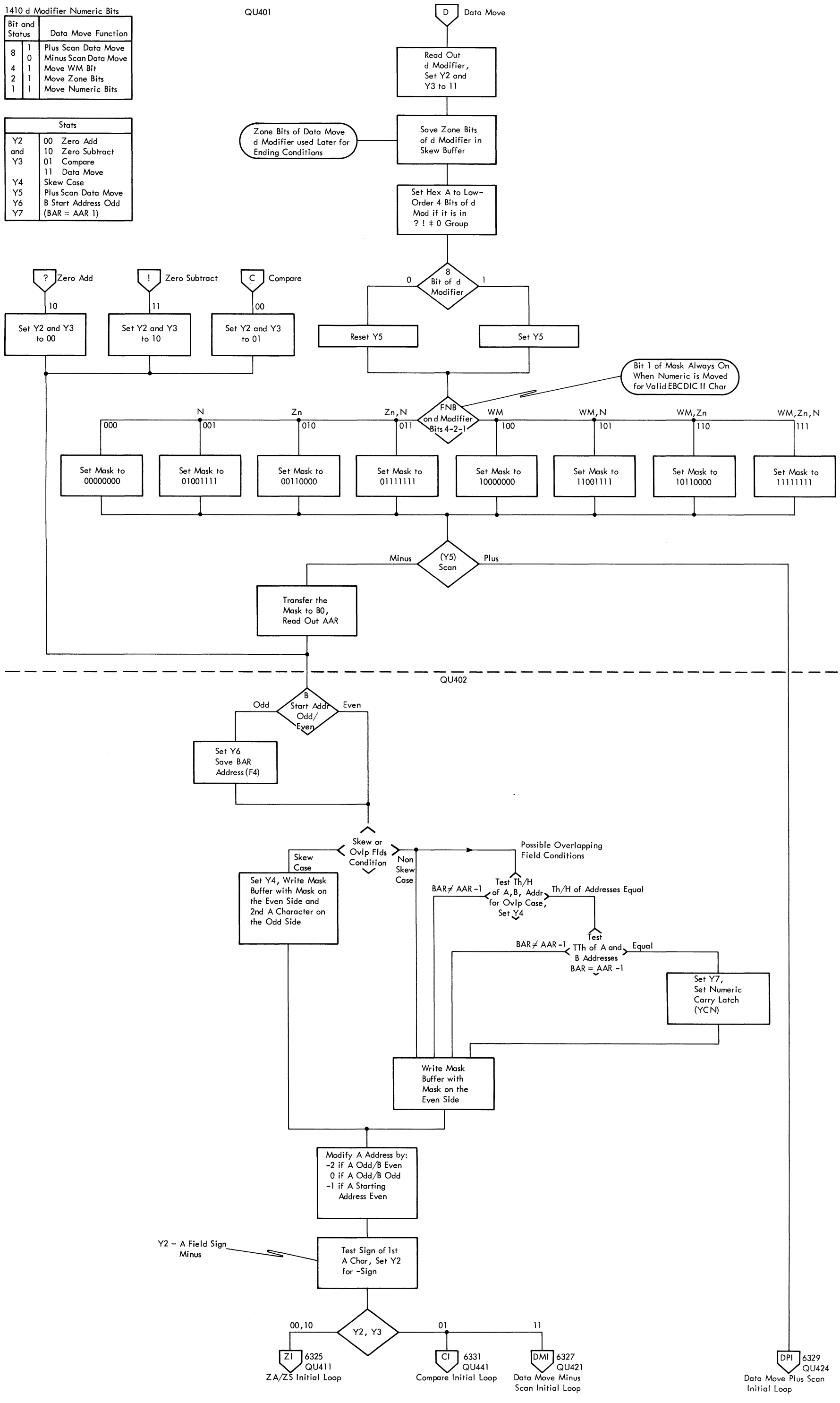


FIGURE 6324. 1410E DATA MOVE, ZERO ADD/SUBTRACT, AND COMPARE

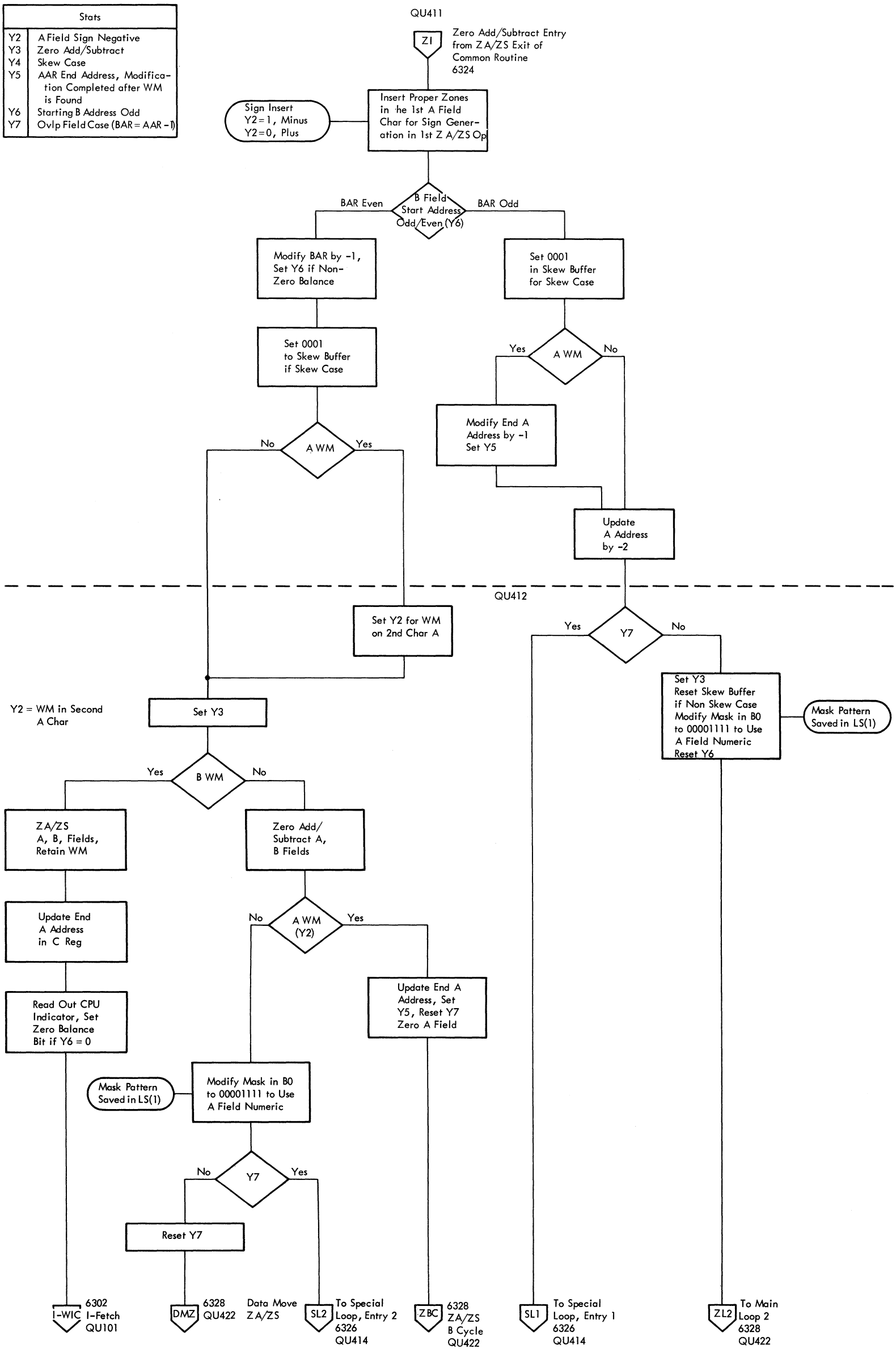


FIGURE 6325. 1410E ZERO ADD/SUBTRACT--INITIAL LOOP

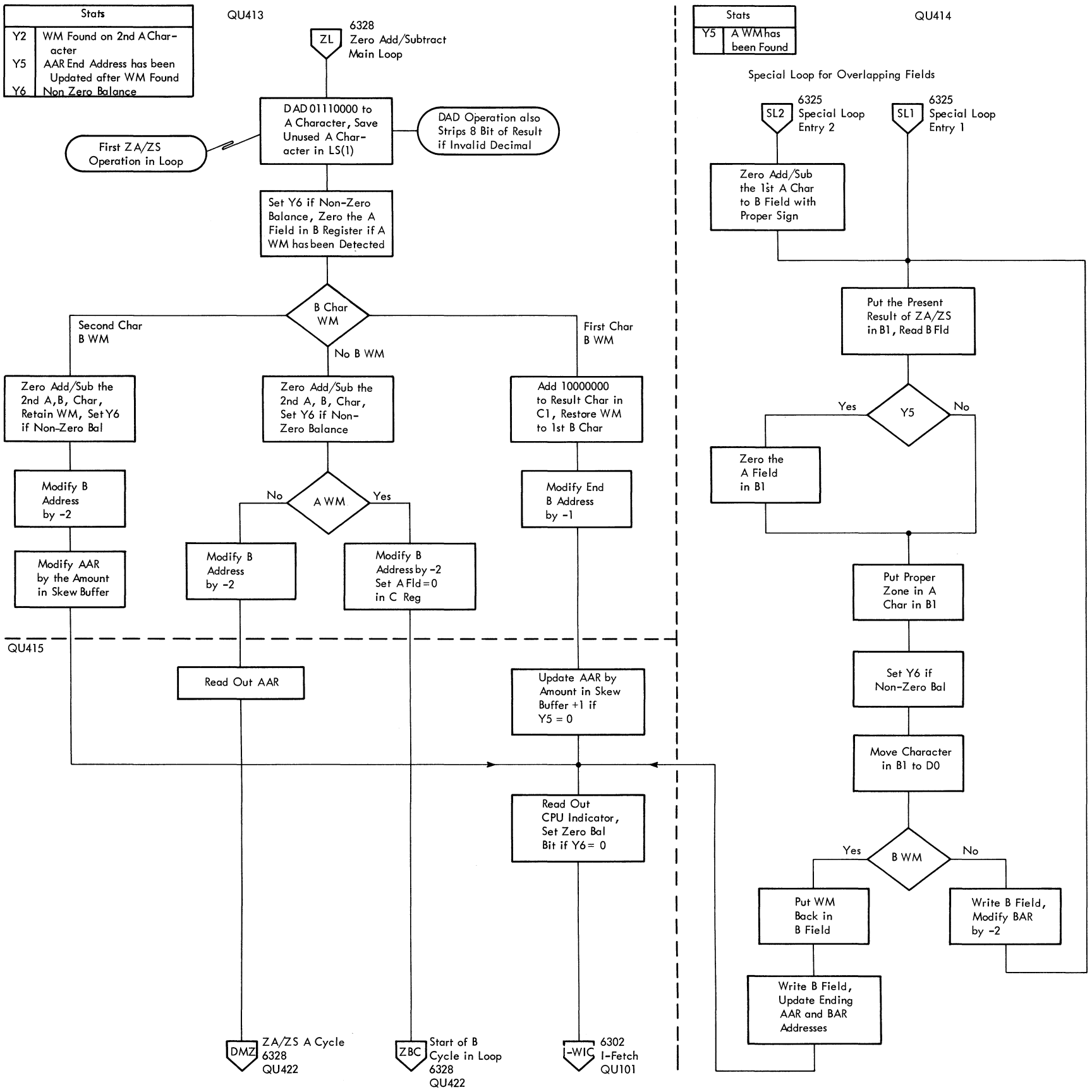


FIGURE 6326. 1410E, ZERO ADD/SUBTRACT MAIN AND SPECIAL LOOP

Stats	
Y2	End Mark Found
Y4	Skew Case
Y6	B Start Address Odd
Y7	BAR = AAR -1 (Ovlp Fields)

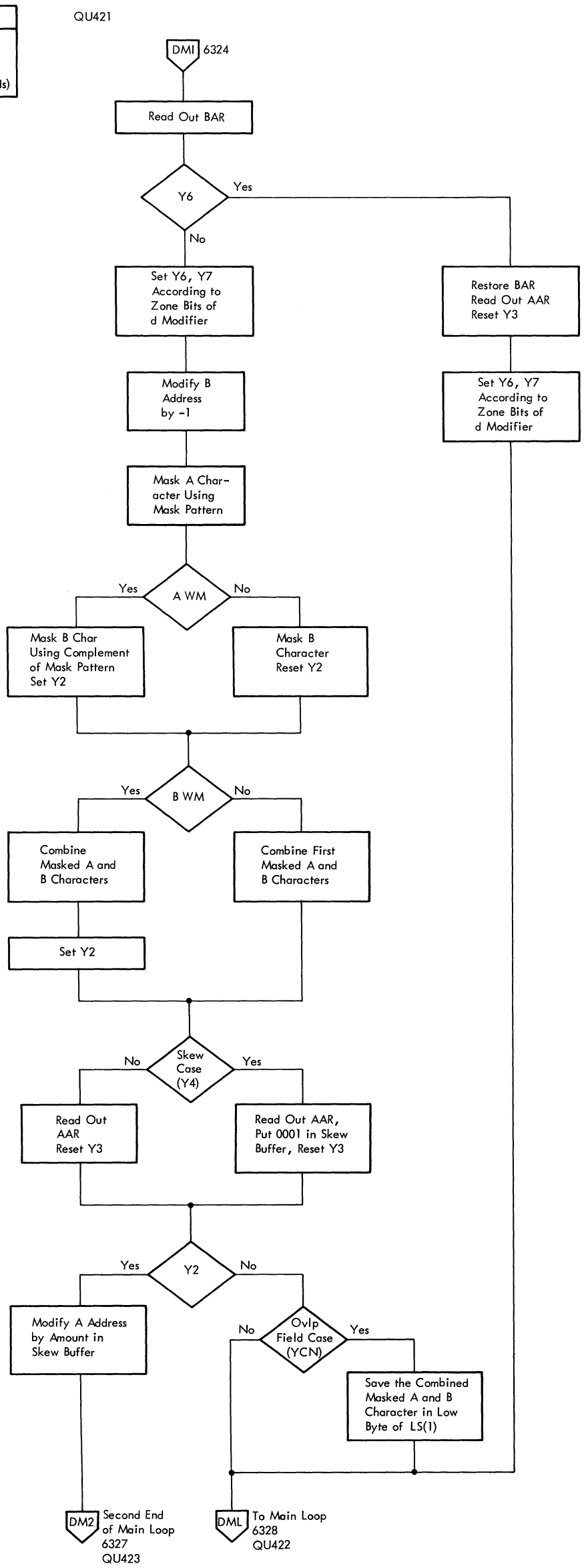


FIGURE 6327. 1410E DATA MOVE MINUS SCAN-INITIAL LOOP

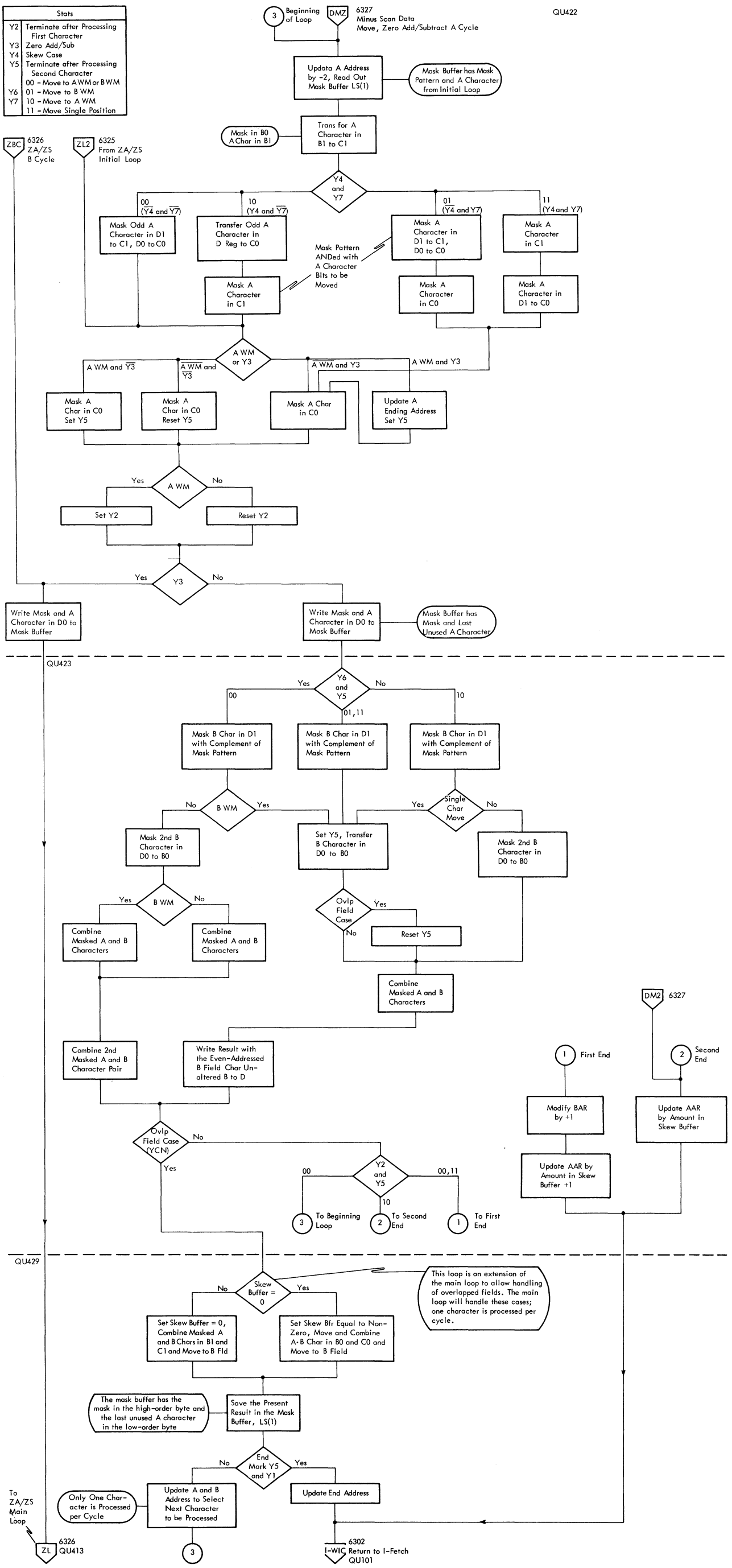


FIGURE 6328. 1410E DATA MOVE MINUS SCAN, ZERO ADD/SUBTRACT A CYCLES

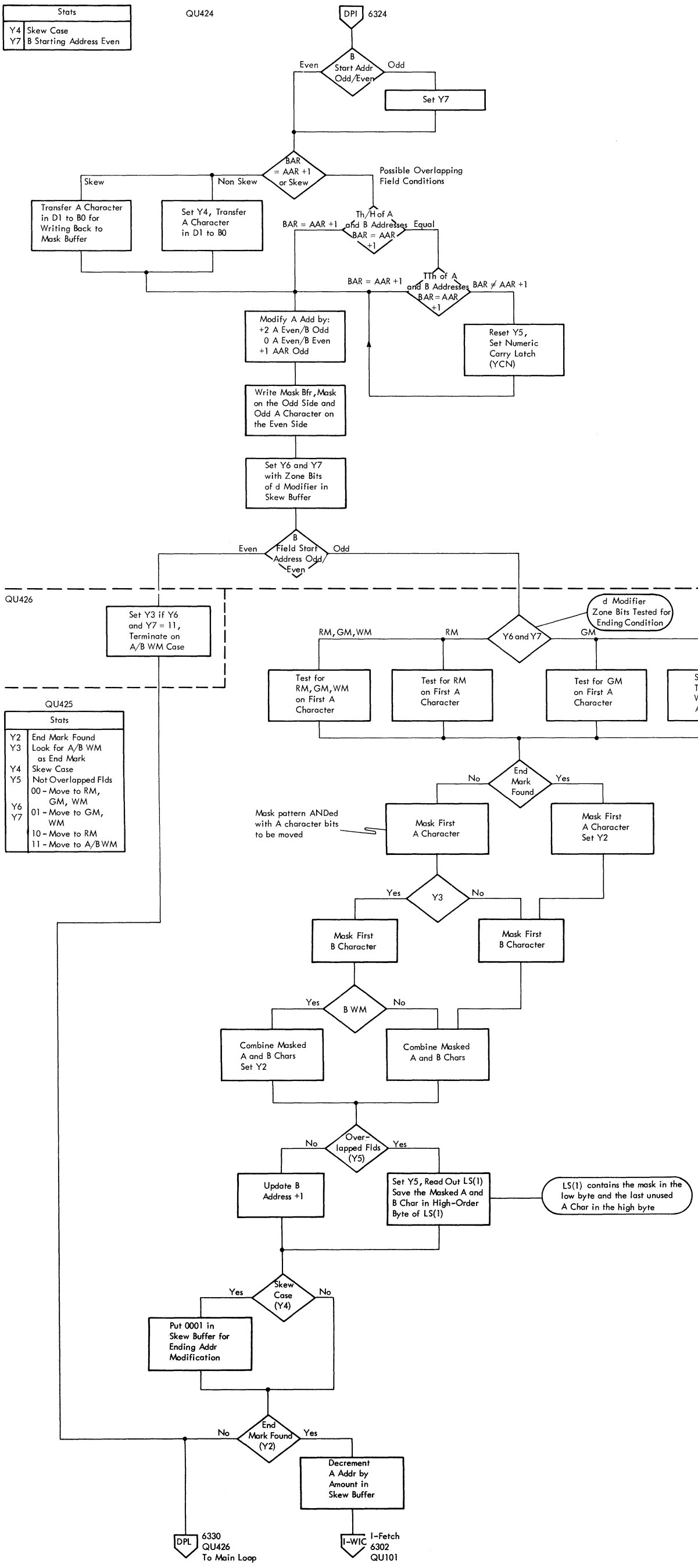


FIGURE 6329. 1410E DATA MOVE PLUS SCAN-INITIAL LOOP

Stats	
Y2	Terminate after Processing First Character
Y3	Look for A/B WM as End Mark
Y4	Skew Case
Y5	Terminate after Processing Second Character
Y6,Y7	00 - Move to RM/GM WM 01 - Move to GM WM 10 - Move to RM 11 - Move to A/B WM

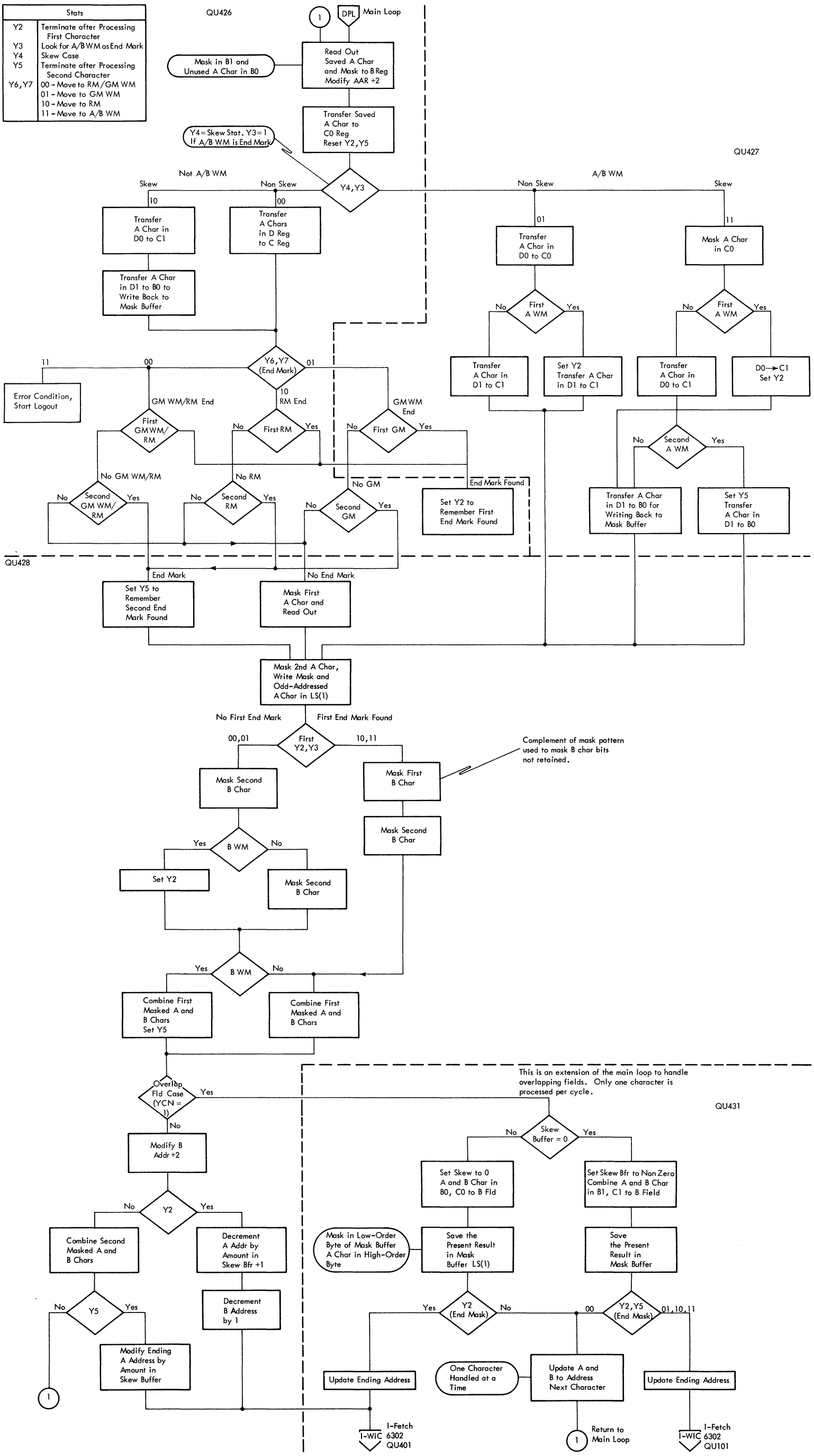


FIGURE 6330. 1410E DATA MOVE PLUS SCAN - MAIN LOOP

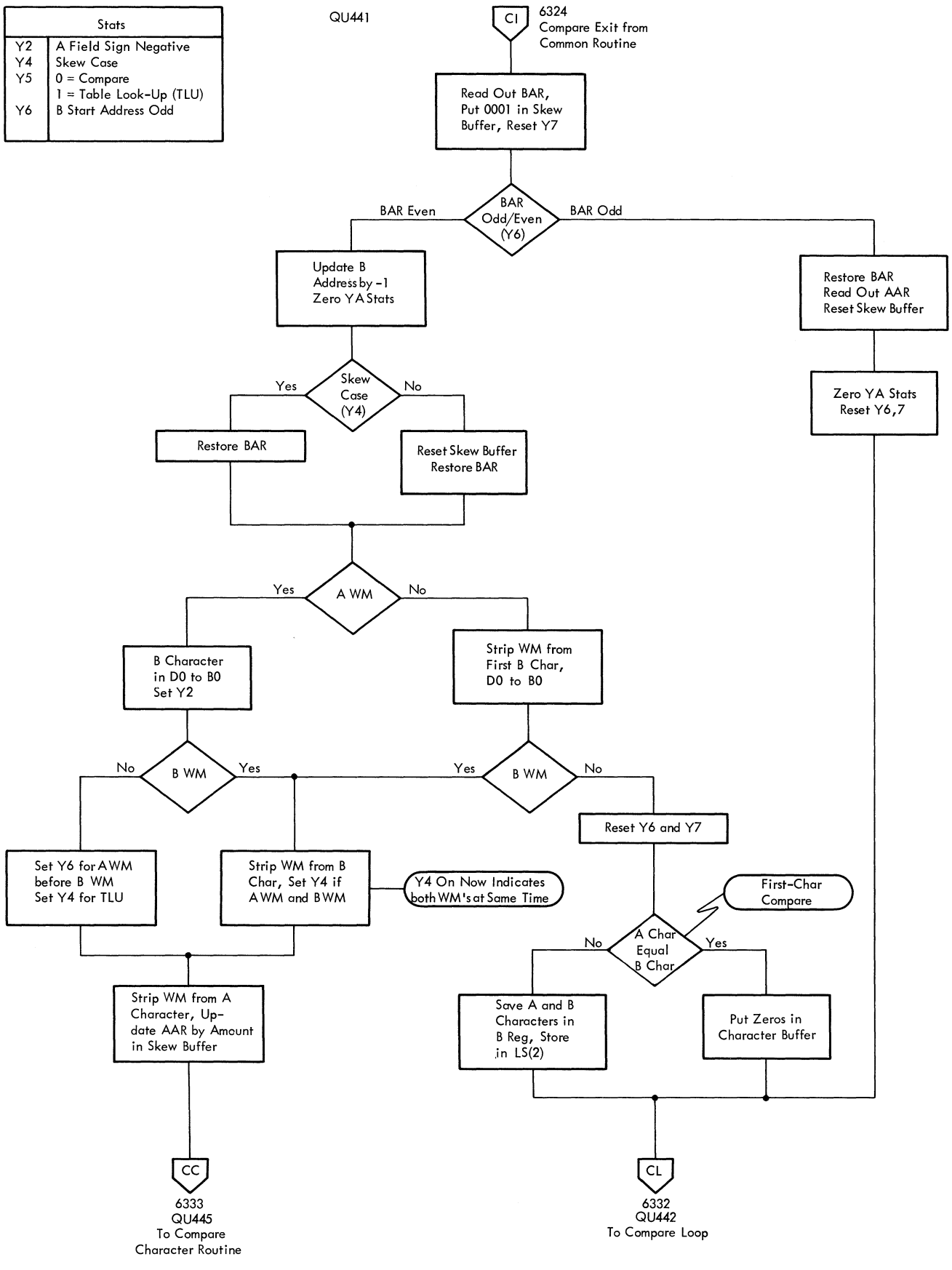


FIGURE 6331. 1410E COMPARE INITIAL LOOP

Stats	
Y2	A WM on Second Character
Y3	A WM on First Character
Y4	Skew Case
Y5	Table Look-up (TLU)
Y6	A WM before B WM (B > A)

QU442

1

CL 6331 Compare Loop

Update A Address by -2, Read A Data Buffer to B Reg Reset Y2 and Y3

Y4 and A WM

Y4 = 0

Transfer A Character in D1 to B1

2nd A WM — Yes: Set Y2; No

Y4 = 1 No A WM

Transfer A Character in D1 to B0

2nd A WM — Yes: Set Y2, Transfer the Unused A Char in D0 to D1; No: Transfer the Unused A Char in D0 to D1

Y4 = 1 and A WM

Set Y3, Transfer A Character in D1 to B0

Update B Address by -2, Strip WM from 2nd B Char

2nd B Char WM — No: Set Y6 (A < B) Update AAR by Amount in Skew Buffer; Yes: Compare 2nd A and B Character Pair, Reset Y2

Equal — No: Put Last Unequal A and B Char Pair in B Reg for Compare; Yes: Transfer Last Unequal Char Pair to B Reg for Compare

Update AAR by Amount in Skew Buffer

CC 6333 QU445 Compare Char Routine

QU443

1st A Char WM (Y3) — No: B WM and Chars Unequal

01 No B WM, Char Unequal

QU444

Transfer 1st B Char to C0

Y2 — Yes: Strip WM from B Char; No: Compare 2nd A and B Char Pair, Read Char Bfr to C Reg

Y2 — Yes: Compare 2nd A and B Character, Clear Character Buffer, Reset Y4

00 No B WM Characters Equal

Y2 — Yes: Strip WM from B Char; No: Compare 2nd A and B Char Pair, Read Char Bfr to C Reg

2nd A and B Char Equal and B2 WM

B WM: Strip WM from B Char, Set Y4 for TLU if A WM and B WM at Same Time

Characters Equal: Write Last Unequal Char Pair in Char Buffer

Not Equal: Transfer Present Unequal A and B Char Pair to Char Buffer

Read Out AAR

1

1st A Char WM (Y3) — Yes: B1 WM

10,11 B WM

B1 WM — Yes: Strip WM and A and B Char, Compare A and B Char, Set Y4 for TLU if A and B WM at Same Time; No: Set Y6 to Remember A < B (Set High) Set Y4 for TLU

A and B Char Pair Equal — No: Update BAR by +1, Unequal Char Pair to B Register; Yes: Update BAR by +1, Read Out Character Buffer to B Reg

Character Buffer, LS(2), Has Last Unequal Pair

Reset Y2

FIGURE 6332. 1410E COMPARE MAIN LOOP

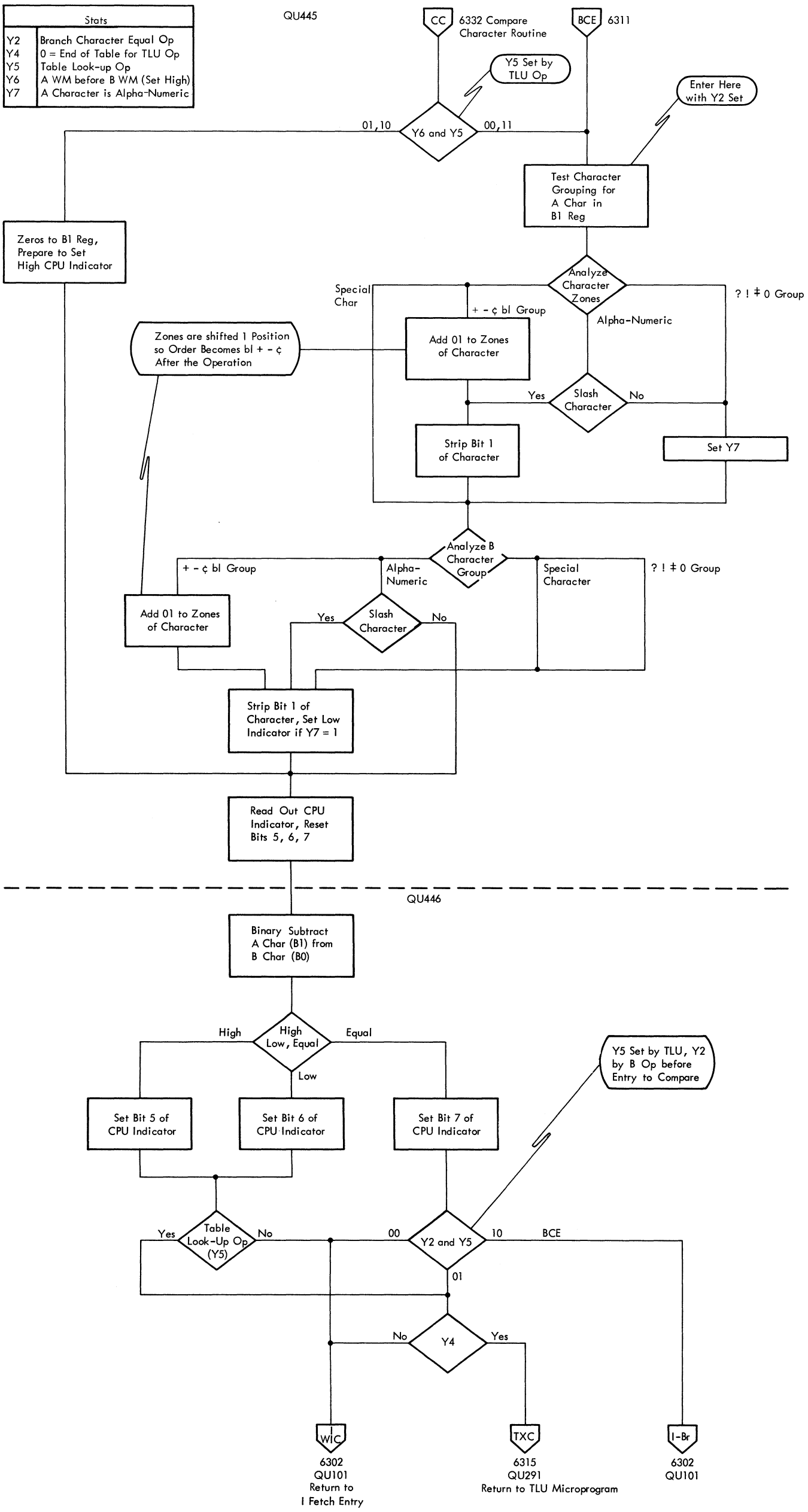


FIGURE 6333. 1410E COMPARE CHARACTER AND ENDING

Stats	
Y3	Non-overlap
Y4	F Channel Op
Y5	Unit Control Op
Y6	Overlap

Register Contents	
B0	Translated Op
C0	System Status
C1	d Modifier
D0	Ch 1 Branch Conditions (R)
D1	Ch 2 Branch Conditions (X)
H	Hex F1
J	Hex F8

Op Code Analysis	
Bits 0 and 1	Format
00	Not used
01	X-control field plus addr plus d-mod (M/L)
10	d-Modifier only (F, K)
11	X-control plus d-mod (U)

d-Modifier Translated Character	
Bit	Valid for
0	X2 - M/L
1	d-Mod - M/L
2	d-Mod - U
3	d-Mod - F

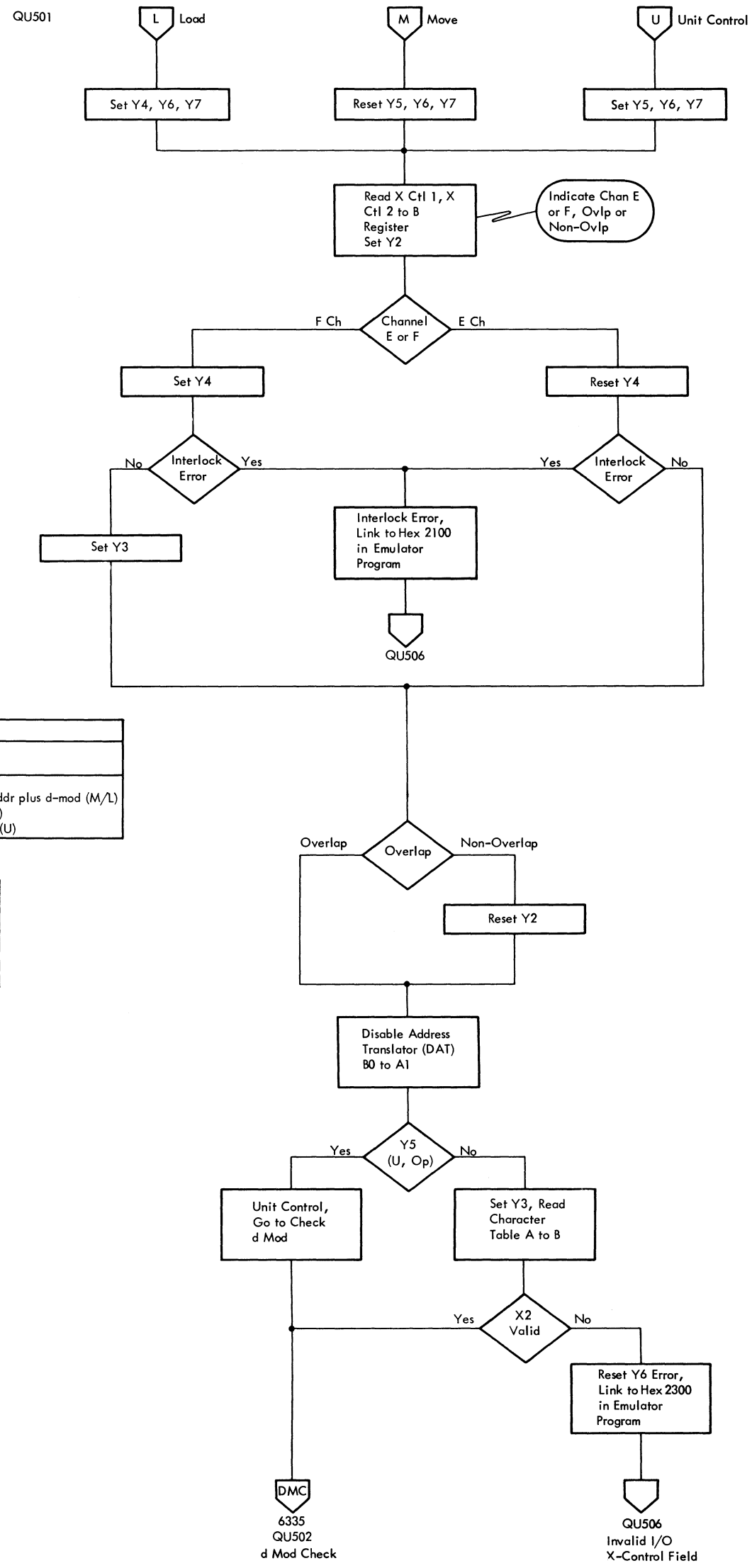


FIGURE 6334. 1410E UNIT CONTROL, I/O MOVE/LOAD, INITIAL LOOP

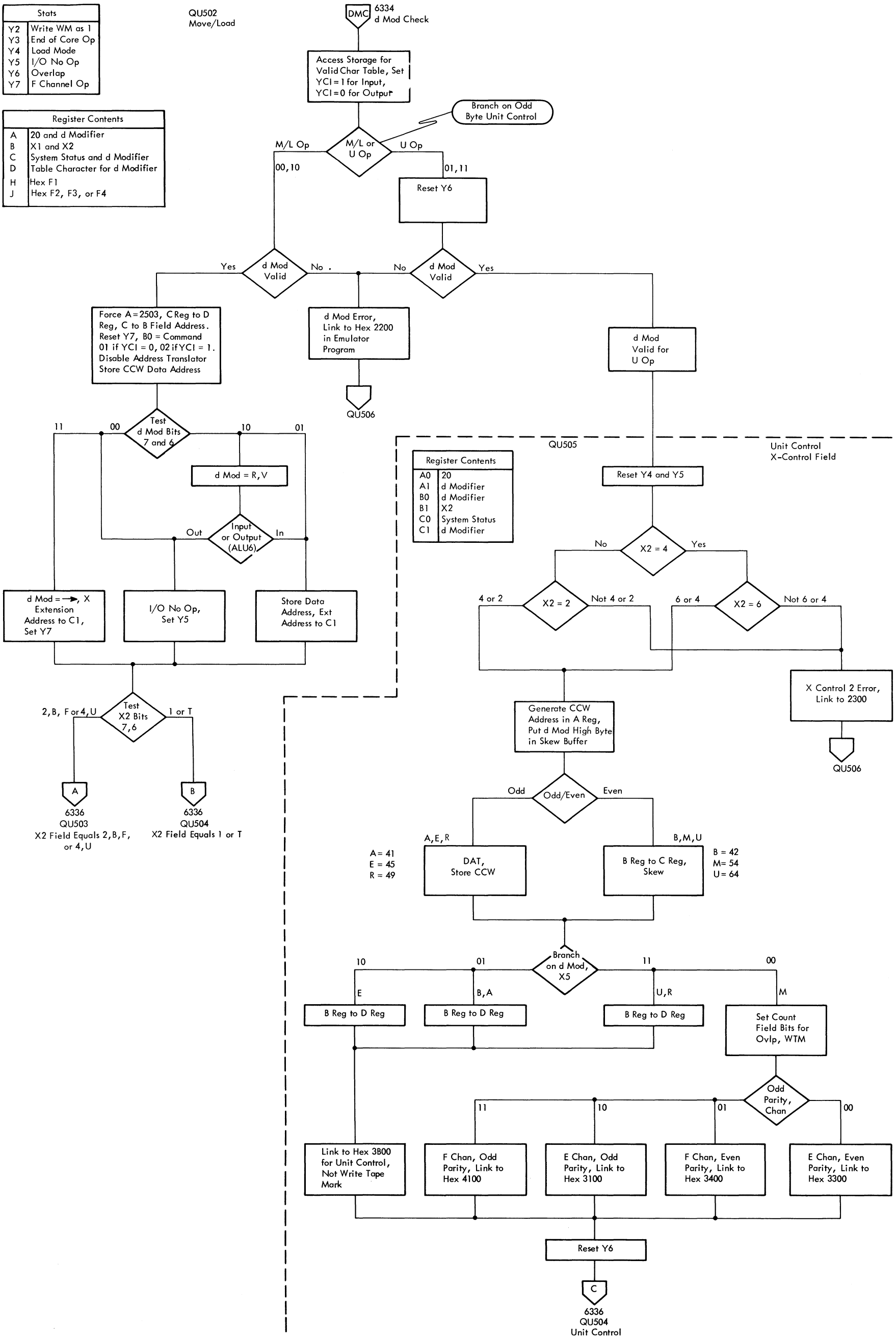


FIGURE 6335. X-CONTROL FIELD TRANSLATION

Device	X2	Code
Punch	4	74
Tape Even	U	64
Tape Odd	B	42
Printer	2	72
Disk	F	46

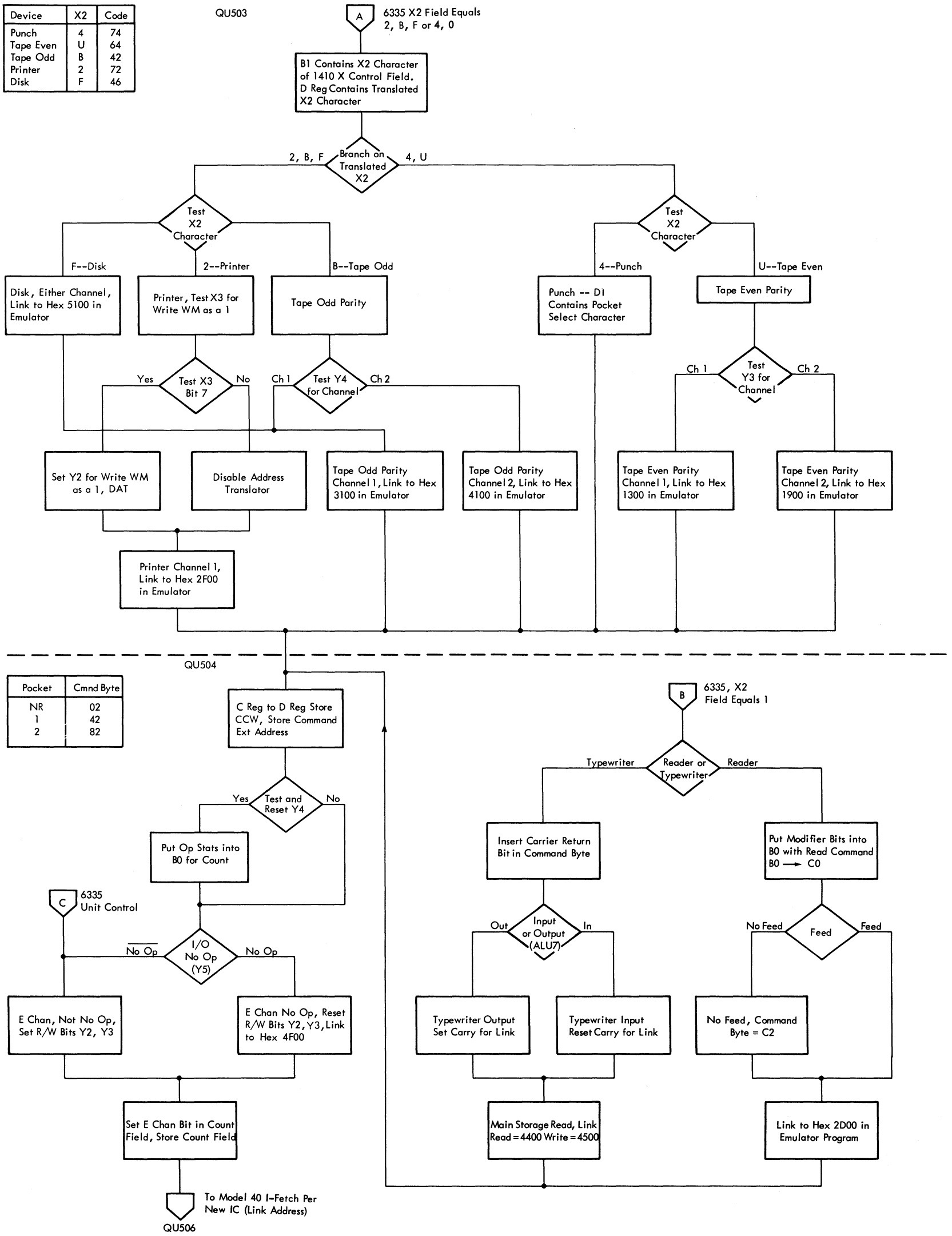


FIGURE 6336. 1410E I/O UNIT SELECTION

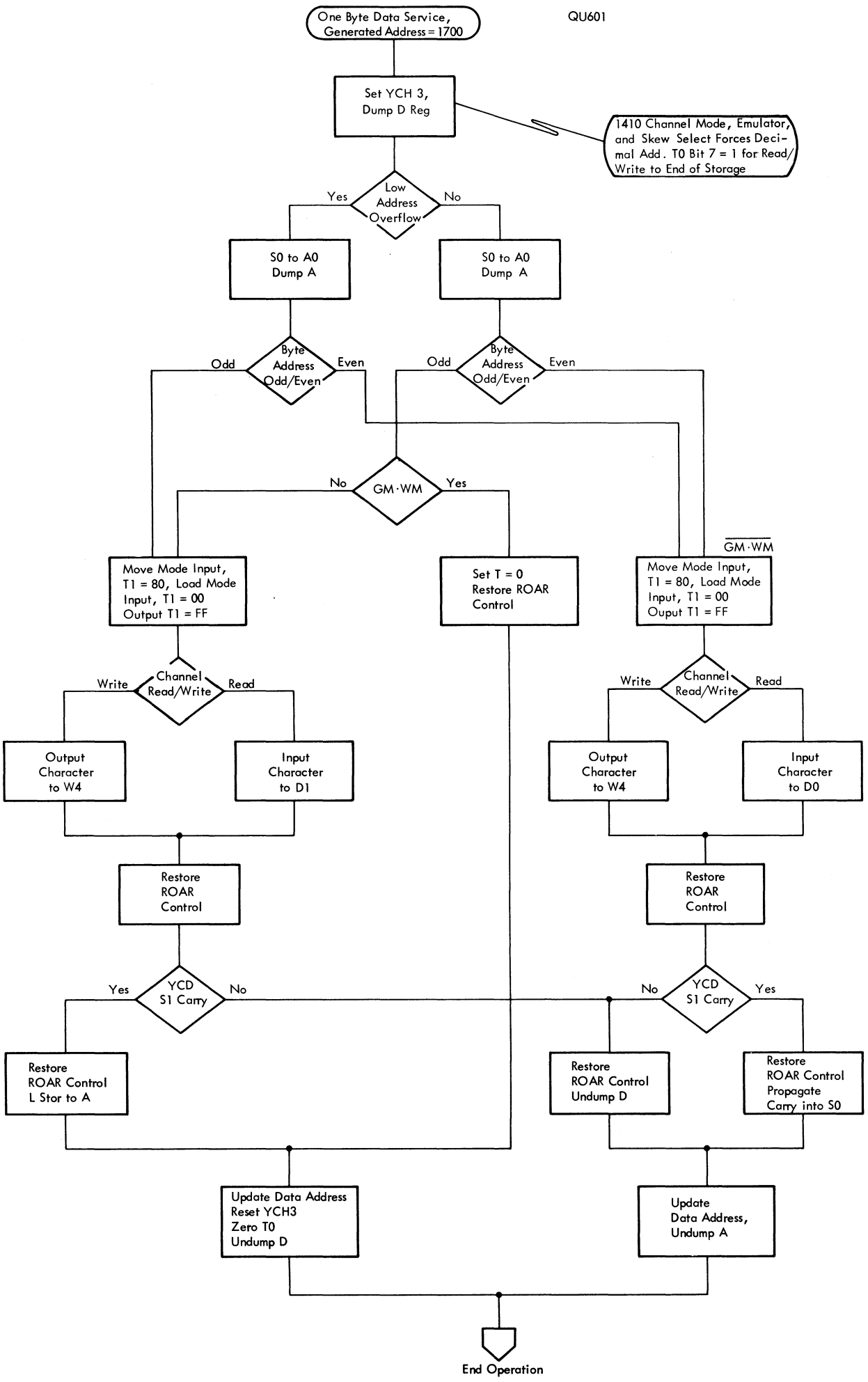


FIGURE 6337. 1410E, ONE-BYTE DATA SERVICE

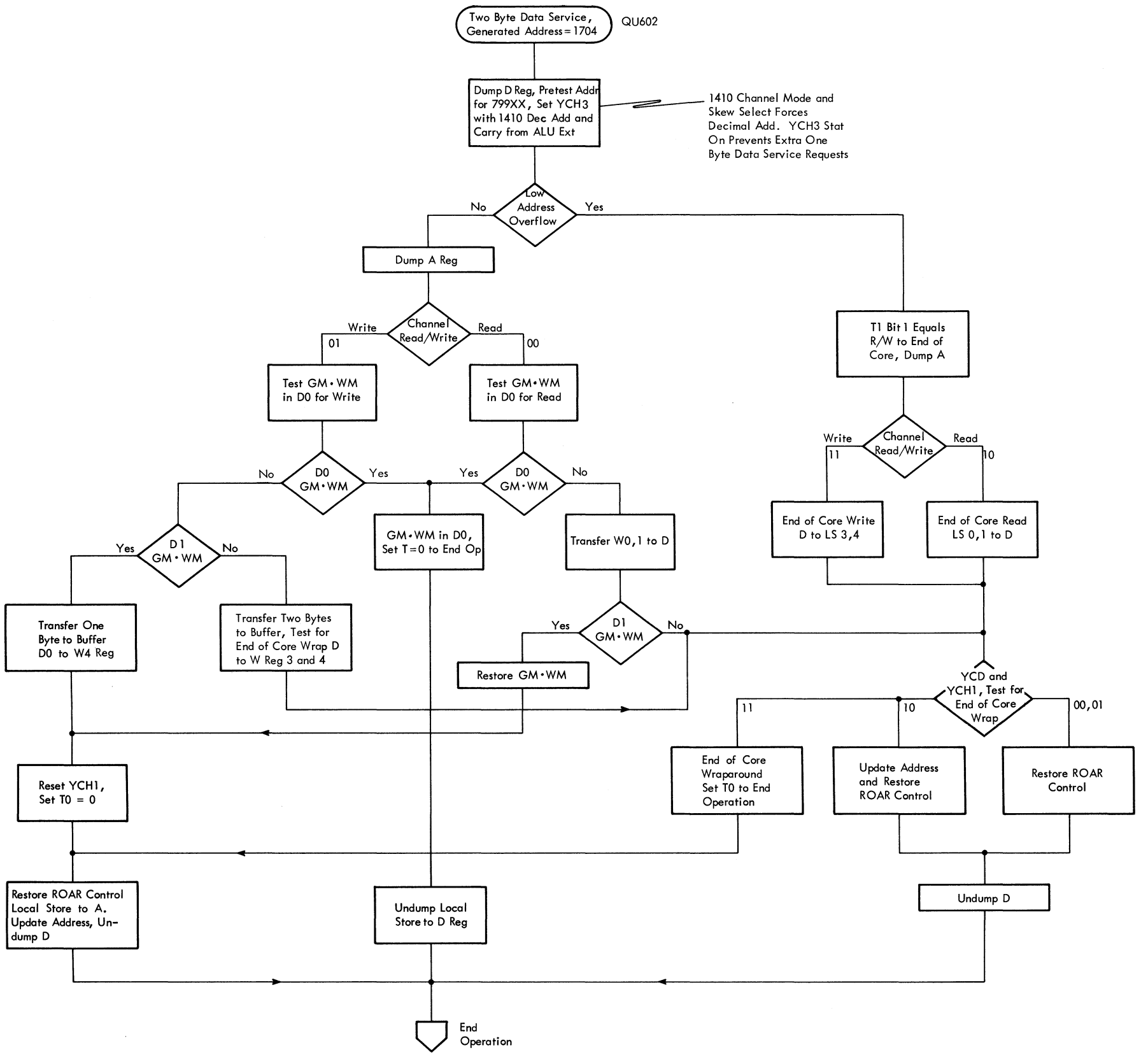


FIGURE 6338. 1410E, TWO-BYTE DATA SERVICE

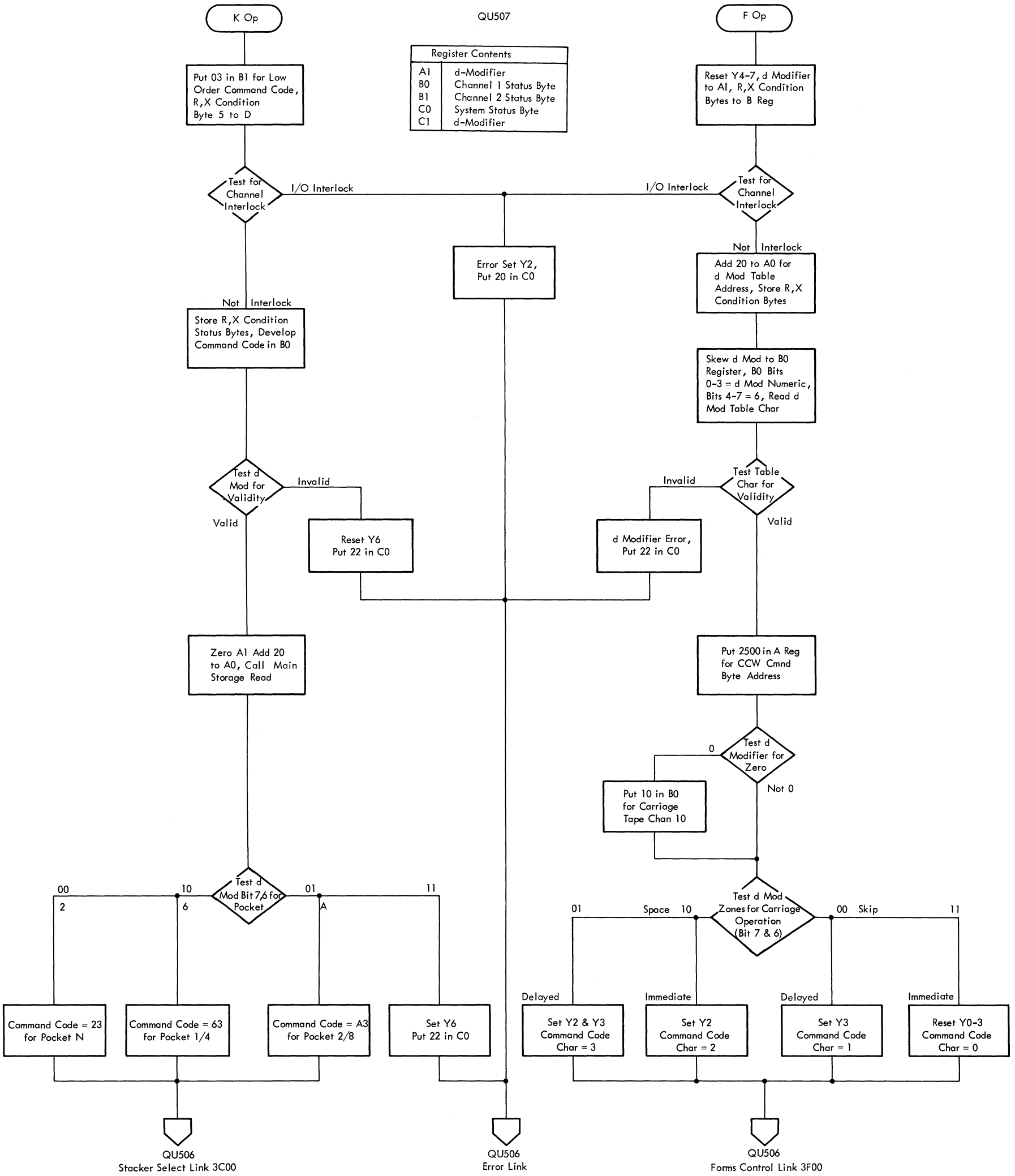


FIGURE 6339. 1410E FORMS CONTROL, STACKER SELECT

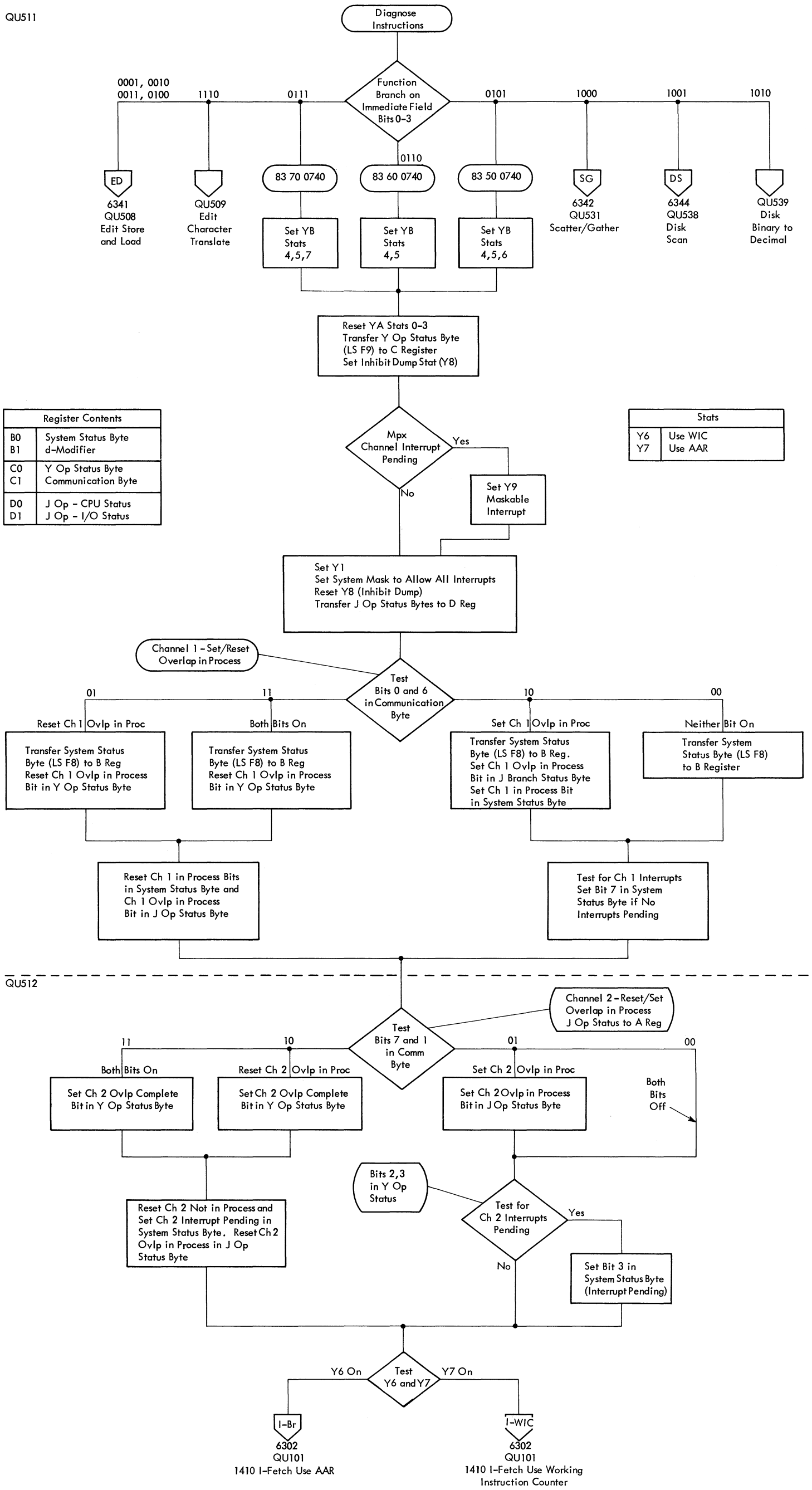


FIGURE 6340. 1410E DIAGNOSE INSTRUCTION, I-FETCH LINKAGES

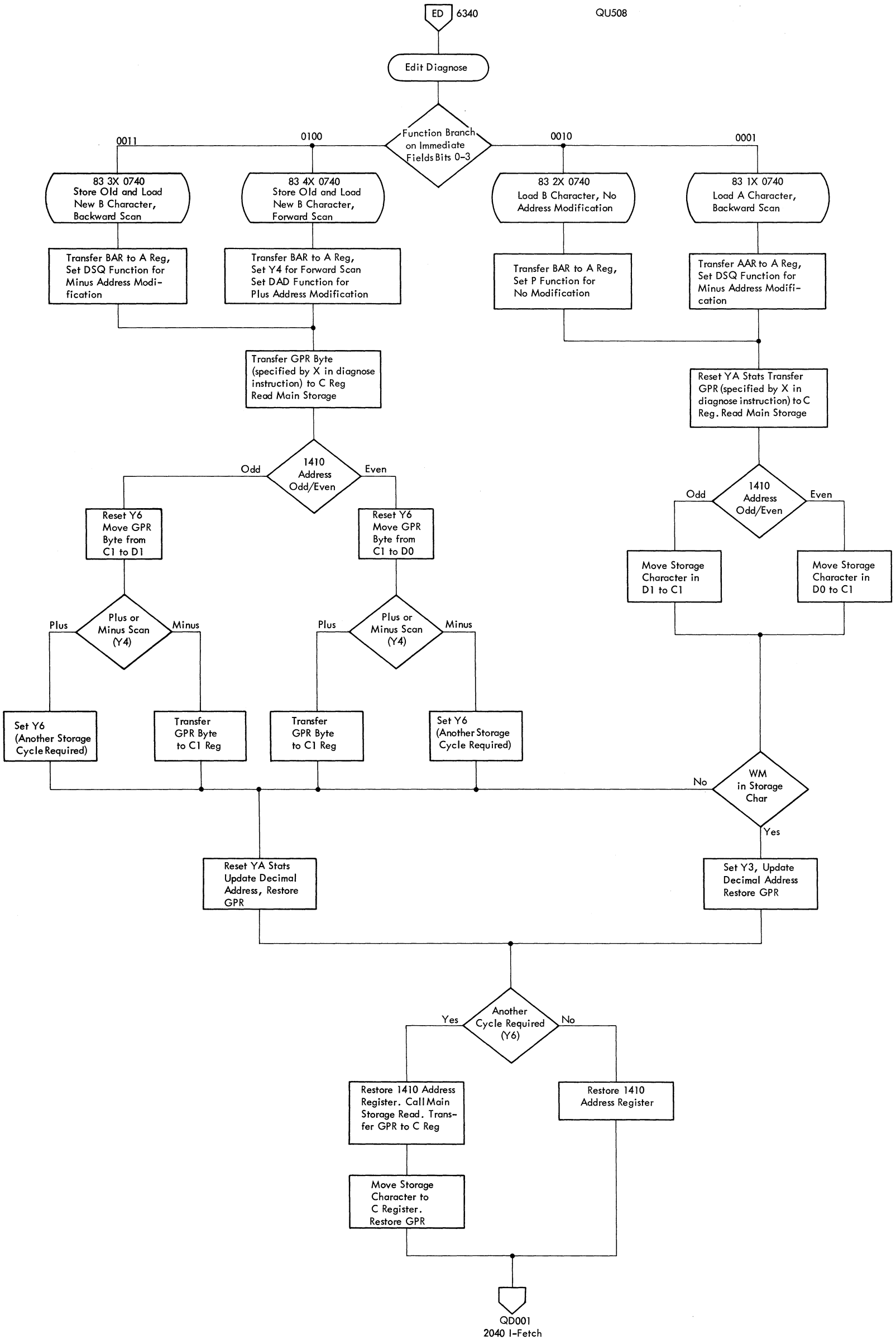


FIGURE 6341. 1410E EDIT DIAGNOSE

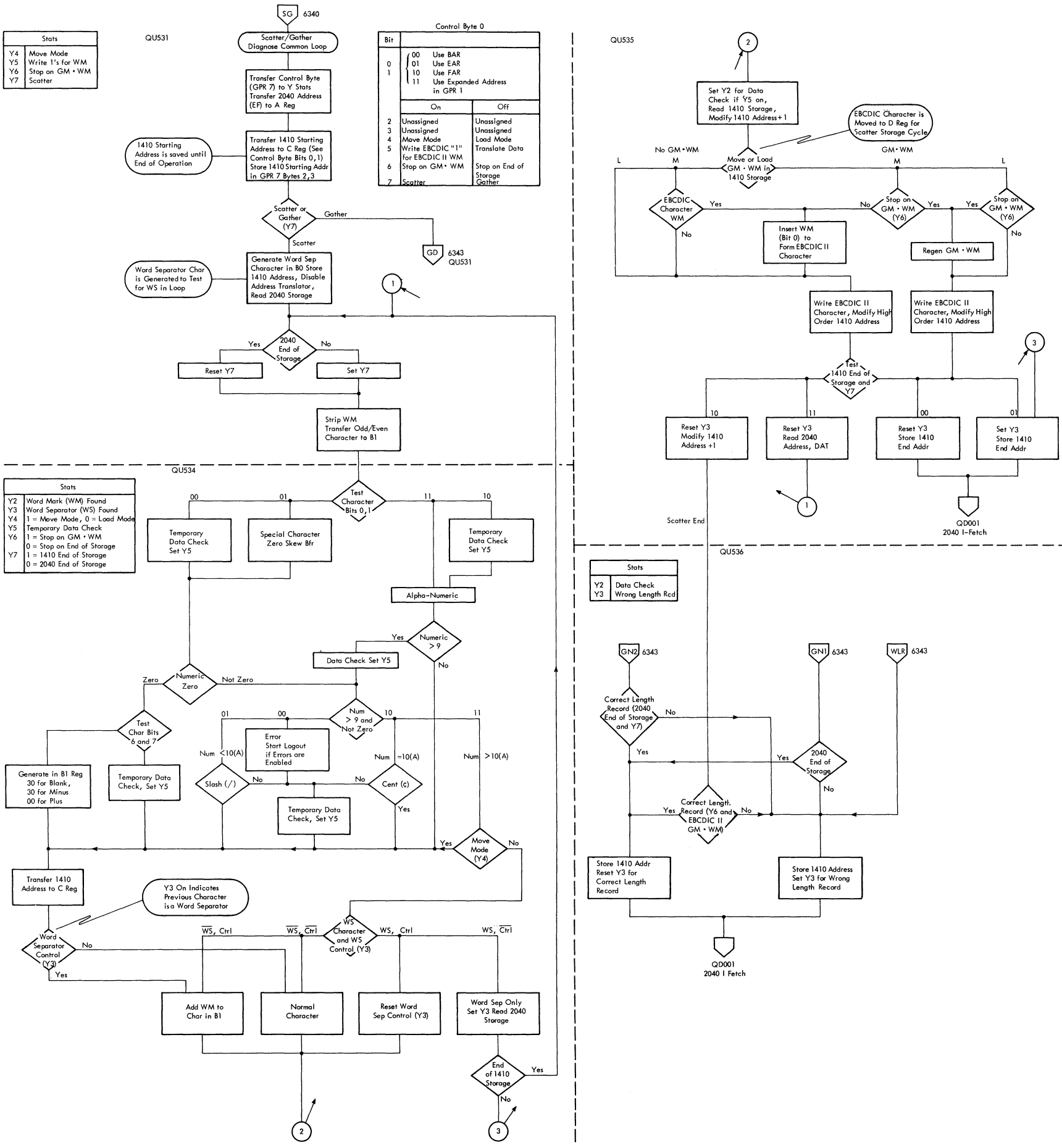


FIGURE 6342. 1410E SCATTER/GATHER DIAGNOSE

Stats	On	Off
Y4	Move Mode	Load Mode
Y5	Write 1's for WM	Translate Data
Y6	Stop on GM·WM	Stop on End of Storage
Y7	Scatter	Gather

QU531

GD 6342

Gather Diagnose Exit from Common Scatter, Gather Routine

Reset Y 0, 1 and 3

Test Y4 and Y5

11 | 10 | 01 | 00

Set Y4 Meaning Write 1s for WMs

Generate Hex FD in B0

Generate Word Separator Character (Hex 6D) in B0

1

Read 1410 Storage Modify 1410 Address +1 EBCDIC 11 Char to B1

QU532

Stats	On	Off
Y2	Word Mark Found	
Y3	Word Separator Found	
Y4	Move Mode	Load Mode
Y5	1 for WM	Translate Data
Y6	Stop on GM·WM	Stop on End of Storage
Y7	1410 End of Stor	2040 End of Storage

1410 End of Storage (EOM)? Yes / No

Set Y7 Regen EBCDIC 11 Character

Regen EBCDIC 11 Character

Test Mode (Y4) and Y5: 01 Translate | 00 | 11 Write 1s for WMs

01: L 1's — Error Start Logout if Errors are Enabled

00: L and Tslt — Load and Translate Strip WM

WM in EBCDIC 11 Char? Yes — Set Y2 if WM

EBCDIC 11 Character a Word Sep? Yes — Set Y3 (Word Sep)

Correct Length Record? No / Yes

M Tslt — Move and Translate Strip WM

11: M and 1's — Move and Write 1's Strip WM

WM in EBCDIC 11 Char? No / Yes — Set Y2 if WM

Correct Length Record? Yes / No

EBCDIC 11 Char GM·WM and 1410 End of Storage, Reset Y2

Emit EBCDIC Blank (Hex 40) in B1 Reg Disable Address Xltr to Read 2040 Storage

GN1 6342 QU536

QU533

Translate Char from EBCDIC 11 to EBCDIC

Read Emulator (2040) Storage Modify 2040 Address +1

Mode and Word Separator (Y4,Y3)

Load Mode — Word Sep — WM in EBCDIC 11 Char? No WM / WM

No WM: Reset Y3 Move Word Sep to D Reg

WM: Reset Y2,3 Move Word Sep to D Reg

No Word Sep

Move Mode — WM in EBCDIC 11 Char? No WM / WM

No WM: Move Character or Blank to D Reg

WM: Reset Y2 Move Generated Word Sep in B0 to D Reg

WM in EBCDIC 11 Char? WM / No WM

WM: Reset Y2 Move Generated Word Sep in B0 to D Reg

No WM: Move Character to D Reg

Store EBCDIC Character in Emulator (2040) Storage

2040 End of Storage? No / Yes

End of Operation? Yes / No

WLR 6342 QU536 Set Wrong Length Record

GN2 6342 QU536 Gather End of Operation

1

FIGURE 6343. 1410E GATHER DIAGNOSE

QU538

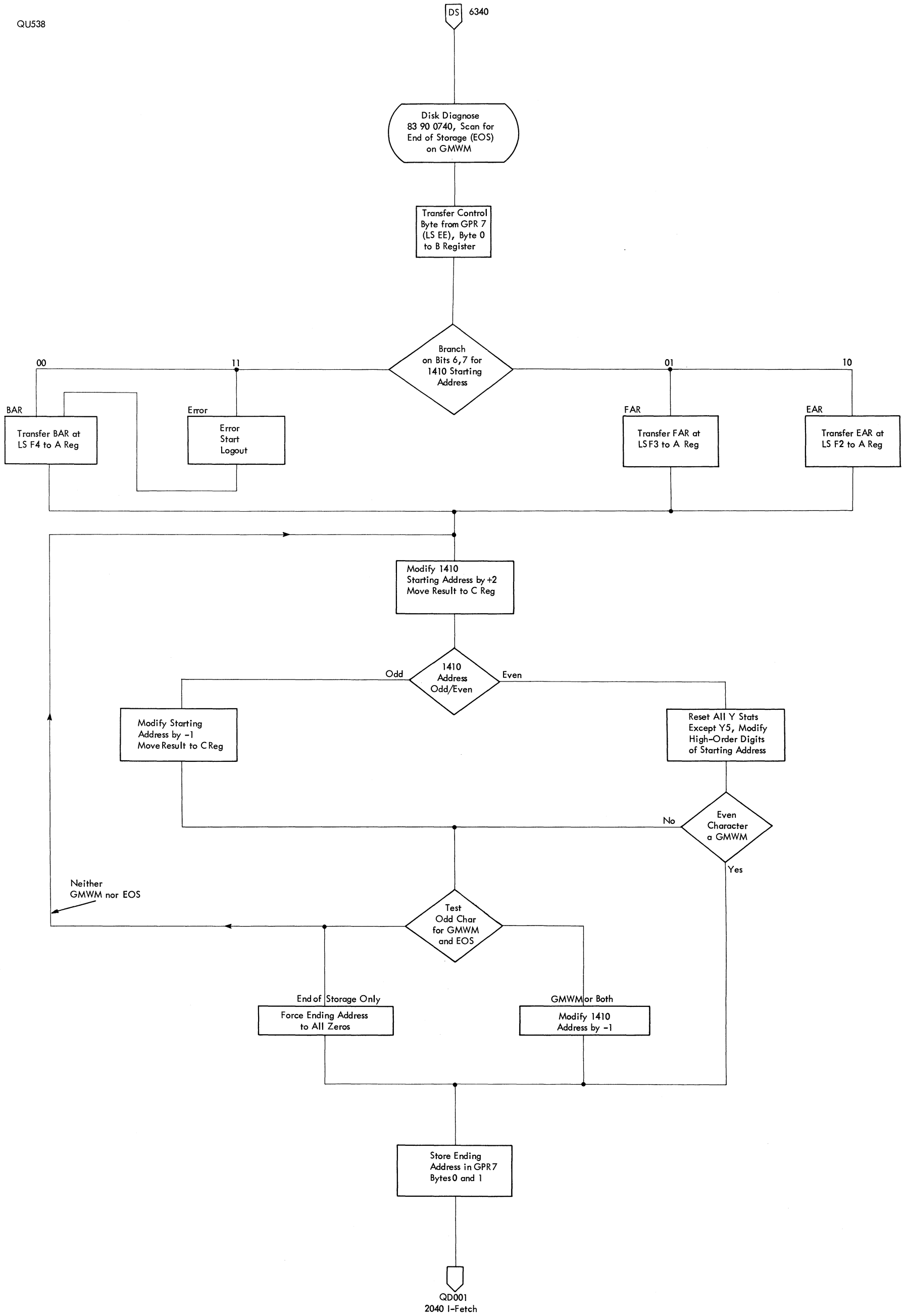


FIGURE 6344. 1410E DISK DIAGNOSE--END OF STORAGE OR GMWM SCAN

IBM **FE Supplement**

System/Unit 360 Model 40

Re: Order No. 223-2842-0,
SY22-2842-0, -1, -2, -3

This Supplement No. SY22-6827

Date May 8, 1970

Previous Supplement Nos.
S23-4036 (applies to 223-2842-0 only)
Y22-6679 } (apply to 223-2842-0
Y22-6695 } and Y22-2842-0)
Y22-6743 (applies to 223-2842-0,
Y22-2842-0, -1)
Y22-6809 (applies to 223-2842-0,
SY22-2842-0, -1, -2)

IBM FIELD ENGINEERING DIAGRAM MANUAL
SYSTEM/360 MODEL 40, 2040 PROCESSING UNIT

This supplement provides replacement pages for (or replaces figures in) the subject publication. Pages to be inserted and/or removed are:

Title Page, Preface
Figure 015
Figures 6220 and 6221, Figure 6222
X-1, X-2
X-3

A changed or added illustration is denoted by the symbol • to the left of the caption.

Summary of Amendments

This supplement provides information on the relocate feature for the 1401/1440/1460 DOS compatibility. The 1400 simulated program may reside in any one of seven (7) different 16k increments of storage as designated by the programmer for the emulator program.

Note: Please file this cover letter at the back of the manual to provide a record of changes.

IBM Corporation, Product Publications, P.O. Box 390, Poughkeepsie, N.Y. 12602

PRINTED IN U.S.A.

READER'S COMMENT FORM

IBM System/360 Model 40,
2040 Processing Unit
FEDM

SY22-2842-3

- How did you use this publication?

  As a reference source ☐
  As a classroom text ☐
  As ..... ☐

- Based on your own experience, rate this publication...

	.....	.....	.....	.....	.....
As a reference source:	Very Good	Good	Fair	Poor	Very Poor
As a text:	Very Good	Good	Fair	Poor	Very Poor

- What is your occupation? .....

- We would appreciate your other comments; please give specific page and line references where appropriate. If you wish a reply, be sure to include your name and address.

- Thank you for your cooperation. No postage necessary if mailed in the U.S.A.

READER'S COMMENT FORM

IBM System/360 Model 40,
2040 Processing Unit
FEDM

SY22-2842-3

- How did you use this publication?

  As a reference source ☐
  As a classroom text ☐
  As ..... ☐

- Based on your own experience, rate this publication...

	.....	.....	.....	.....	.....
As a reference source:	Very Good	Good	Fair	Poor	Very Poor
As a text:	Very Good	Good	Fair	Poor	Very Poor

- What is your occupation? .....

- We would appreciate your other comments; please give specific page and line references where appropriate. If you wish a reply, be sure to include your name and address.

- Thank you for your cooperation. No postage necessary if mailed in the U.S.A.

YOUR COMMENTS, PLEASE . . . . . .

Your answers to the questions on the back of this form, together with your comments, will help us produce better publications for your use. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Note: Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

fold fold

FIRST CLASS
PERMIT NO. 419
POUGHKEEPSIE, N.Y.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . . . . .

IBM CORPORATION
P.O. BOX 390
POUGHKEEPSIE, N.Y. 12602

ATTENTION: FE MANUALS, DEPT. B96

fold fold

IBM

International Business Machines Corporation
Field Engineering Division
112 East Post Road, White Plains, N.Y. 10601

cut along this line

YOUR COMMENTS, PLEASE . . . . . .

Your answers to the questions on the back of this form, together with your comments, will help us produce better publications for your use. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Note: Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

fold fold

FIRST CLASS
PERMIT NO. 419
POUGHKEEPSIE, N.Y.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . . . . .

IBM CORPORATION
P.O. BOX 390
POUGHKEEPSIE, N.Y. 12602

ATTENTION: FE MANUALS, DEPT. B96

fold fold

IBM

International Business Machines Corporation
Field Engineering Division
112 East Post Road, White Plains, N.Y. 10601

S/360 Model 40 2040 Processing Unit (FEDM) Printed in U.S.A. SY22-2842-3

IBM®
International Business Machines Corporation
Field Engineering Division
112 East Post Road, White Plains, N. Y. 10601

SY22-2842-3

cut here

System

System
Maintenance
Library