



# **IBM** Field Engineering Diagram Manual

**System/360 Model 40  
2040 Processing Unit**

## 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	Title
<b>DATA FLOW CHARTS</b>	
011	Selector Channel Data Flow
012	CPU Data Flow
013	Microprogram Data Flow
014	CPU Microprogram Flow Chart
015	ROS Control Word
101	Multiplex and MS Unit Data and Control
<b>SIMPLIFIED LOGIC DIAGRAMS (SLD)</b>	
501	LSAR Parity Generation
502	Clock Control (SP)
504	Main Storage Control and Timing Circuits (2 Sheets)
505	Function and Control Registers
506	Decimal Filler
507	Decimal Correction
508	Carry Latches
509	Selector Channel Controls (2 Sheets)
510	Mid-Pac Power Supply Wiring Diagram (2 Sheets)
510A	2040 Mid-Pac Wall Frame Wiring Diagram
511	2.5 kHz HF Power Supply Wiring Diagram
512	Multiplex Channel Controls
513	Main Storage X-Dimension Drive
<b>CONDENSED LOGIC FLOW CHARTS (CLF)</b>	
599	How to Use Flow Charts
600	Instruction Matrix
601	Instruction Fetch Microprogram
602	2nd Level Instruction Fetch, RX Fixed Point
603	2nd Level Instruction Fetch, RX Floating-Point
604	2nd Level Instruction Fetch, RS and S1 Operations
605	2nd Level Instruction Fetch, SS Logical
606	2nd Level Instruction Fetch, SS Decimal
607	Machine Status at 1st and 2nd Level Function Branches
608	Branch and Link
609	Set and Insert Storage Key
610	Convert Decimal to Binary
611	RR Fixed Point Sign Operation
612	Branch on Count
613	Convert Binary to Decimal
614	Set Program Mask
615	RR and RX Fixed Point Arithmetic and Logic
616	RX Fixed Point Add and Subtract
617	RX Compare Algebraic
618	Branch on Condition
620	RR and RX Fixed Point Multiply
621	RR and RX Fixed Point Multiply, Notes
622	RR and RX Fixed Point Multiply, Detail of Loops
623	Fixed Point Divide Initialization
624	Fixed Point Divide Loop
625	RR Floating-Point Sign Operations
626	RR and RX Floating-Point Operation (2 Sheets)
627	Floating-Point Load and Store
628	Floating-Point Multiply/Divide Initialization (2 Sheets)
629	Floating-Point Multiply Loop
630	Floating-Point Divide Loop
631	Test Under Mask
632	Branch on Index
633	Set System Mask
634	RS Load and Store Multiple
635	Shifts
636	SI Operations, AND, OR, EXOR, MOVE
637	Read Direct and Write Direct
638	Load PSW (2 Sheets)
639	Diagnose Instruction
641	SS Translate
642	SS Translate and Test
643	Edit, Edit and Mark (2 Sheets)
644	SS Edit, Refill
647	SS Logical Operations, Move Zone and Numeric
648	SS Logical Operations, Move Complete
649	SS Logical Operations, Compare
650	SS Decimal Divide
651	Decimal Divide Example
652	Decimal Divide Add/Subtract Paths
653	SS Decimal Multiply
654	Decimal Pack
655	Decimal Unpack
656	Decimal Move with Offset
657	General Flow Chart for Decimal Add Sub Compare
658	SS Decimal Load and Process Operand 1
659	SS Decimal Load Zero and Add Entry
660	SS Decimal Load Operand 2 and Process
661	SS Decimal Terminate
662	SS Decimal Compare
665	Start I/O Instruction (Multiplex Channel)
666	I/O Codes, Common Decoding; Test Channel and Mpx Halt I/O (2 Sheets)
667	Start I/O Microprogram Mpx Channel (2 Sheets)
668	Test I/O Multiplex Channel Microprogram
669	Multiplex Channel Microprogram (4 Sheets)
670	Multiplex Channel Status
671	I/O Interrupts and Update Timer Microprogram
674	Selector Channel Status
675	Selector Channel I/O Instructions Microprogram (4 Sheets)
686	Store PSW - General Flow

MALFUNCTION ANALYSIS PROCEDURES (MAP)

901 Interpret Errors (2 Sheets)  
 906 Control Check  
 907 Early Check  
 908 Late Check  
 911 Read Only Storage (3 Sheets)  
 912 Local Storage  
 913 Storage Protect  
 914 Main Storage (64K MAP)  
 915 Multiplex Channel  
 916 Selector Channel  
 917 Mid-Pac Power Supply  
 918 2.5 kc HF Power Supply

1401/1460 COMPATIBILITY FEATURE FLOW CHARTS

6200 1401 Instruction Fetch  
 6200A 1401 Instruction Fetch  
 6200B 1401 Instruction Fetch  
 6201 1401 N Operation  
 6202 1401 Add and Subtract  
 6202A 1401 Add and Subtract  
 6203 1401 Compare  
 6203A 1401 Compare  
 6204 Store 1401 AAR or BAR  
 6205 Multiway Branch  
 6206 1401 Increment-Decrement  
 6207 Scatter--Gather  
 6207A Scatter--Gather  
 6207B Scatter (DOS Compatibility Feature) (2 Sheets)  
 6207C Gather (DOS Compatibility Feature)  
 6208 1401 Multiply and Divide  
 6208A 1401 Multiply and Divide  
 6208B 1401 Multiply and Divide  
 6209 1401 Move, Load, Zero and Add, Zero and Subtract  
 6209A 1401 Move, Load, Zero and Add, Zero and Subtract  
 6209B 1401 Move, Load, Zero and Add, Zero and Subtract  
 6210 1401 I/O M, L, U Operations  
 6211 1401 Unit Record Operations  
 6212 1401 Carriage Control and Stacker Select  
 6213 1401 Address Modify  
 6213A 1401 Address Modify  
 6214 1401 Set Word Mark, Clear Word Mark, Clear Storage,  
 and Special Clears  
 6215 1401 Move Characters and Suppress Zeros  
 6216 1401 1 and 2Byte Data Service  
 6217 Set Selector Channel to 1401 Mode  
 6218 1401 Tape Operations  
 6219 1401 Branch if: Word Mark or Zone, Bit Equal or  
 Character Equal  
 6220 1401 Emulator Program Entry  
 6221 1401 Diagnose  
 6222 1401 Branch Tests  
 6223 1401 Move and Binary Decode  
 6224 1401 Move and Binary Code  
 6225 1401 Read and Punch Column Binary  
 6226 1401 Index Factor Fetch  
 6227 1401 Index Add

1410/7010 COMPATIBILITY FEATURE FLOW CHARTS (CLF)

6300 1410/7010 Operation Codes in EBCDIC-II  
 6301 1410E Instruction Fetch-Overall  
 6302 1410E Instruction Fetch Start  
 6303 1410E X Control Field Readout  
 6304 1410E Address Readout  
 6305 1410E Instruction Fetch Ending  
 6306 1410E NOP and Non-Interruptible Op Codes  
 6307 1410E Chaining  
 6308 1410E Indexing (Two Sheets)  
 6309 1410E Branch on Channel Status  
 6310 1410E Priority Test and Branch, Branch on Internal Indicator  
 6311 1410E Branch if: Character, WM/Zone, or Bit Equal  
 6312 1410E Store Address Register (Two Sheets)  
 7010E Store and Restore Status  
 1410E Set/Clear WM and Clear Storage  
 6315 1410E Table Lookup  
 6316 1410E Add, Subtract, Multiply, and Divide  
 6317 1410E Add, Subtract-Initial Loop  
 6318 1410E Add, Subtract-Main Loop  
 6319 1410E Add, Subtract-Special Loop and Ending  
 6320 1410E Multiply-First Scan  
 6321 1410E Multiply Loop  
 6322 1410E Divide Initial Loop  
 6323 1410E Divide Loop and Ending  
 6324 1410E Data Move, Zero Add/Subtract, and Compare  
 6325 1410E Zero Add/Subtract-Initial Loop  
 6326 1410E Zero Add/Subtract-Main and Special Loops  
 6327 1410E Data Move Minus Scan-Initial Loop  
 6328 1410E Data Move Minus Scan, Zero Add/Subtract A Cycles  
 6329 1410E Data Move Plus Scan-Initial Loop  
 6330 1410E Data Move Plus Scan-Main Loop  
 6331 1410E Compare-Initial Loop  
 6332 1410E Compare-Main Loop  
 6333 1410E Compare Character and Ending  
 6334 1410E Unit Control, I/O Move/Load-Initial Loop  
 6335 1410E X-control Field Translation  
 6336 1410E I/O Unit Selection  
 6337 1410E One-Byte Data Service  
 6338 1410E Two-Byte Data Service  
 6339 1410E Forms Control and Stacker Select  
 6340 1410E Diagnose Instructions, I-Fetch Linkages  
 6341 1410E Edit Diagnose  
 6342 1410E Scatter/Gather Diagnose  
 6343 1410E Gather Diagnose  
 6344 1410E Disk Diagnose--End of Storage or GMWM Scan

INDEX

X-1  
 X-2  
 X-3



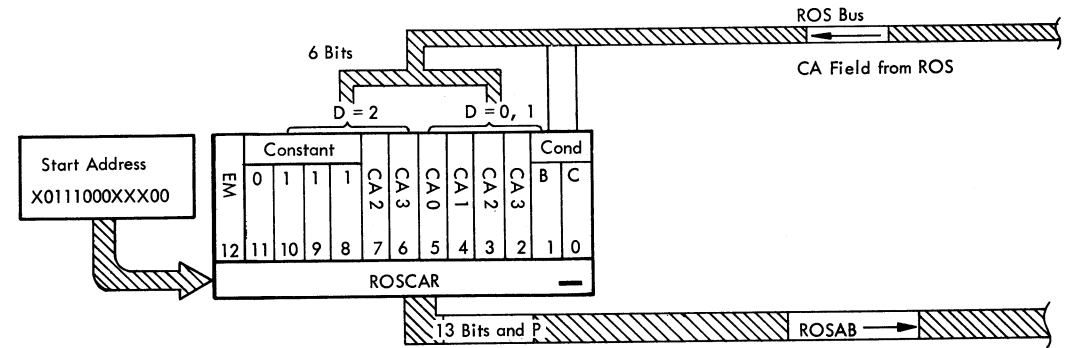
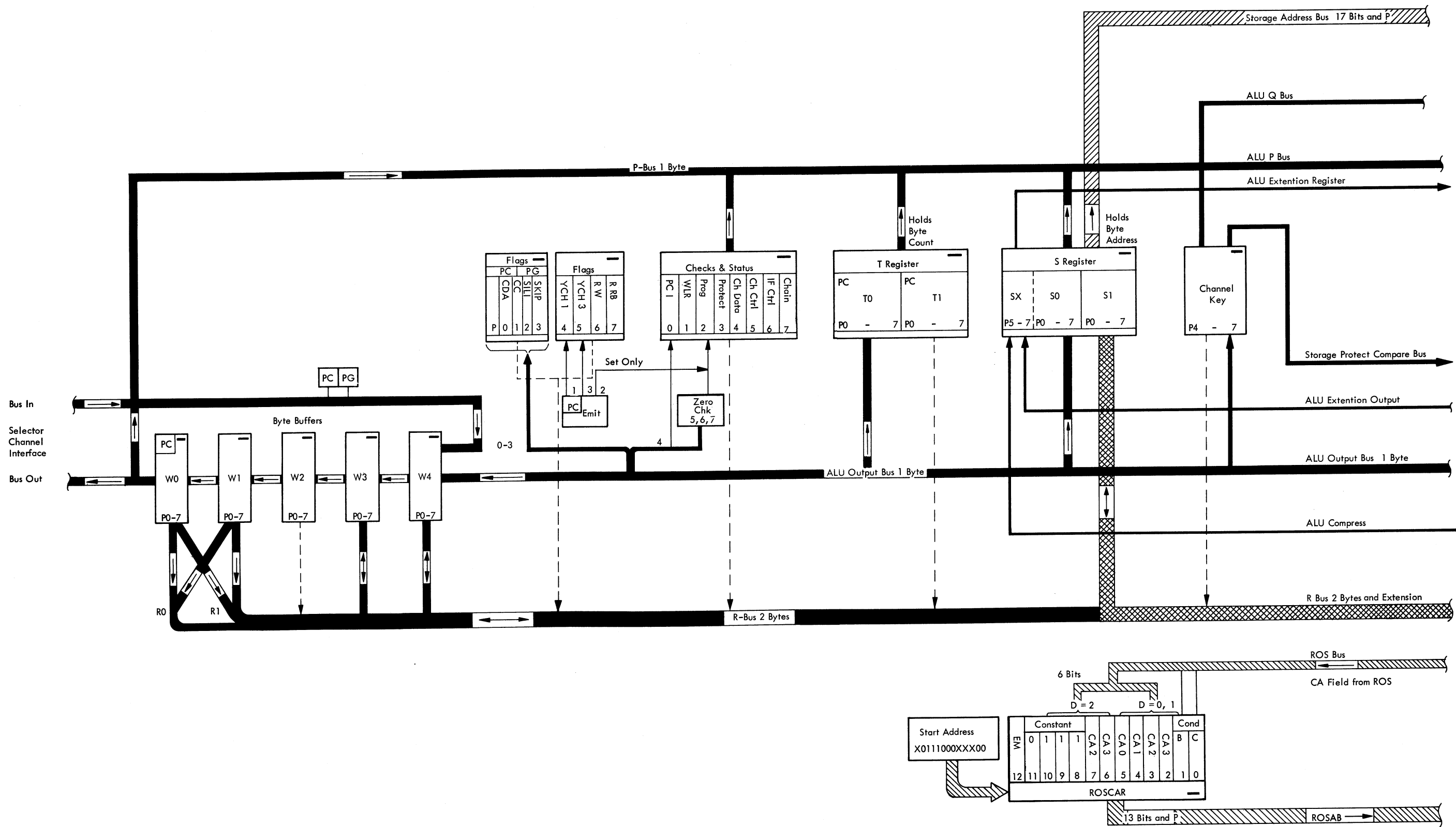


FIGURE 011. SELECTOR CHANNEL DATA FLOW

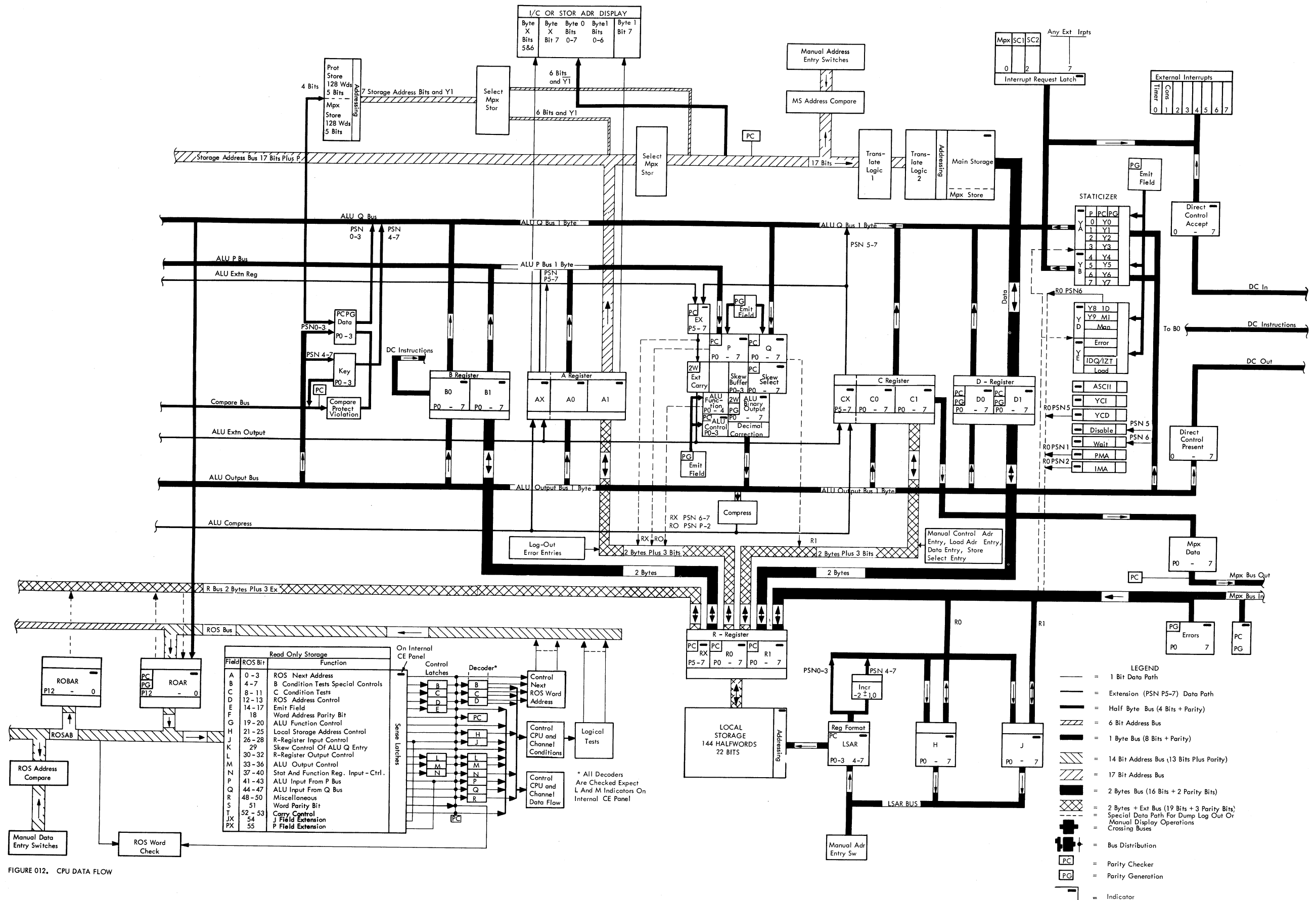


FIGURE 012. CPU DATA FLOW

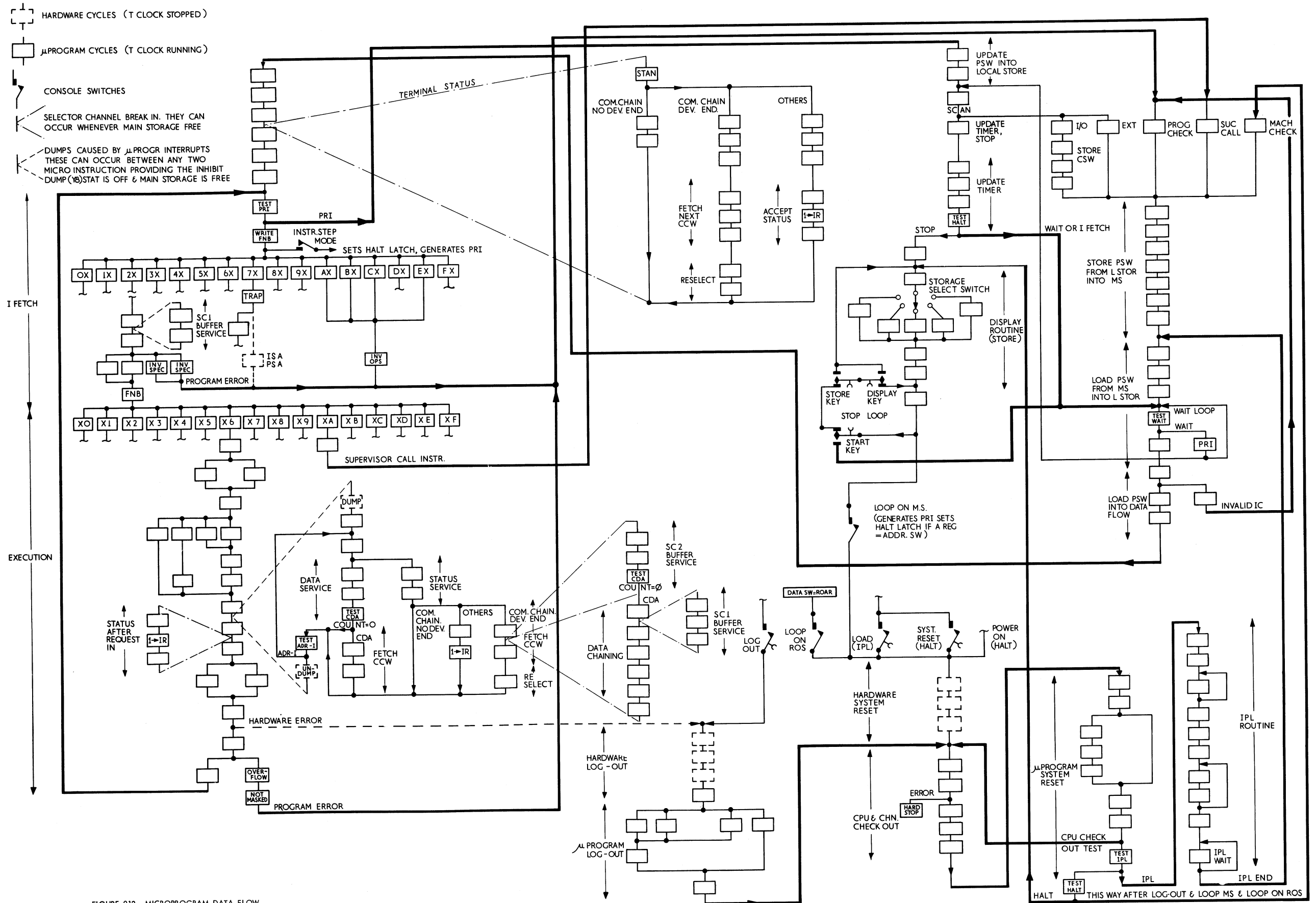


FIGURE 013. MICROPROGRAM DATA FLOW

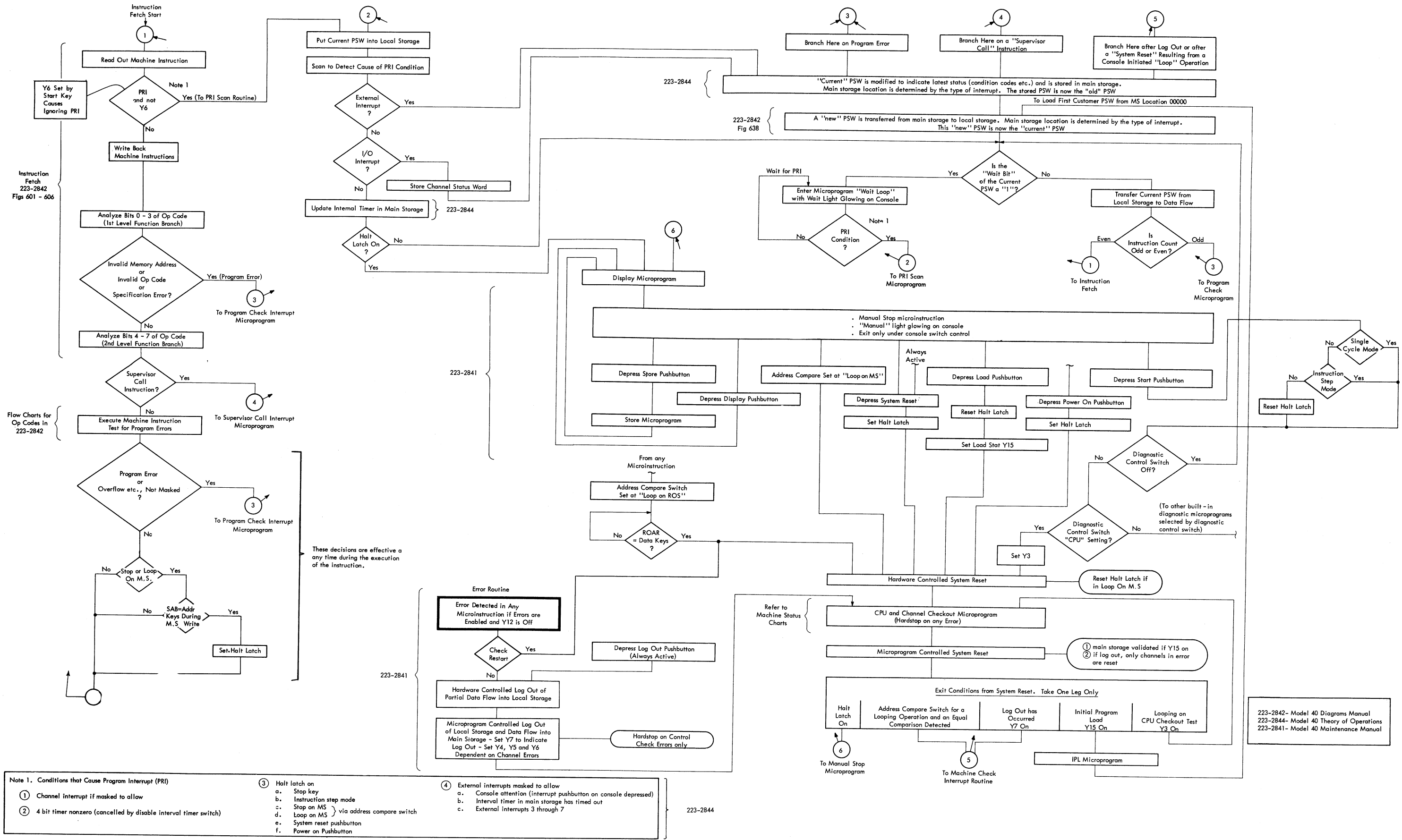
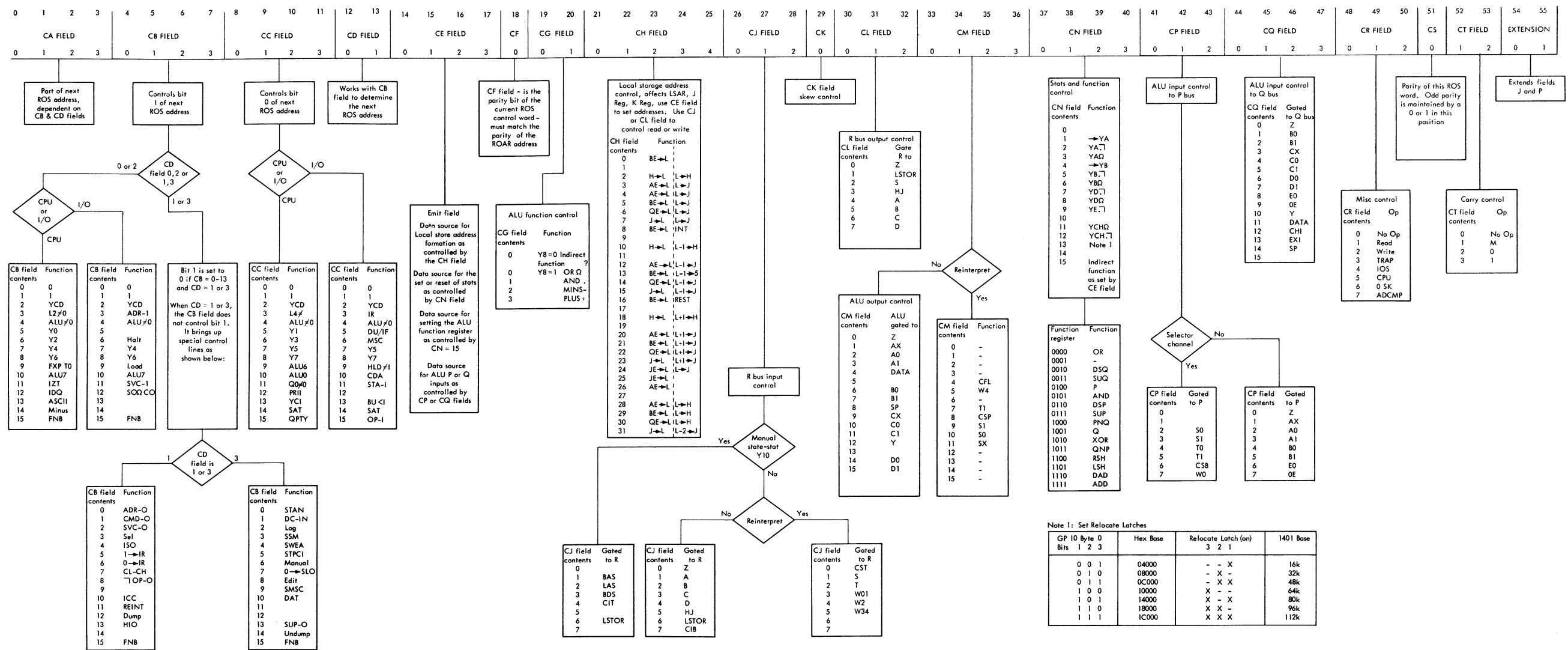


FIGURE 014. CPU MICROPROGRAM FLOW CHART (CHANNEL DATA SERVICE NOT SHOWN)



● FIGURE 015. ROS CONTROL WORD

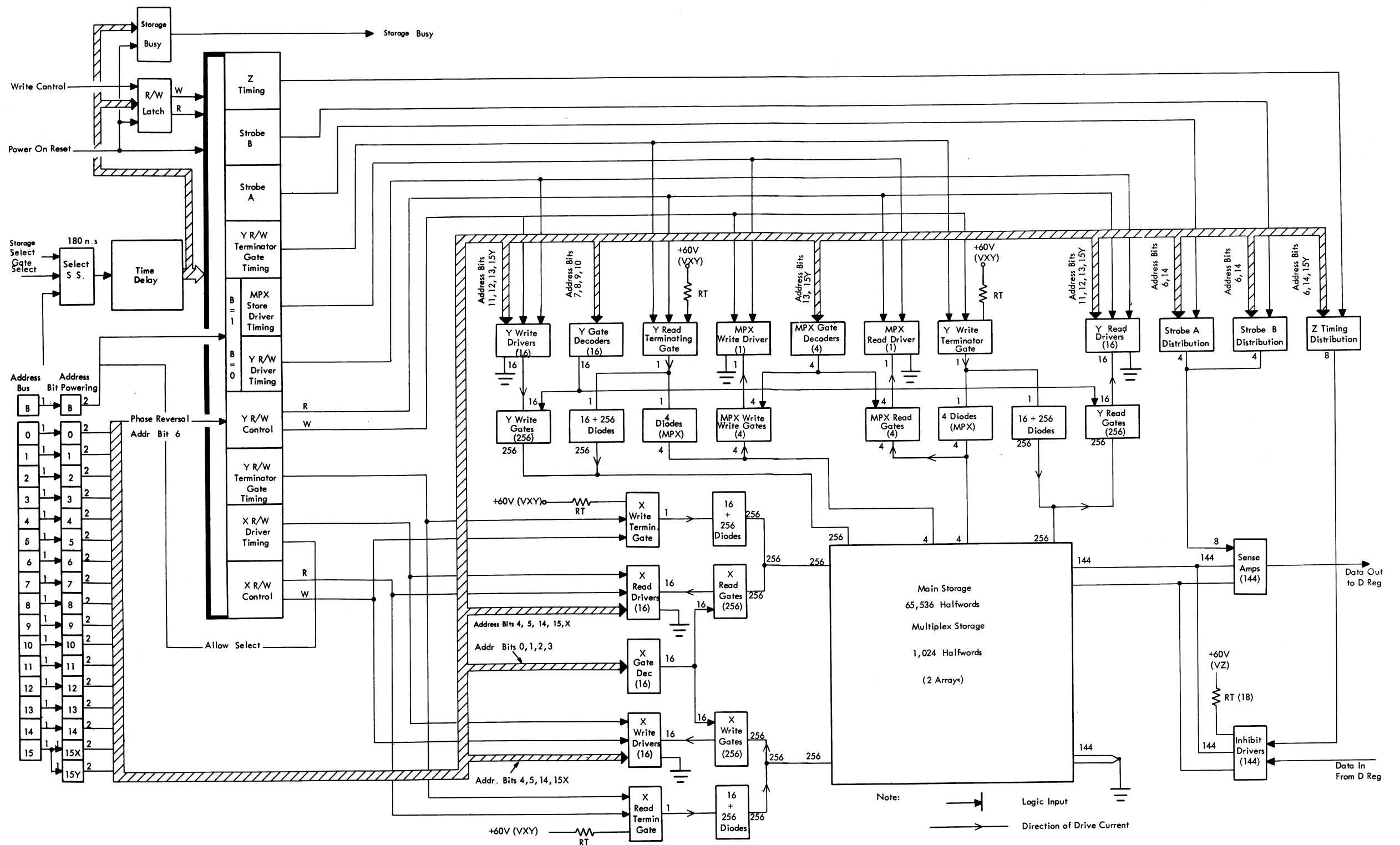


FIGURE 101. MULTIPLEX AND MS UNIT DATA AND CONTROL



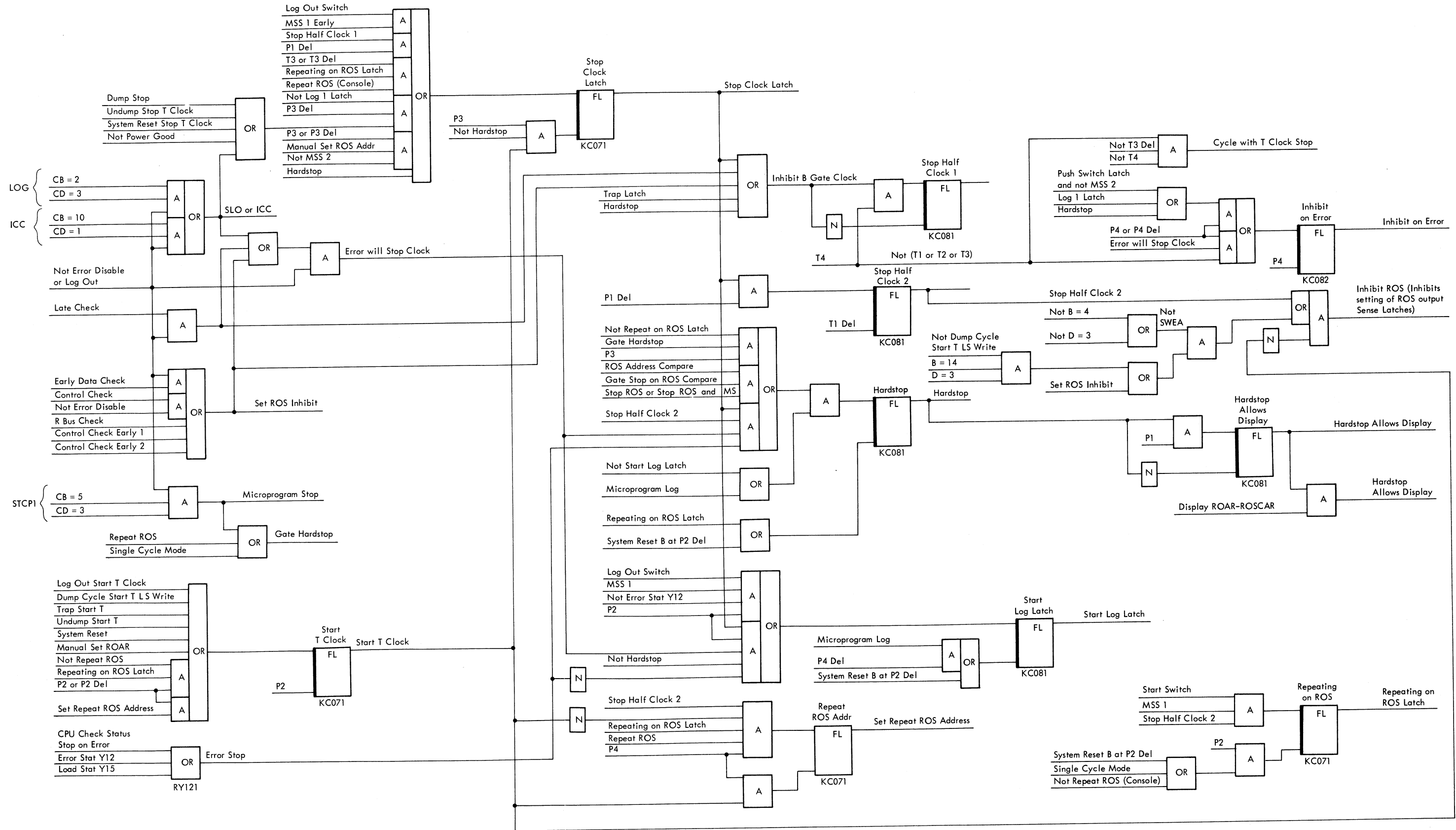


FIGURE 502. CLOCK CONTROL (KC071-KC081)



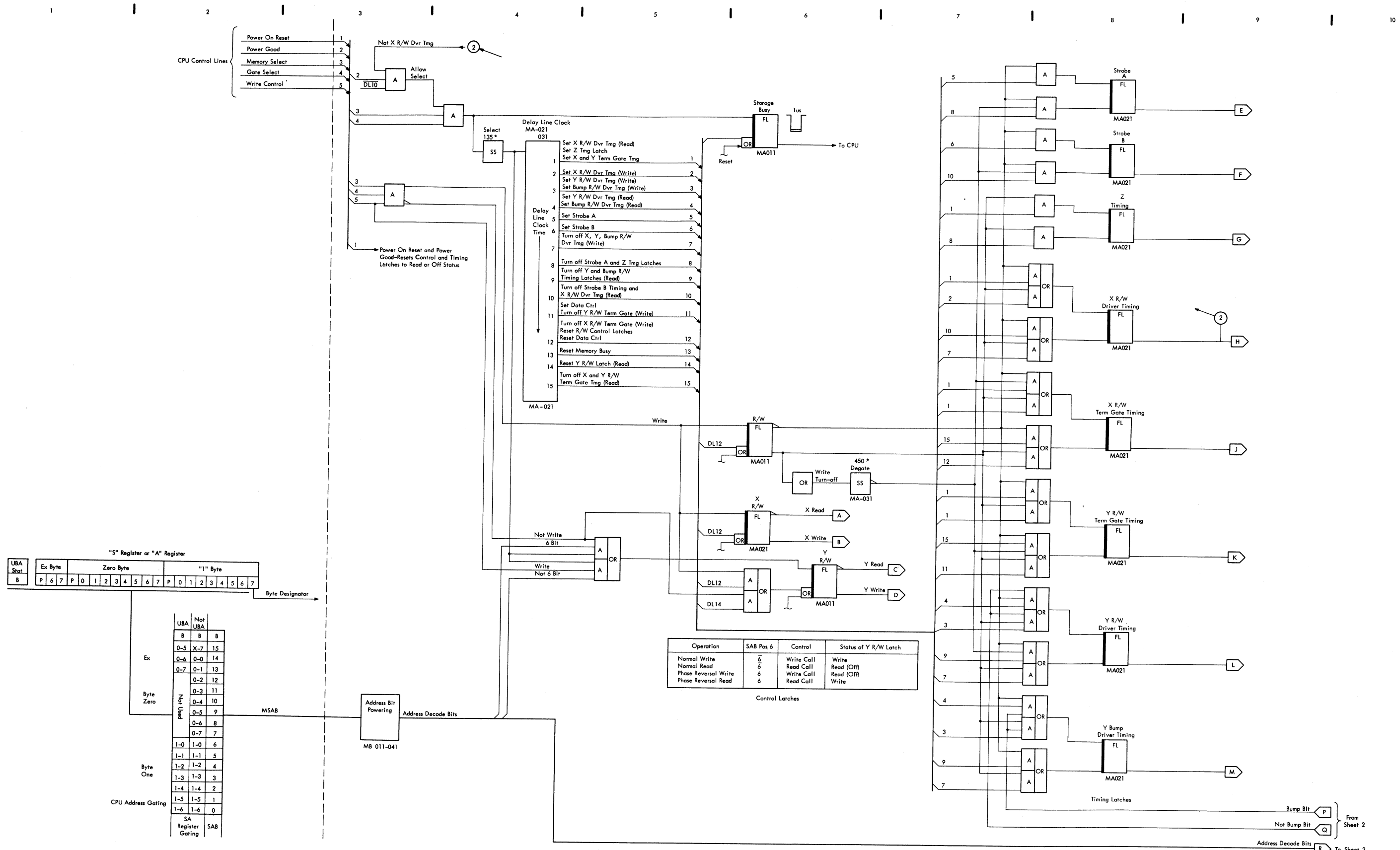


FIGURE 504. MAIN STORAGE CONTROL AND TIMING CIRCUITS (SHEET 1 of 2)

Bump Bit P  
 Not Bump Bit Q  
 Address Decode Bits R To Sheet 2

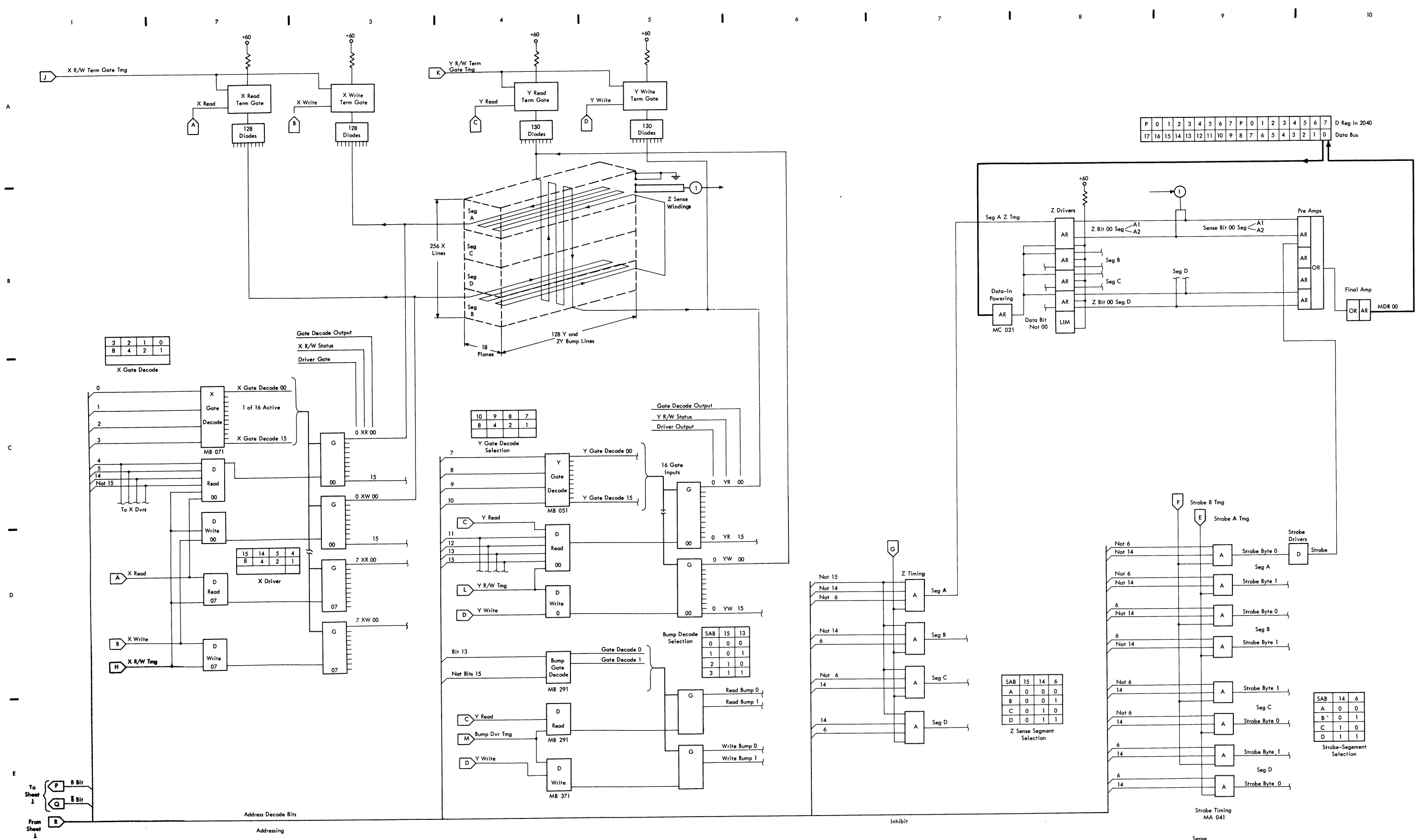
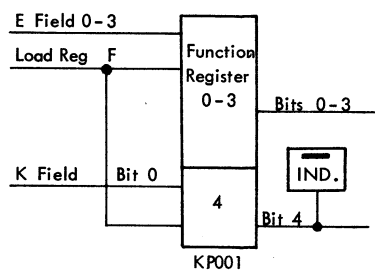
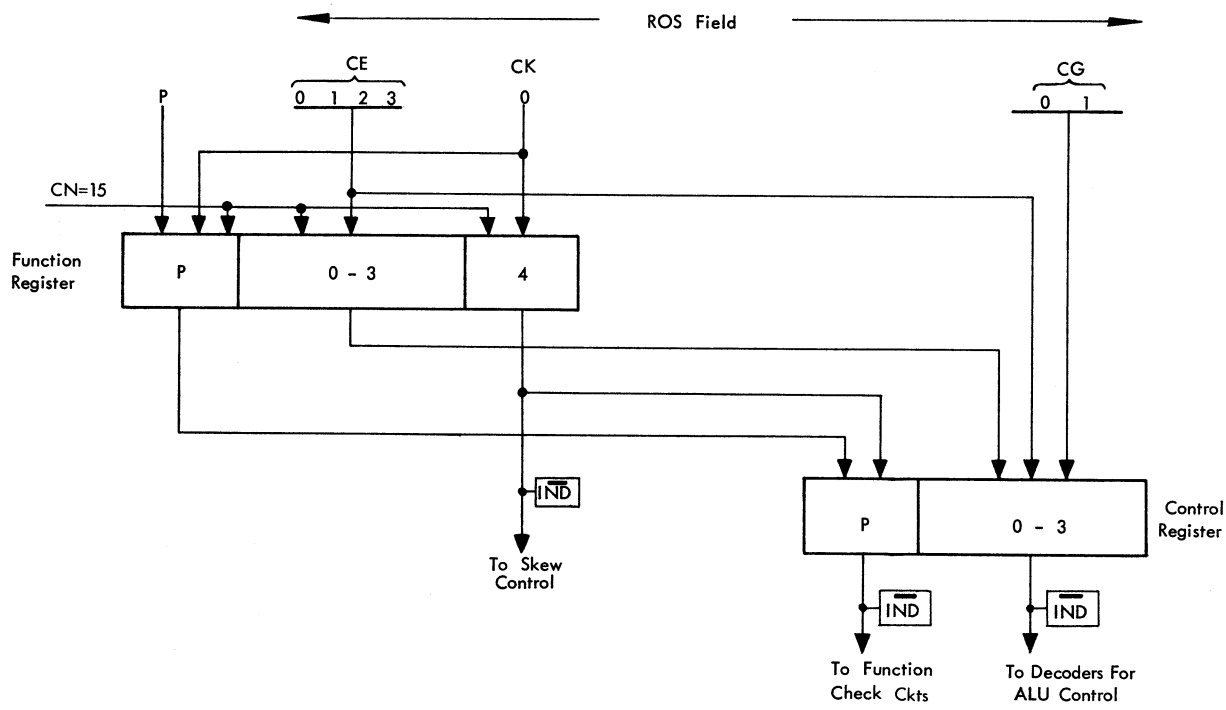
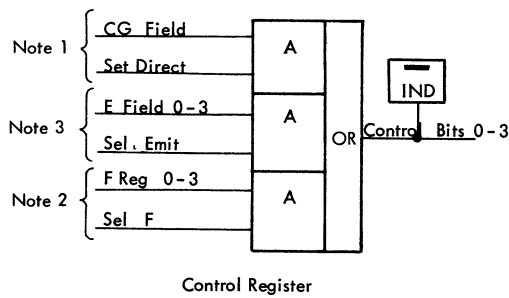


FIGURE 504. MAIN STORAGE CONTROL AND TIMING CIRCUITS (SHEET 2 OF 2)



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



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

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

Boolean Expression

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

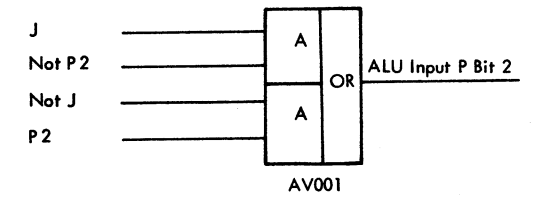
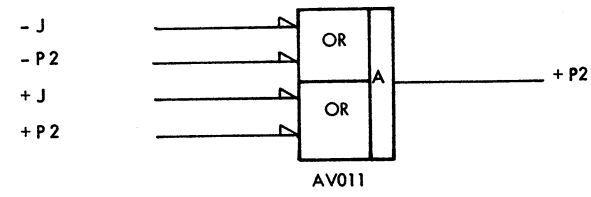
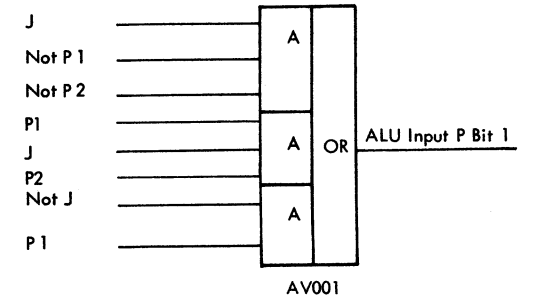
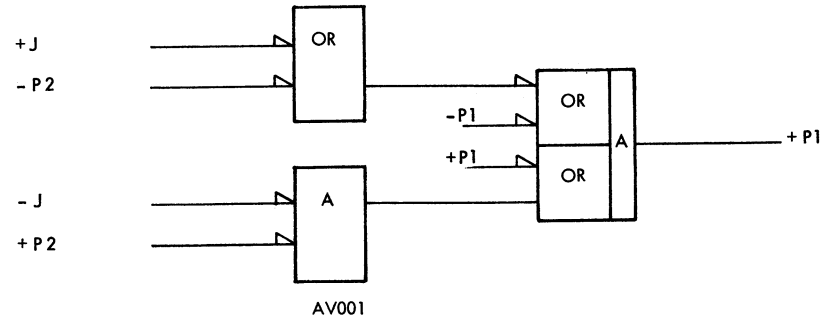
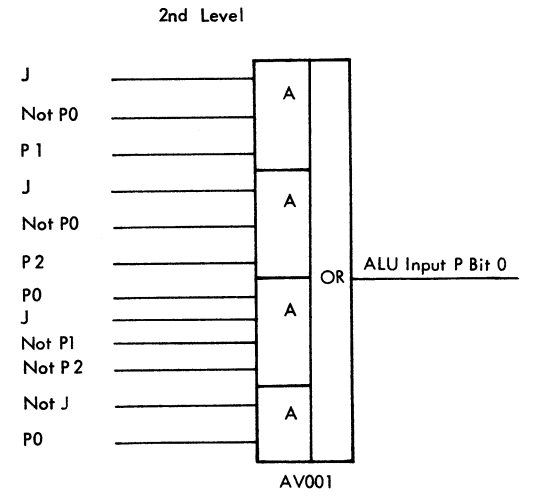
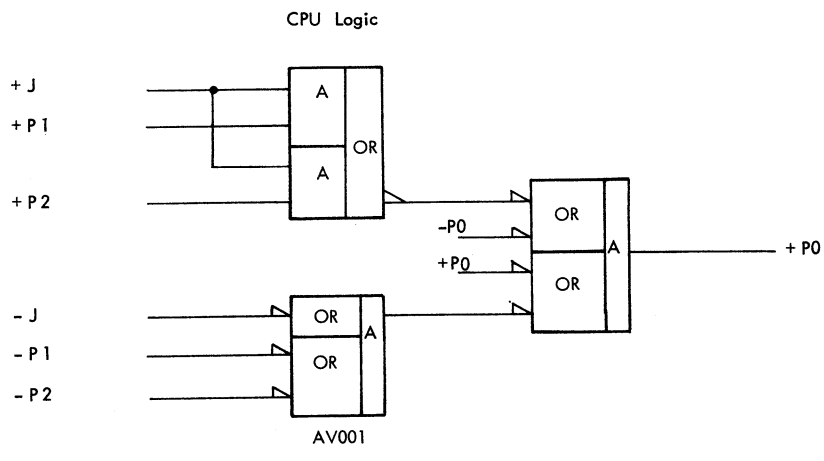
Note 2 Select F Register If CG = 0 (and Y8=0)  
And CN ≠ 15

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

	CG Field Bits		Control Bits			
	0	1	0	1	2	3
P Or Q	0	0	0	0	0	0
P And Q	0	1	0	1	0	1
P Minus Q	1	0	0	0	1	1
P Plus Q	1	1	1	1	1	1

To Change Function Register & Use In Same Cycle,  
Both Control And Function Registers 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.

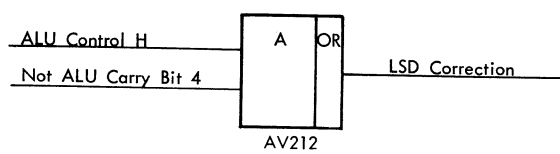
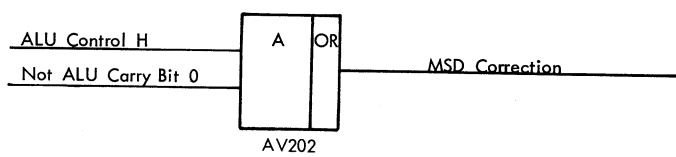
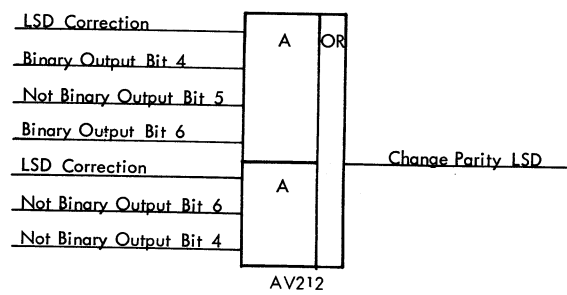
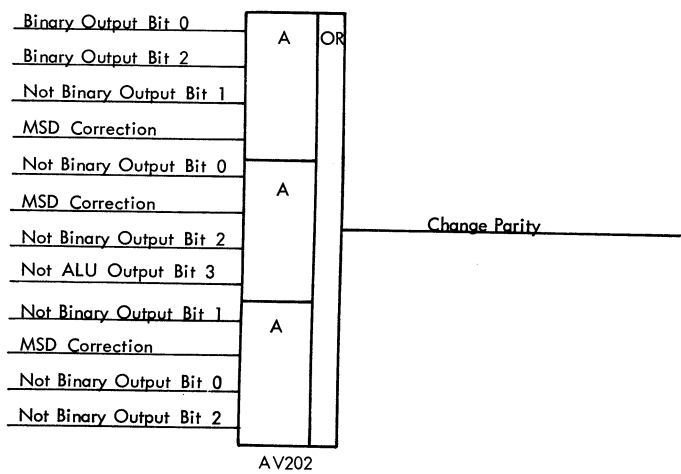
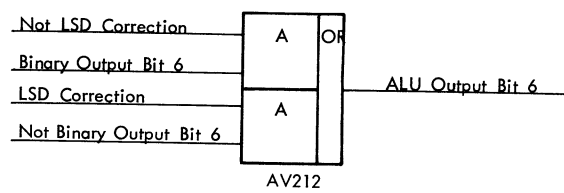
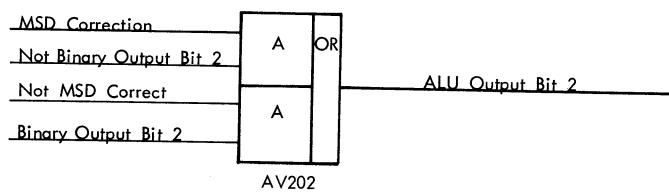
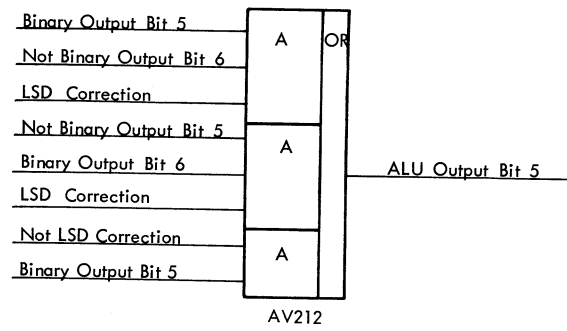
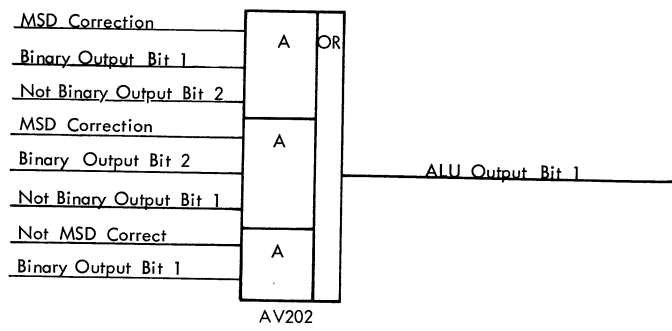
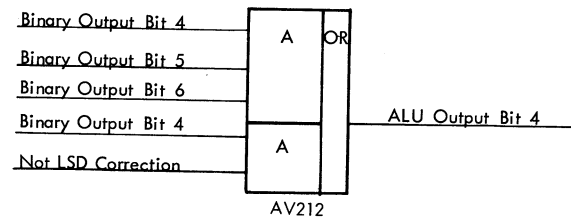
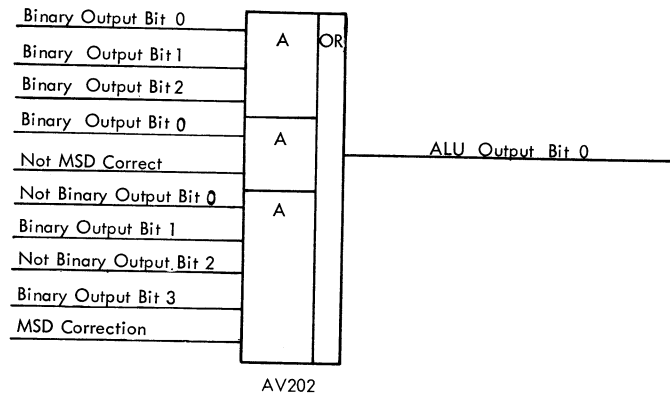
FIGURE 505. FUNCTION & CONTROL REGISTERS



	Boolean Expression	
	Decimal Add Operation	Normal Operation
Bit Position 0	$\overline{J}P0P1$ or $\overline{J}P0P2$ or $\overline{J}P0P1P2$ or $\overline{J}P0$	$\overline{J}P0$
Bit Position 1	$\overline{J}P1P2$ or $\overline{J}P1P2$	or $\overline{J}P1$
Bit Position 2 (Inversion)	$\overline{J}P2$	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	$0 \cdot 1 \cdot 2 + \bar{0} \cdot 1 \cdot \bar{2} \cdot 3$
Bit 1	$1 \cdot \bar{2} + \bar{1} \cdot 2$
Bit 2	Invert
Bit 3	No Change (Not Shown)

Boolean Expression LSD

Bit 4	$4 \cdot 5 \cdot 6$
Bit 5	$5 \cdot \bar{6} + \bar{5} \cdot 6$
Bit 6	Invert
Bit 7	No Change (Not Shown)

FIGURE 507 DECIMAL CORRECTION

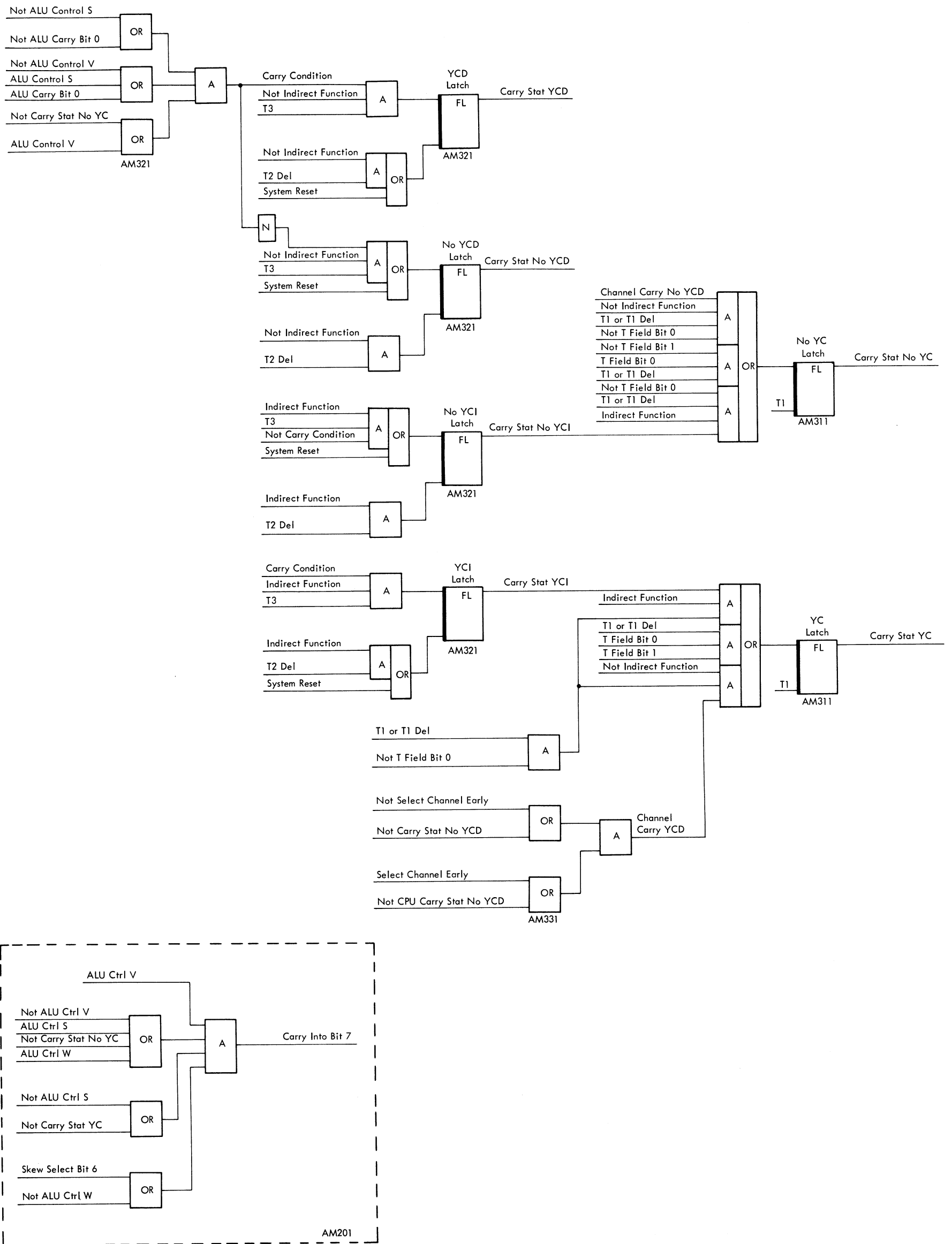


FIGURE 508. CARRY LATCHES

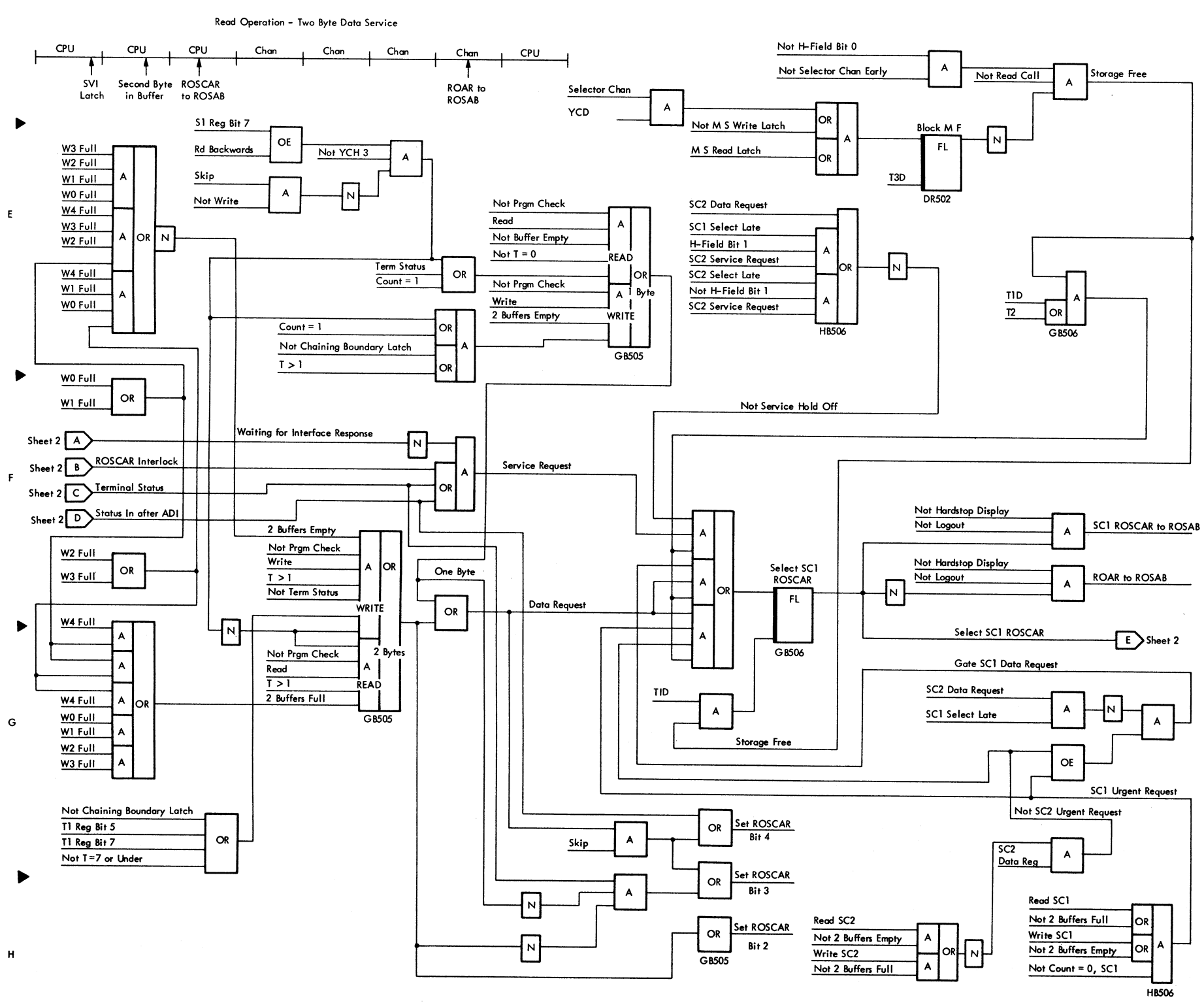
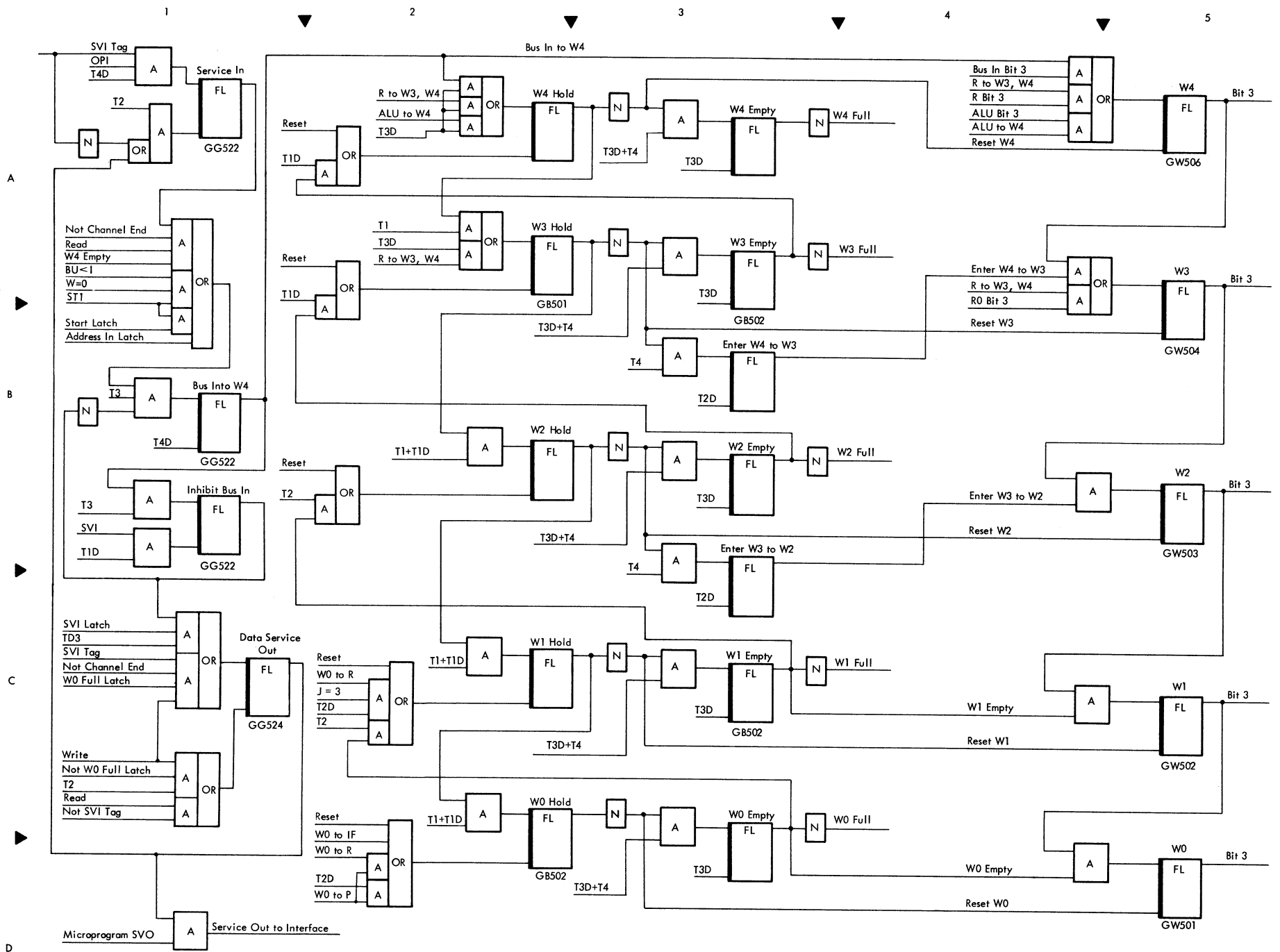


FIGURE 509. SELECTOR CHANNEL CONTROLS (SHEET 1 OF 2)

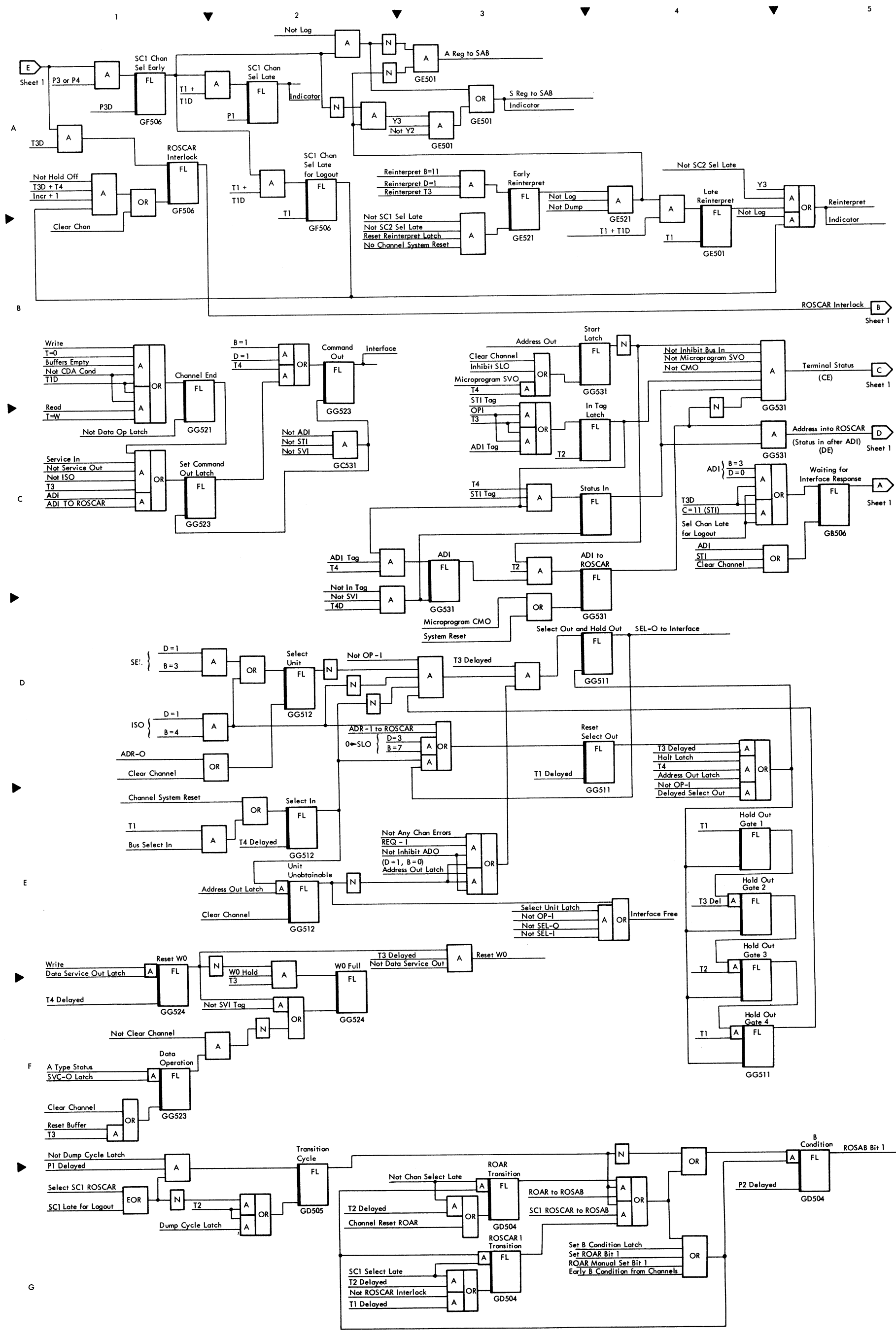


FIGURE 509. SELECTOR CHANNEL CONTROLS (SHEET 2 OF 2)



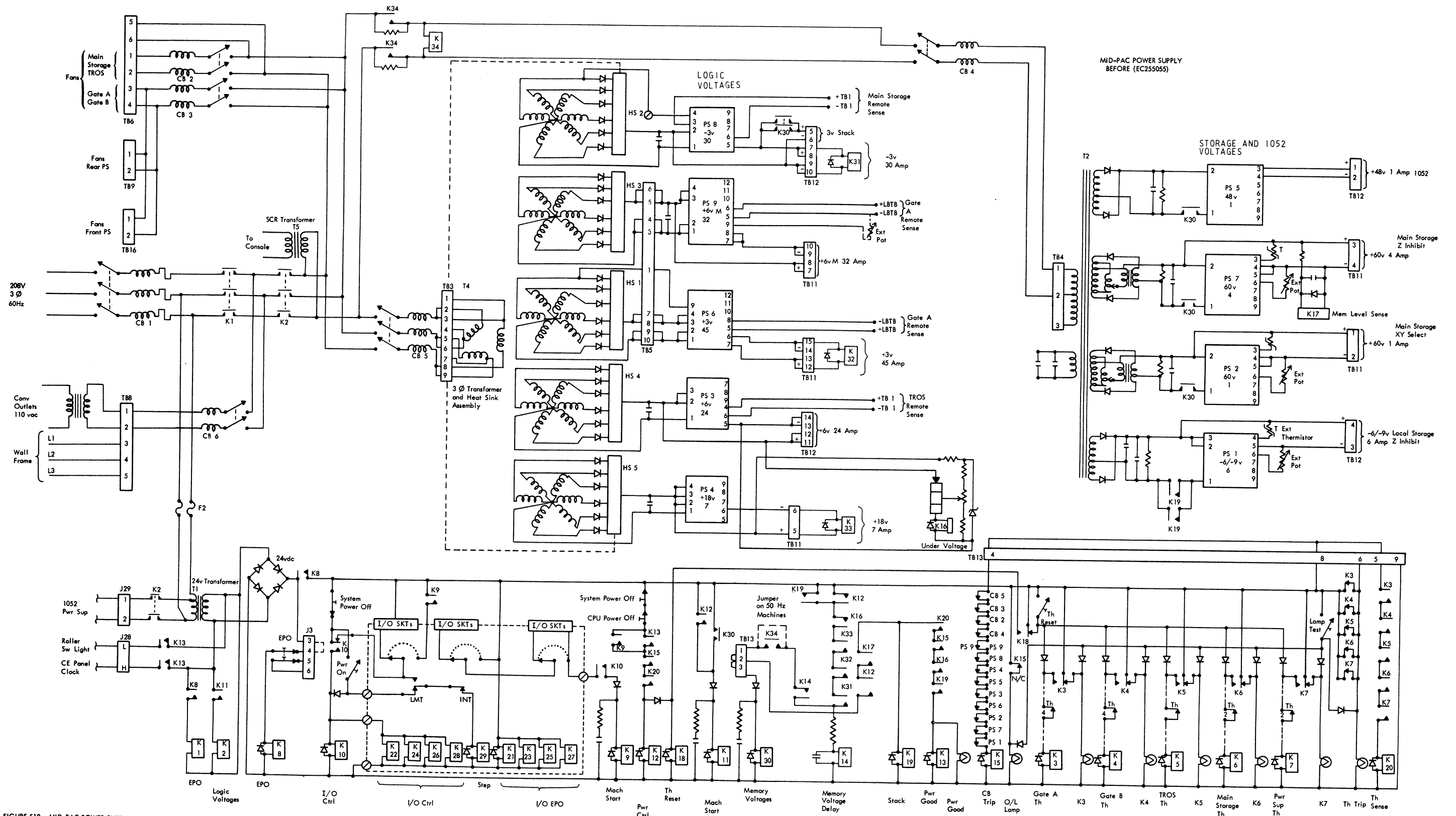


FIGURE 510. MID-PAC POWER SUPPLY WIRING DIAGRAM (SHEET 1 OF 2)

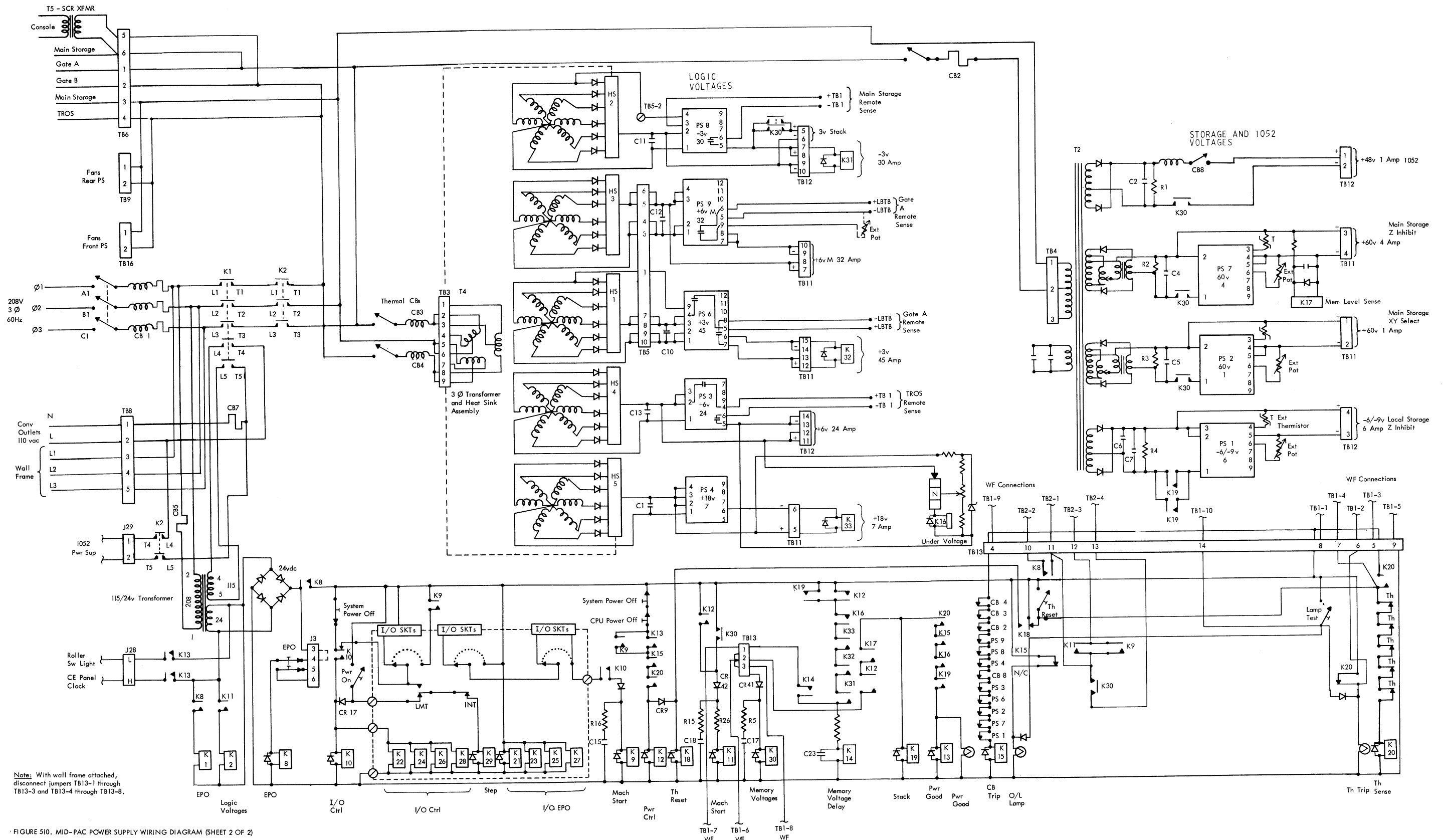
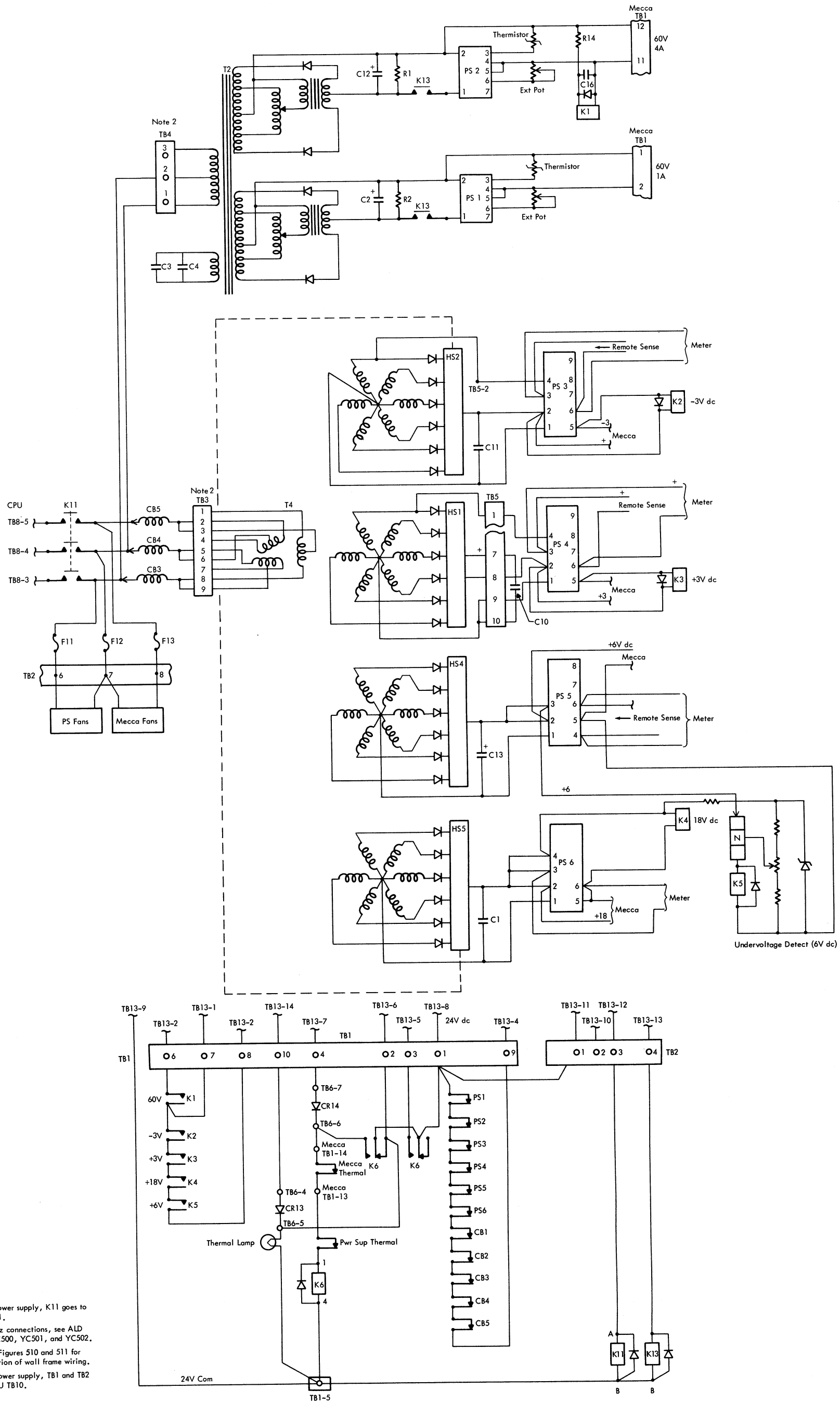


FIGURE 510. MID-PAC POWER SUPPLY WIRING DIAGRAM (SHEET 2 OF 2)



**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

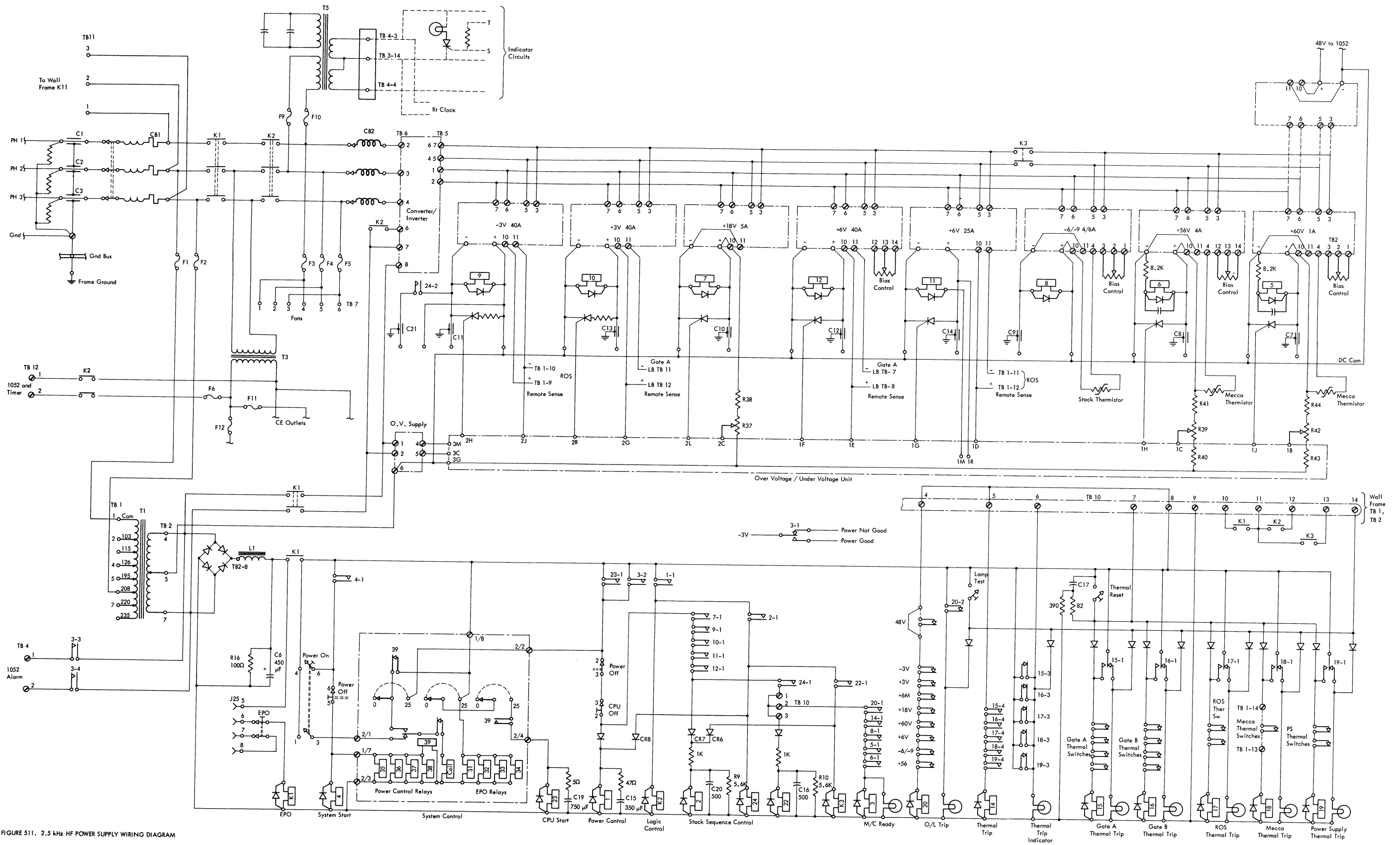
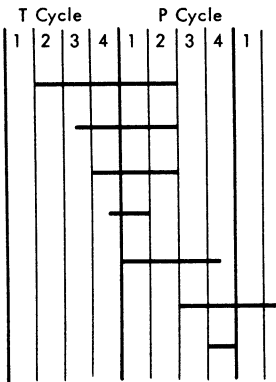
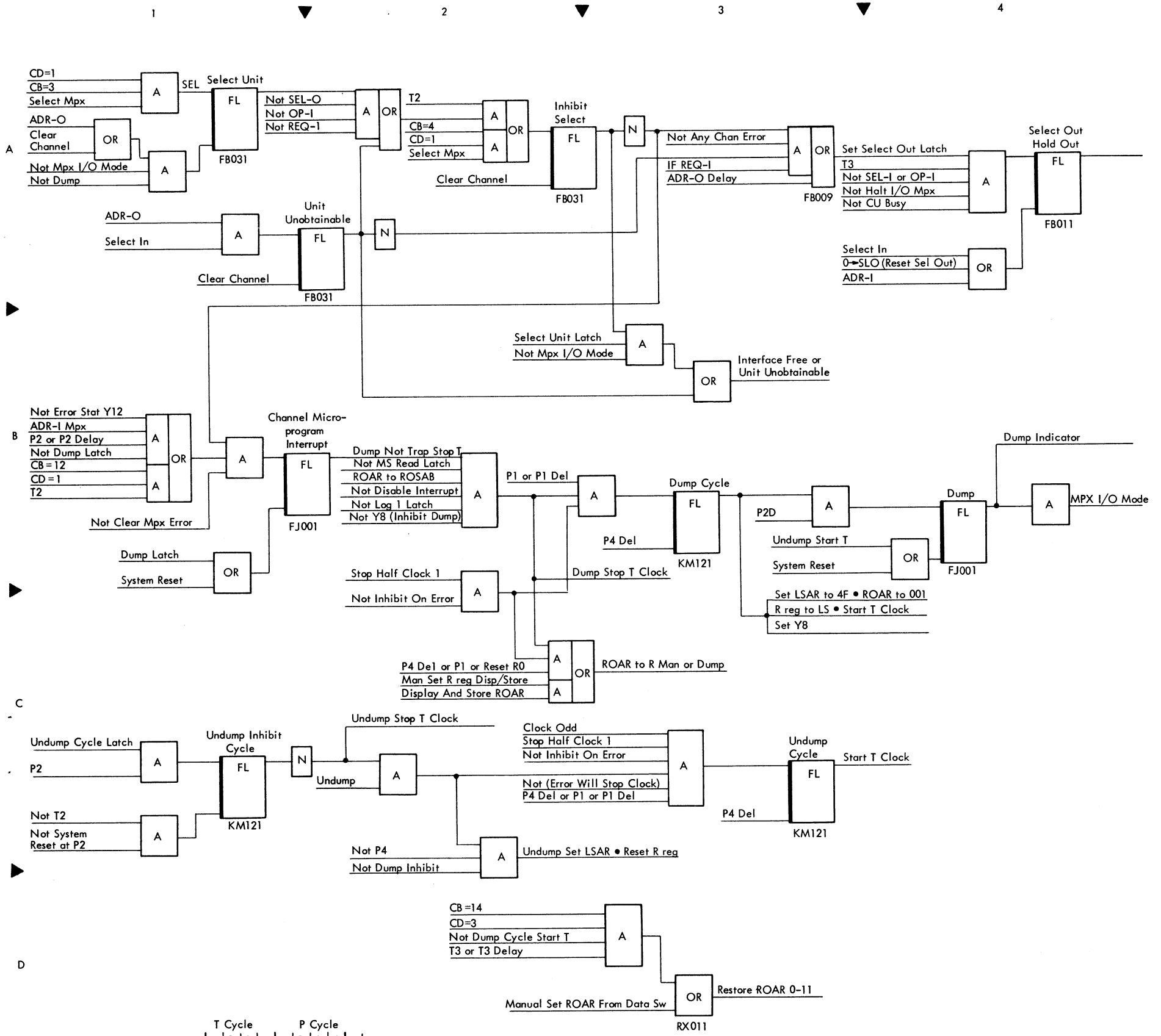


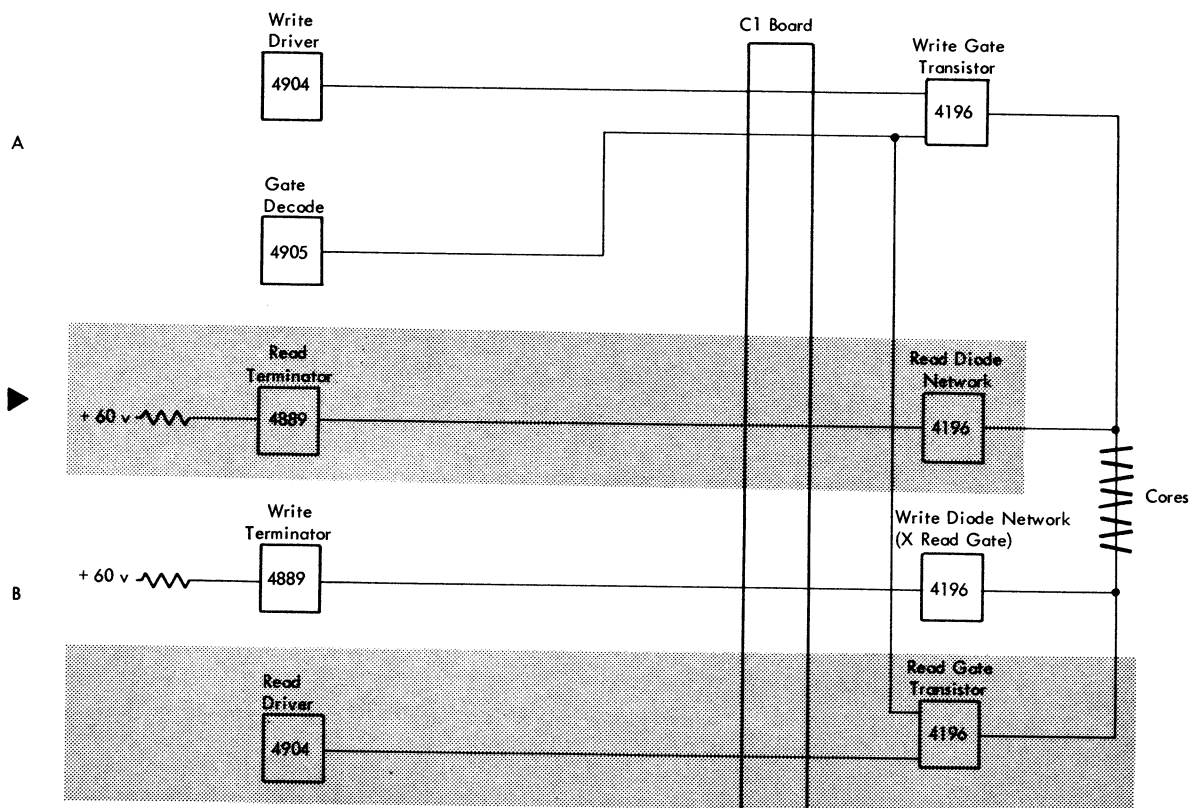
FIGURE 511. 2.5 kHz HF POWER SUPPLY WIRING DIAGRAM



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

SIMPLIFIED DRIVE SYSTEM



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

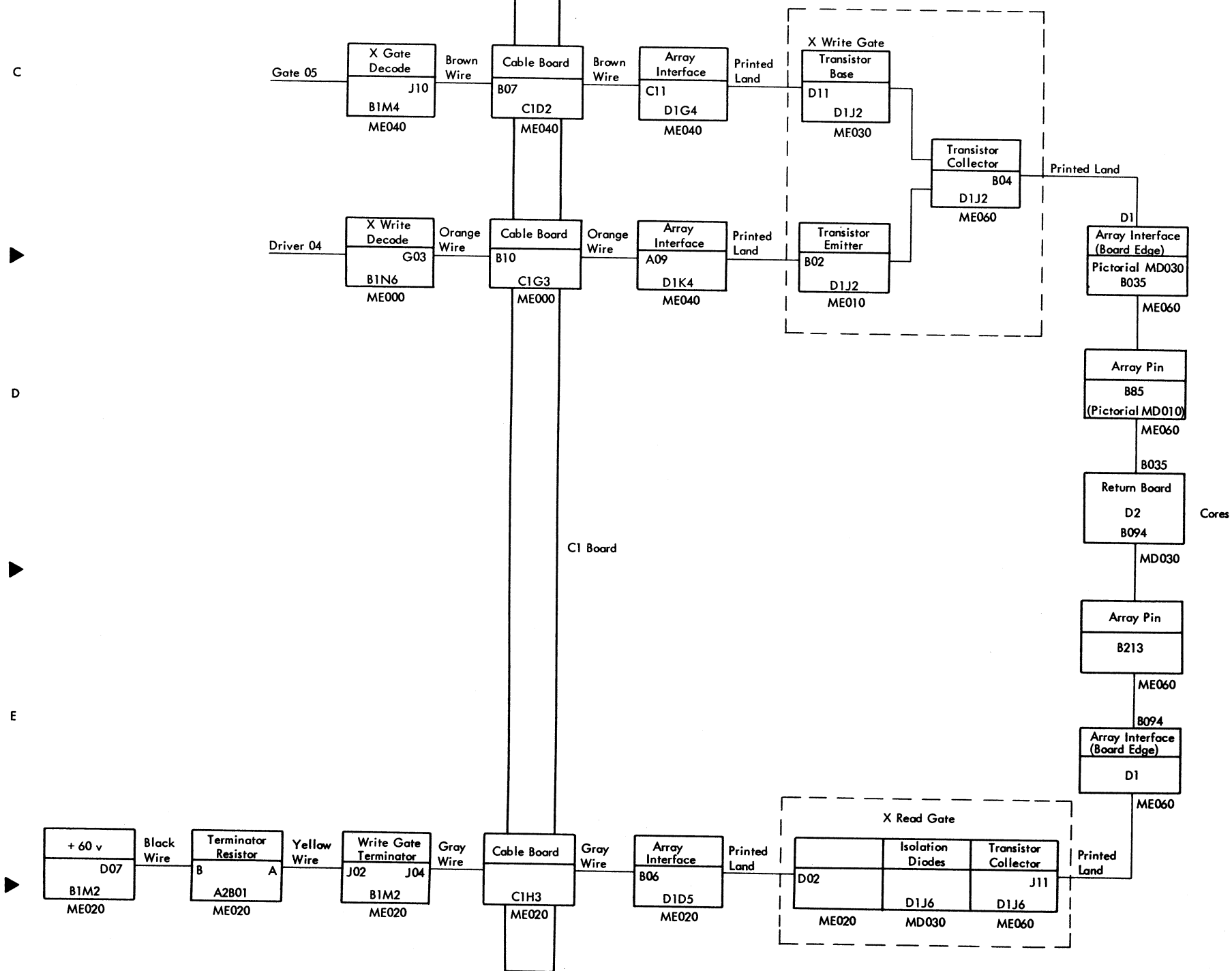


FIGURE 513. MAIN STORAGE X-DIMENSION DRIVE

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

A

	-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 394 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 AND 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 Normalized 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 303 QH031 303	42 STC Store Character 305 QH031 305	43 IC Insert Character 307 QH031 307	44 EX Execute 309 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 311 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 319 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 AND AND 615 QH011 308	55 CL Compare Logical 615 QH011 30A	56 OR OR 615 QH011 30C	57 X Exclusive OR 615 QH011 30E	58 LD Load 615 QH011 310	59 CD Compare 615 QH011 312	5A AD Add 615 QH011 314	5B SD Subtract 615 QH011 316	5C MD Multiply 620 QJ091 318	5D DD 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 347 QP001 347	94 AND AND 636 QP001 349	95 CLI Compare Logical 348 QP001 348	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 Half 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 AND AND 649 QP041 C08	D5 CLC Compare Logical 649 QP041 C0A	D6 OR OR 649 QP041 C0C	D7 XC Exclusive OR 649 QP041 C0E						DC TR Translate 641 QP061 C18	DD TRT Translate and Test 642 QP061 C1A	DE TRD 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 C18			

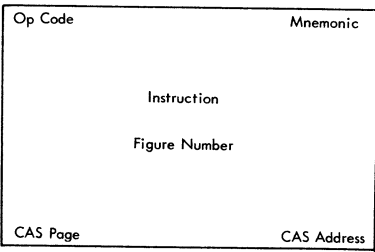


FIGURE 600. INSTRUCTION MATRIX



NO EXAMPLE SHOWN.

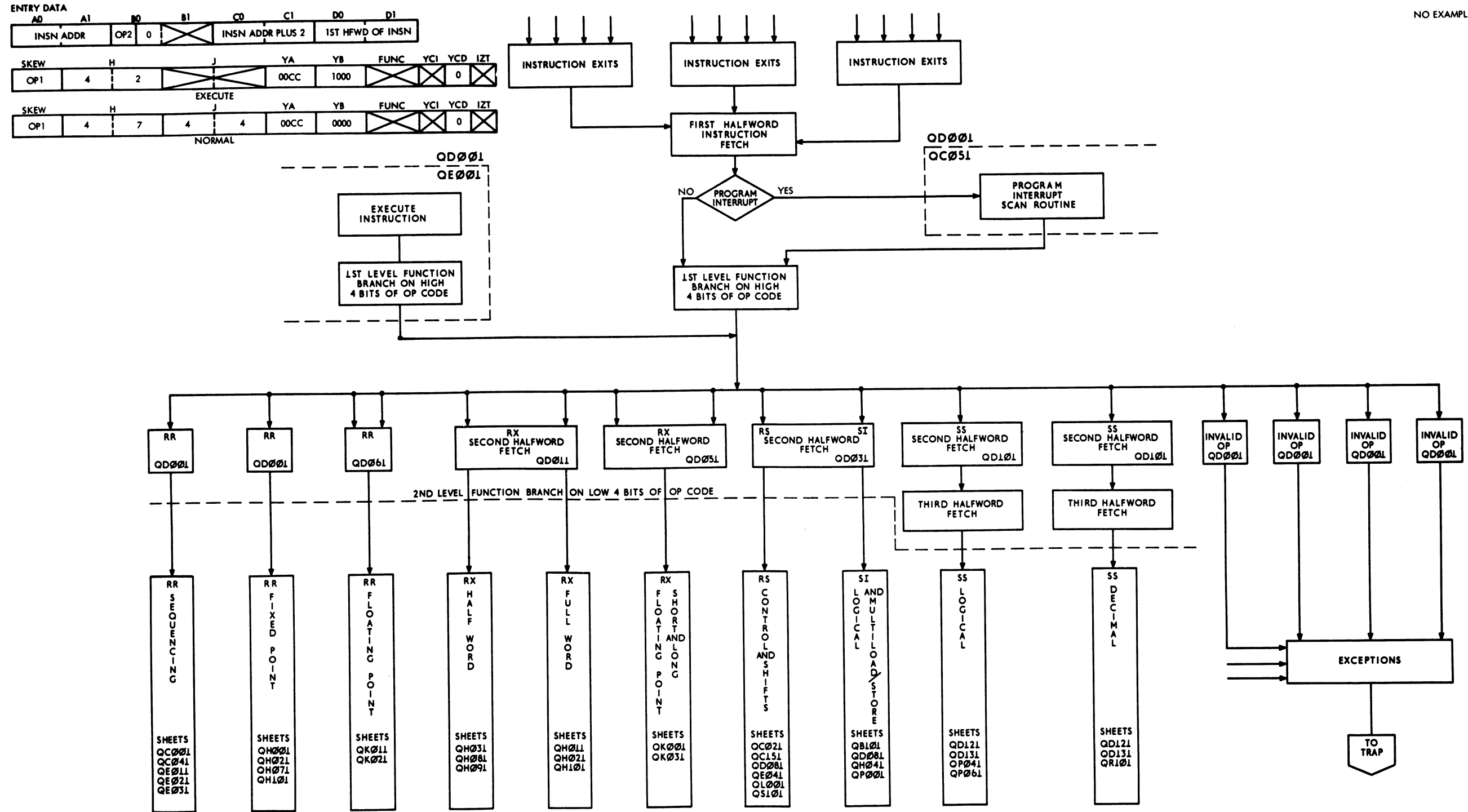
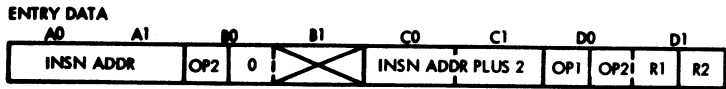
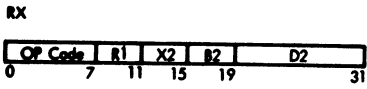
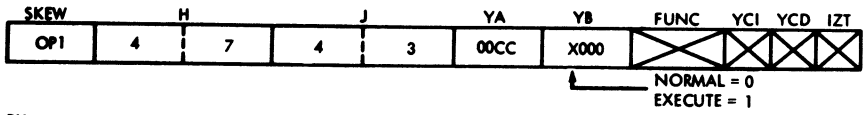


FIGURE 601. INSTRUCTION FETCH MICROPROGRAM



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

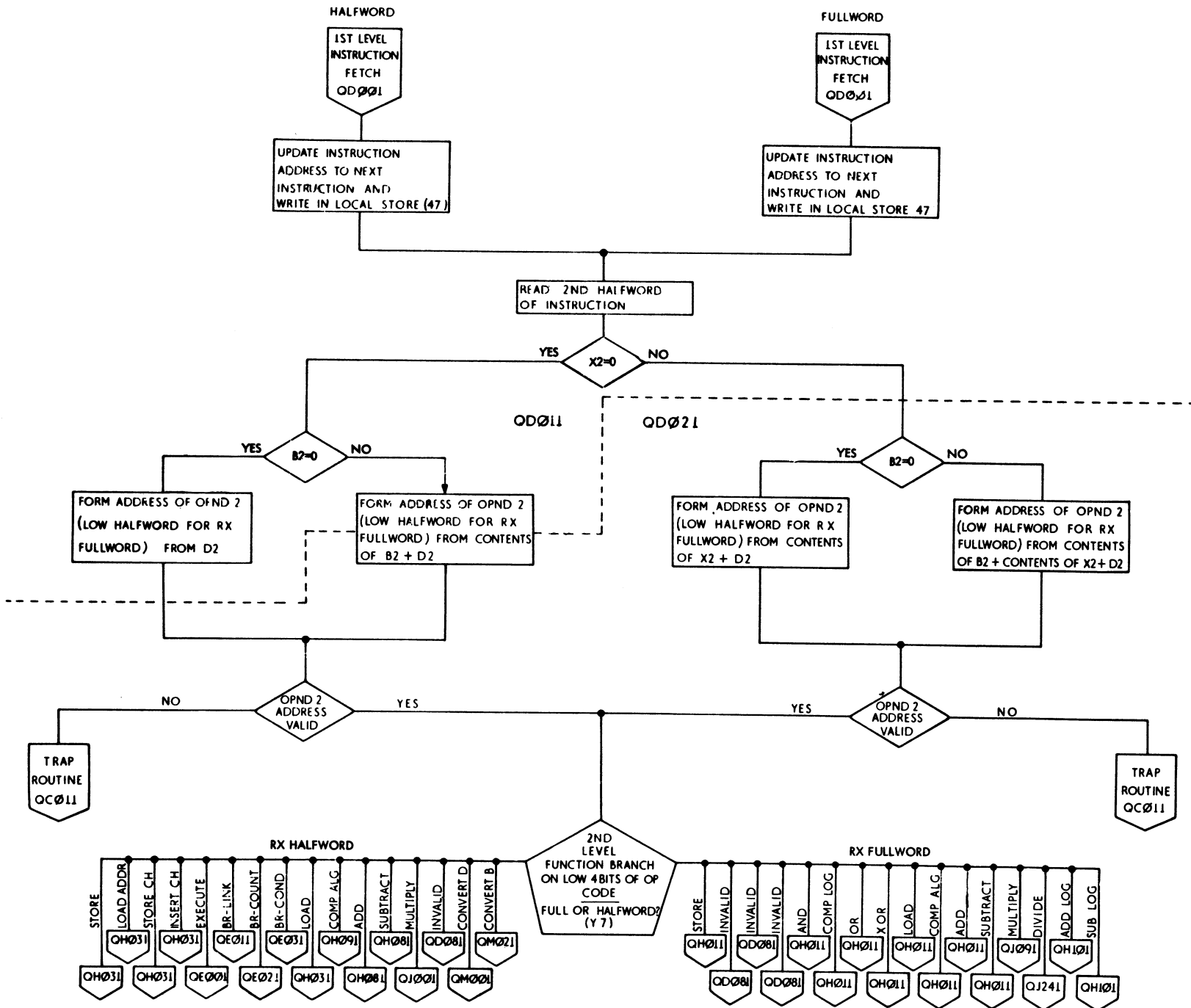
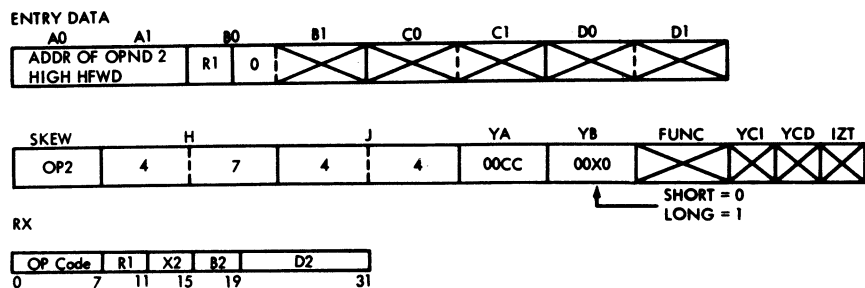


FIGURE 602.2ND LEVEL INSTRUCTION FETCH, RX FIXED POINT



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

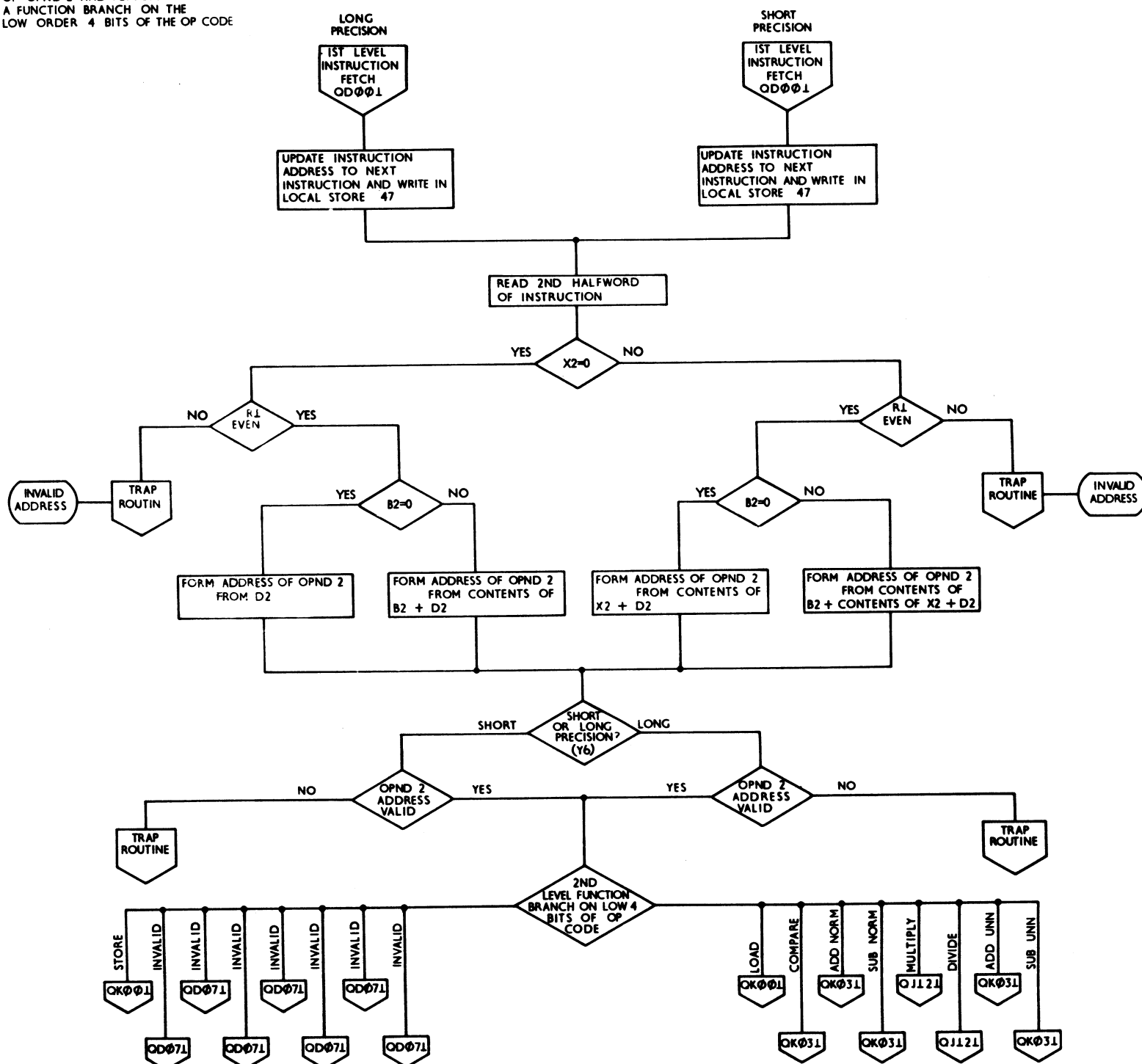
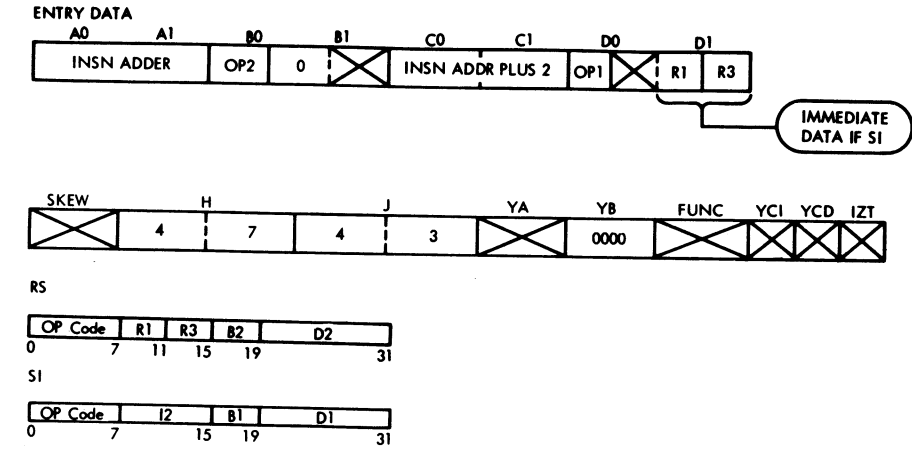


FIGURE 603.2ND LEVEL INSTRUCTION FETCH, RX FLOATING POINT (QD051)

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.

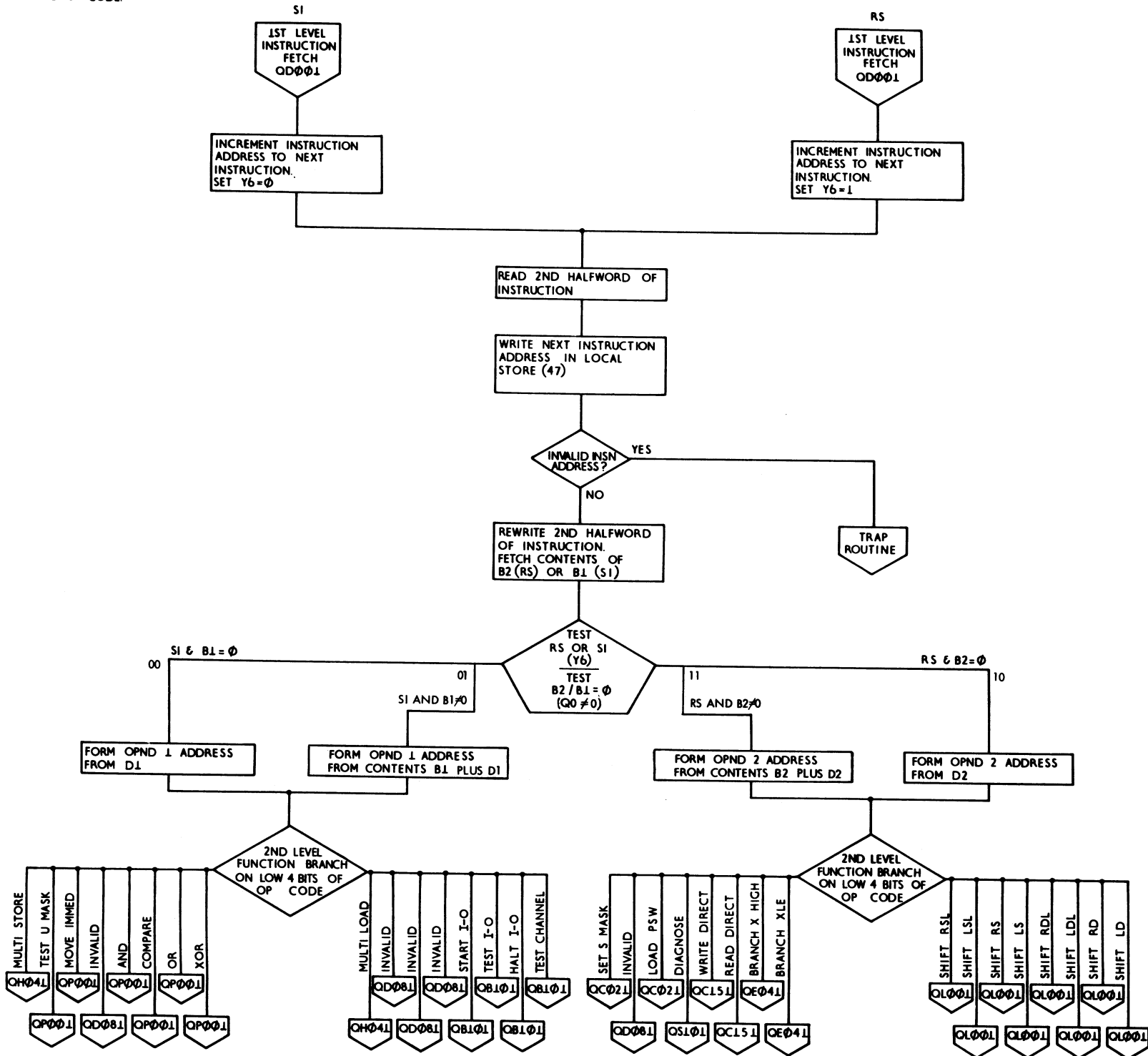
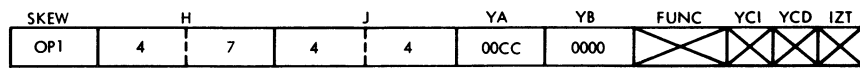
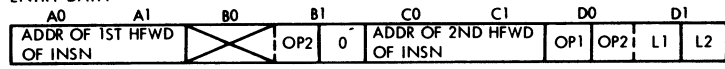
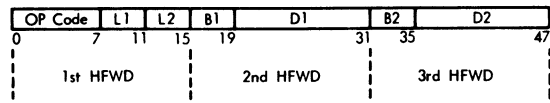


FIGURE 604. 2ND LEVEL INSTRUCTION FETCH, RS AND SI OPERATIONS (QD031)

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

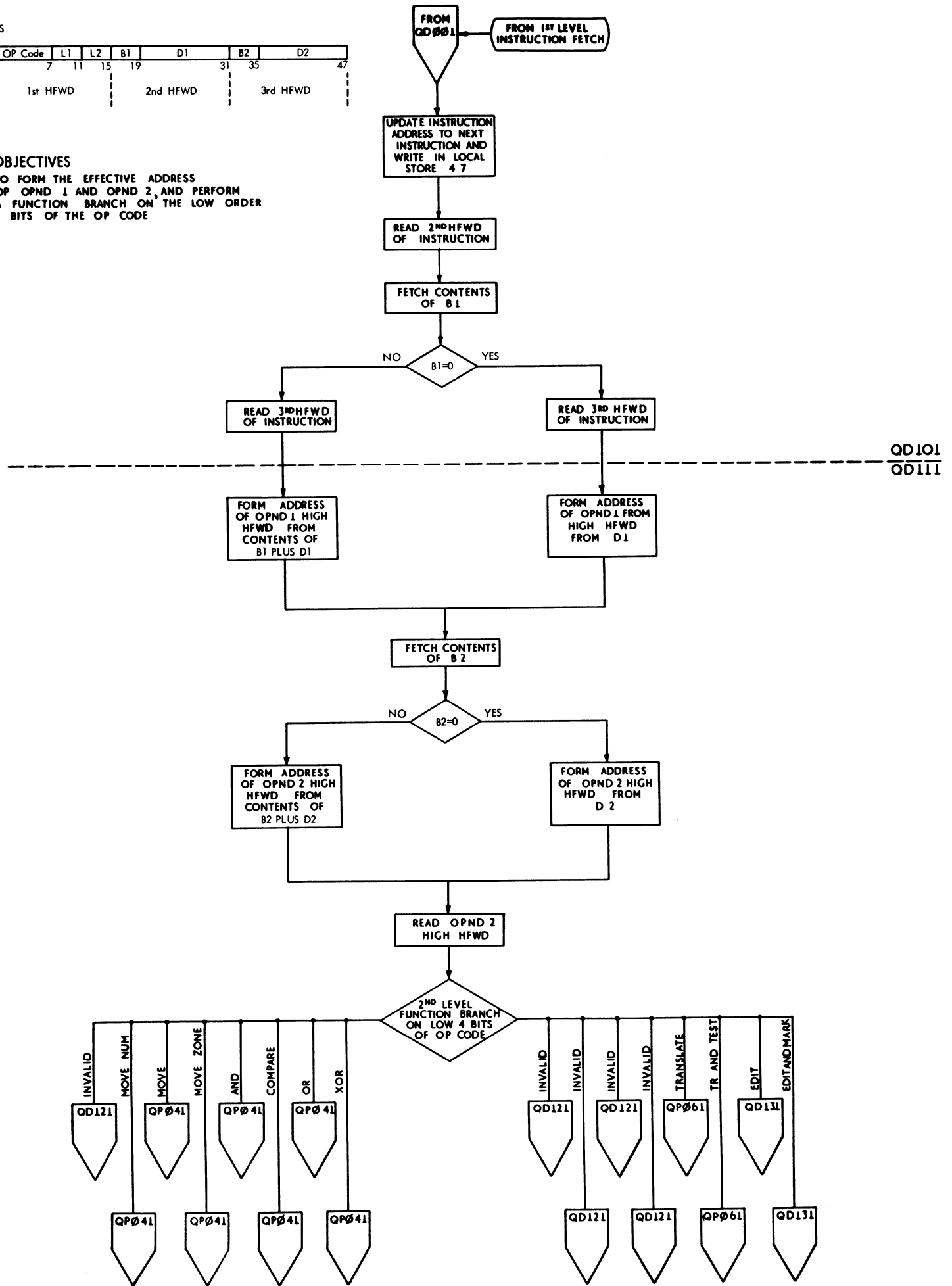
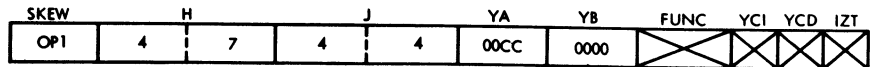
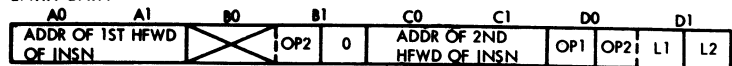
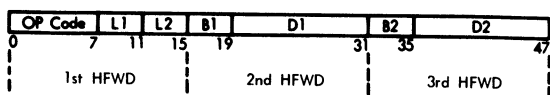


FIGURE 605. 2ND LEVEL INSTRUCTION FETCH, SS LOGICAL

ENTRY DATA



SS



NO EXAMPLE SHOWN.

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.

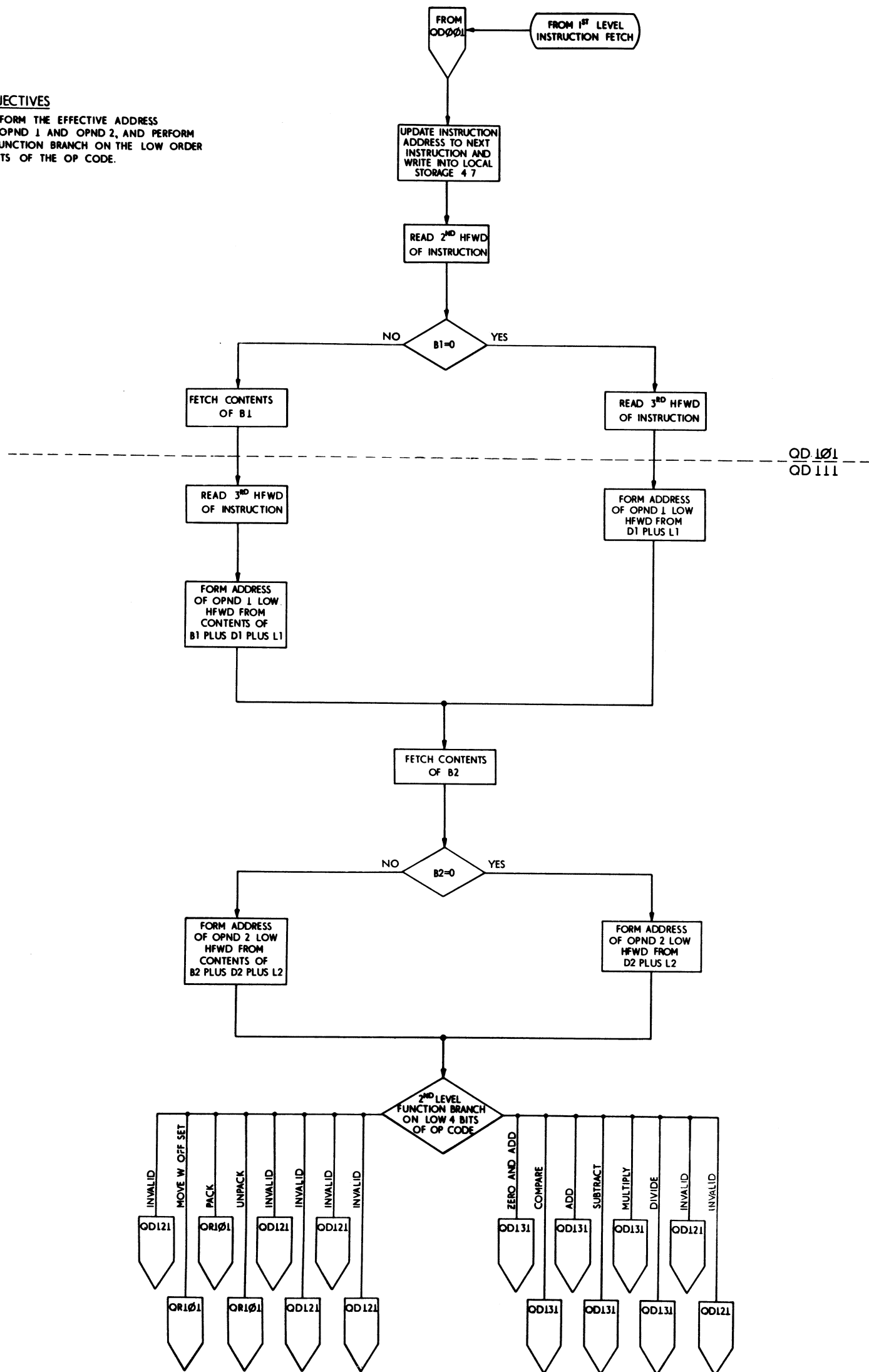


FIGURE 606 2ND LEVEL INSTRUCTION FETCH, SS DECIMAL

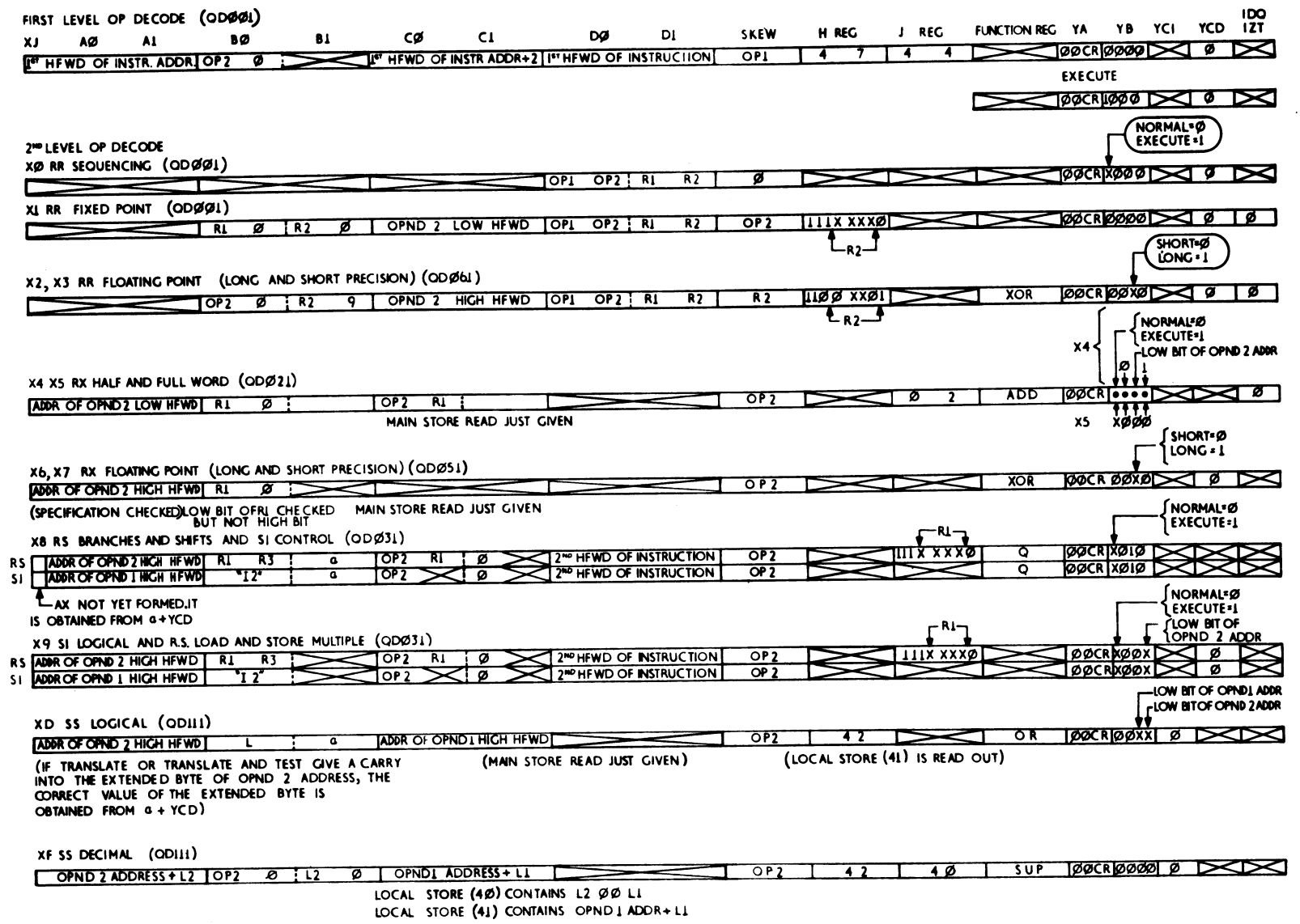
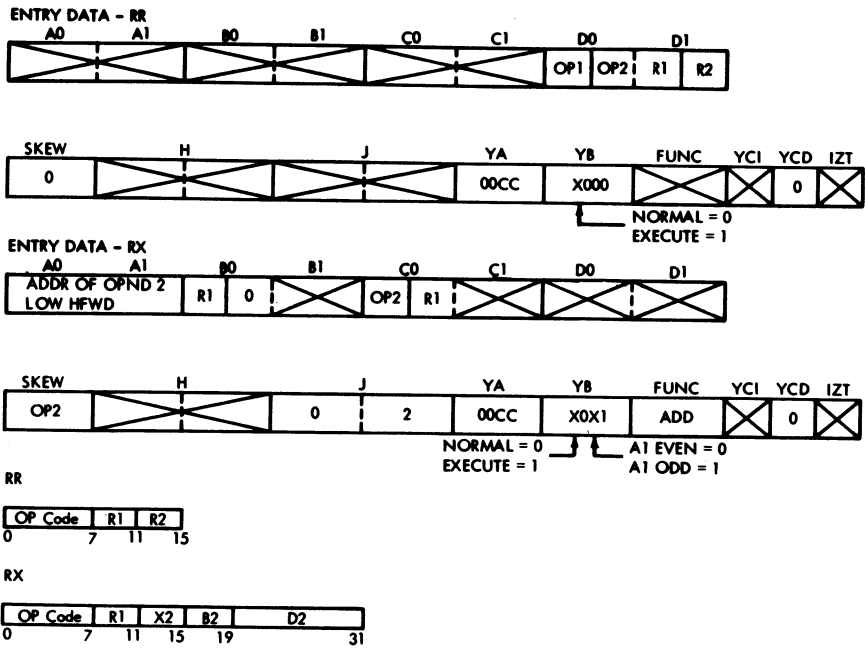


FIGURE 607 MACHINE STATUS AT 1ST AND 2ND LEVEL FUNCTION BRANCHES



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=∅ THIS INDICATES NO BRANCH  
OP CODE  
∅5-RR  
45-RX HALF

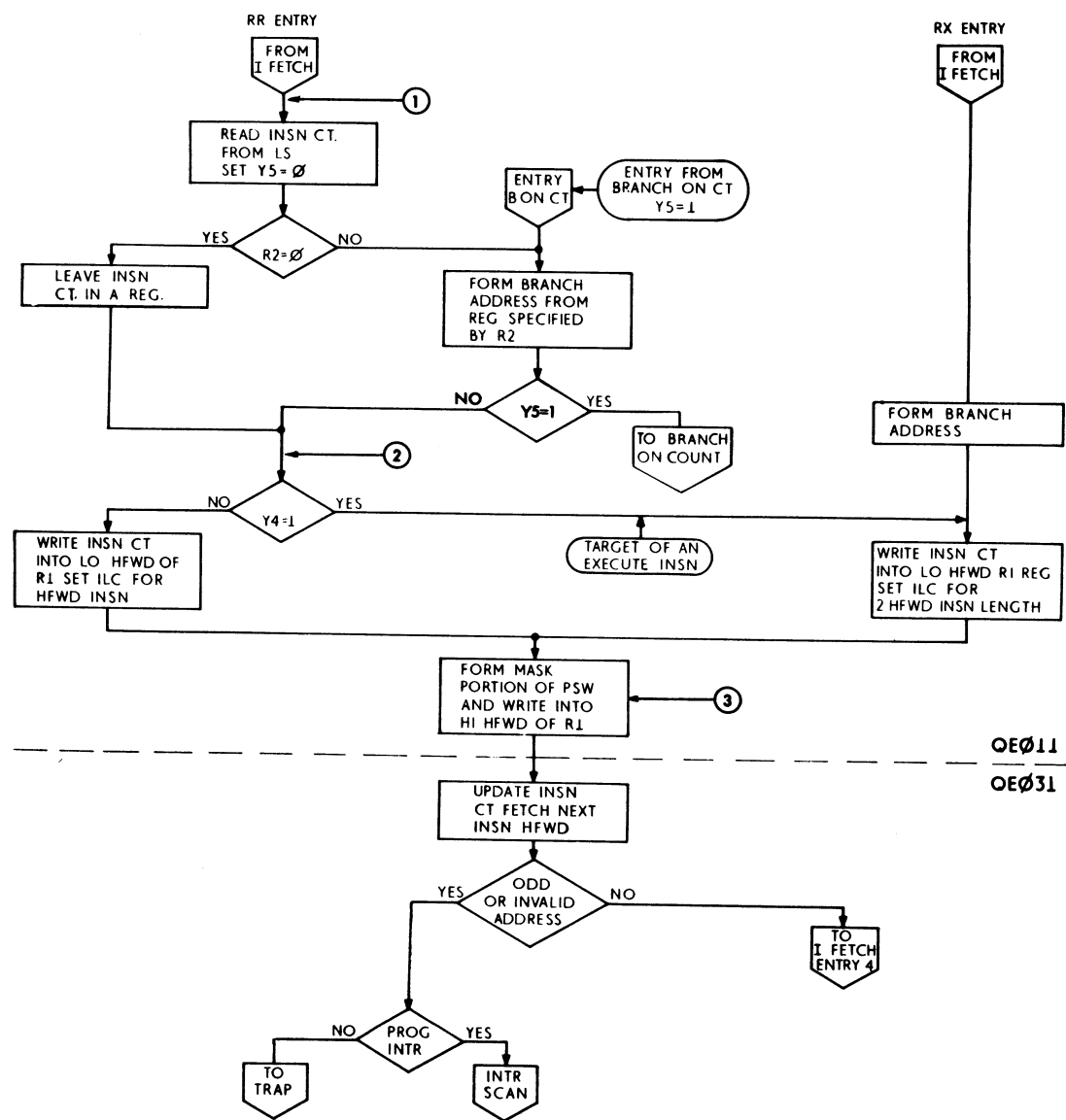
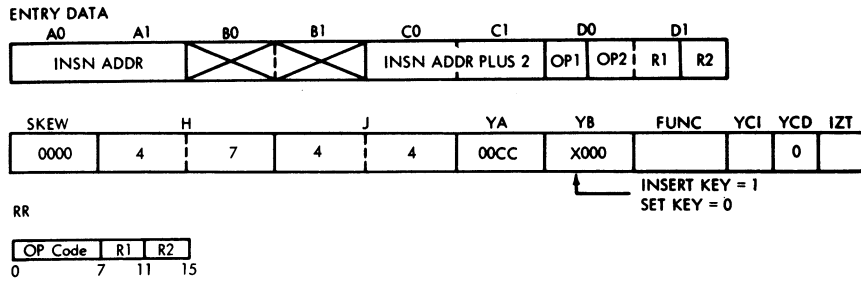


FIGURE 608. BRANCH AND LINK





**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.

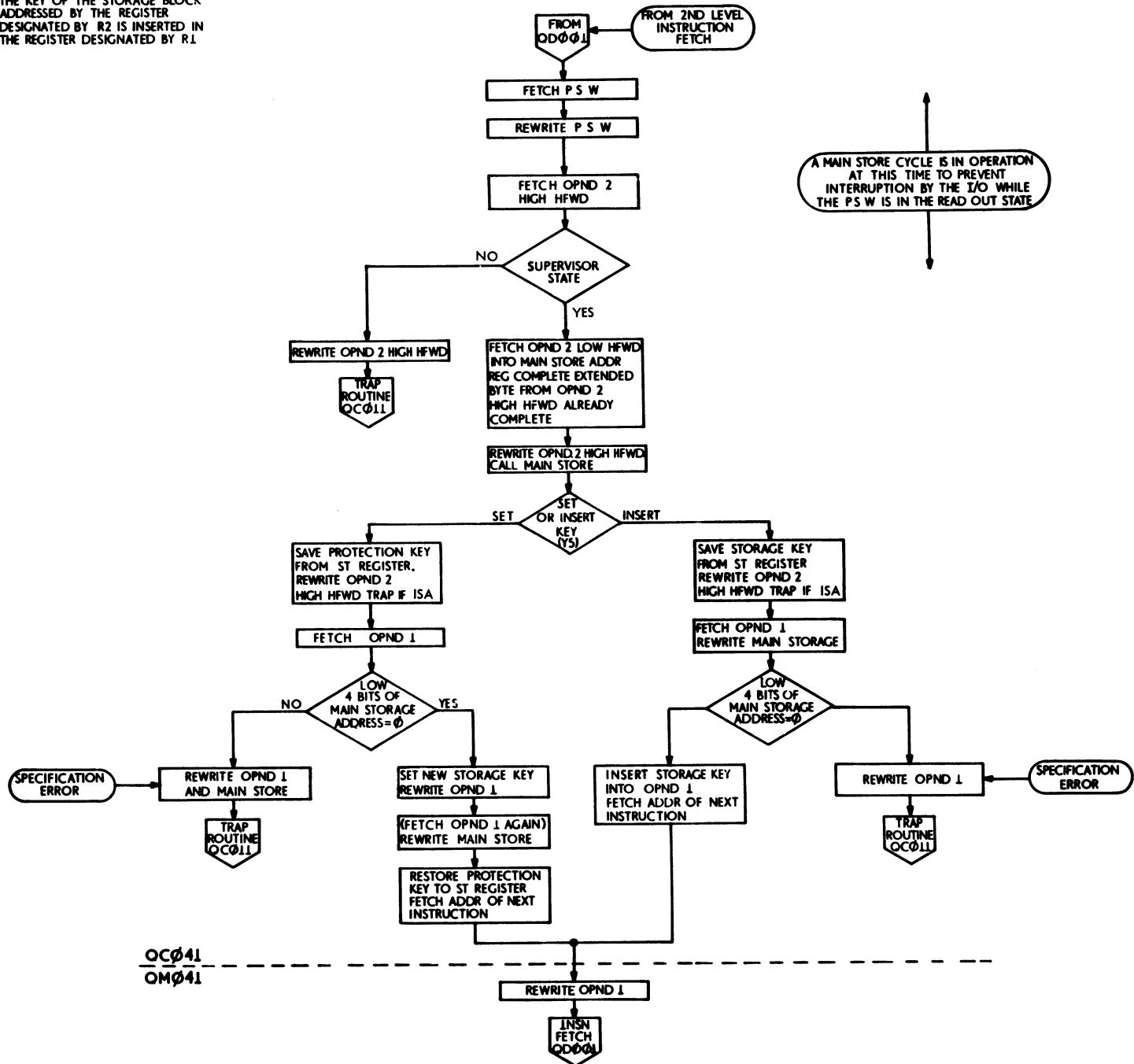
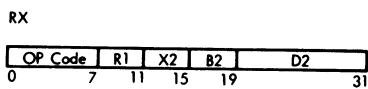
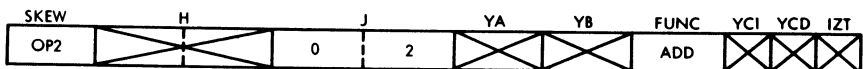
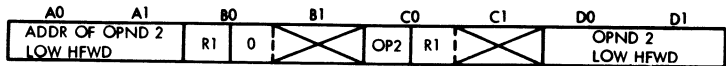


FIGURE 609, SET AND INSERT STORAGE KEY

ENTRY DATA



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

①	ROS	REG 5	REG 6	A REG	B REG	C REG	D REG
	31F	000068C4	000068C0	6AF8	2000	0000	0000
②	2AF	FFA42580		2040	2001	0000	0000
③	2A0	FFA42580		0000	2000	0000	0000
④	2C0	bbb0580		62C8	2000	0000	0580

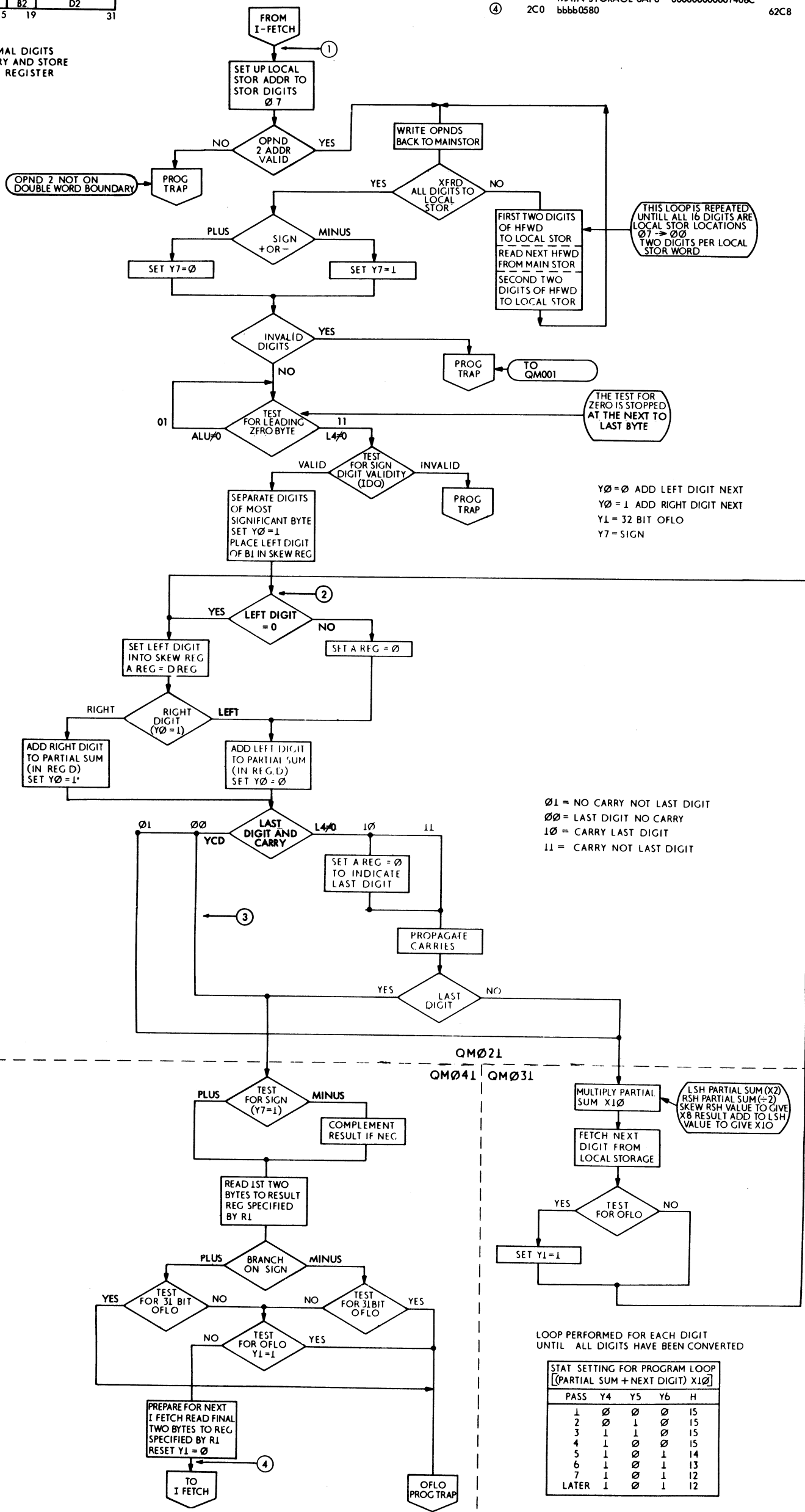
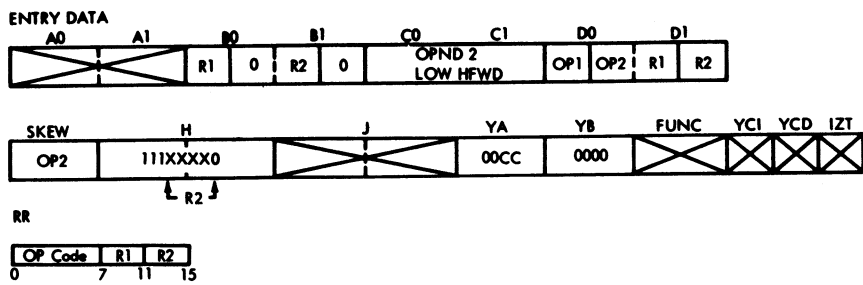


FIGURE 610 CONVERT DECIMAL TO BINARY



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	388 00006666	61FF	0000	0001	0022
③	1E8 66660001	3FFF	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)

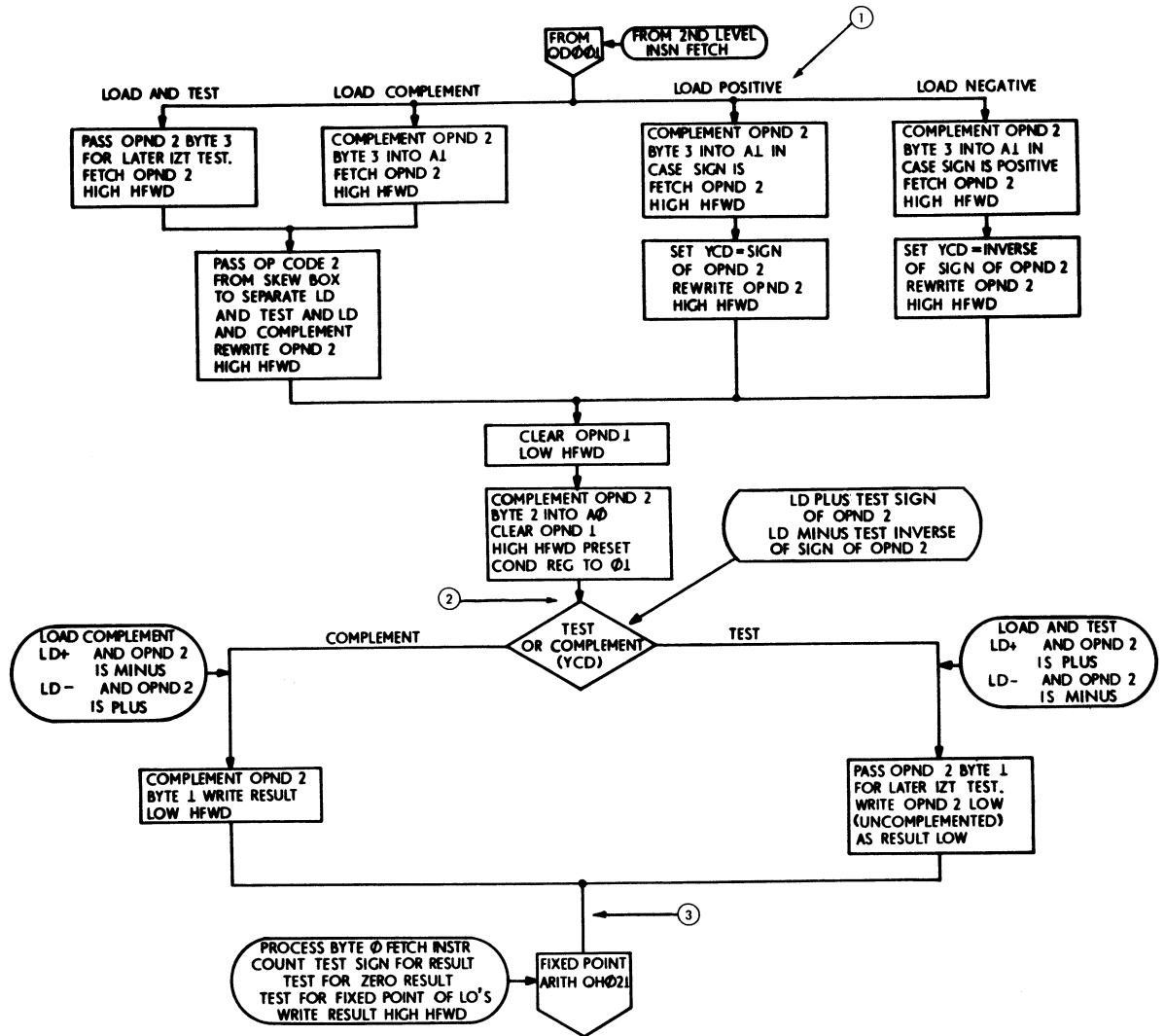


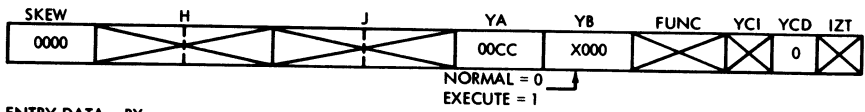
FIGURE 611 FIXED POINT SIGN OPERATIONS (QH071)

ENTRY DATA - RR

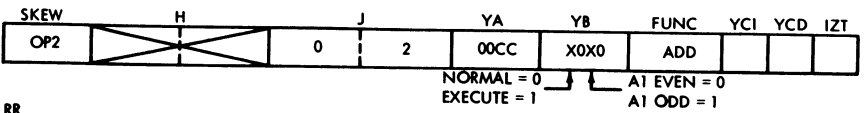
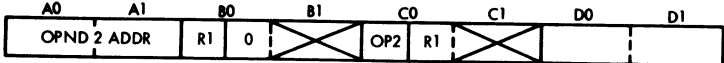


BRANCH ON COUNT BCT 02, 05 DMC SECTION 122, ROUTINE 25

	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



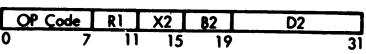
ENTRY DATA - RX



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

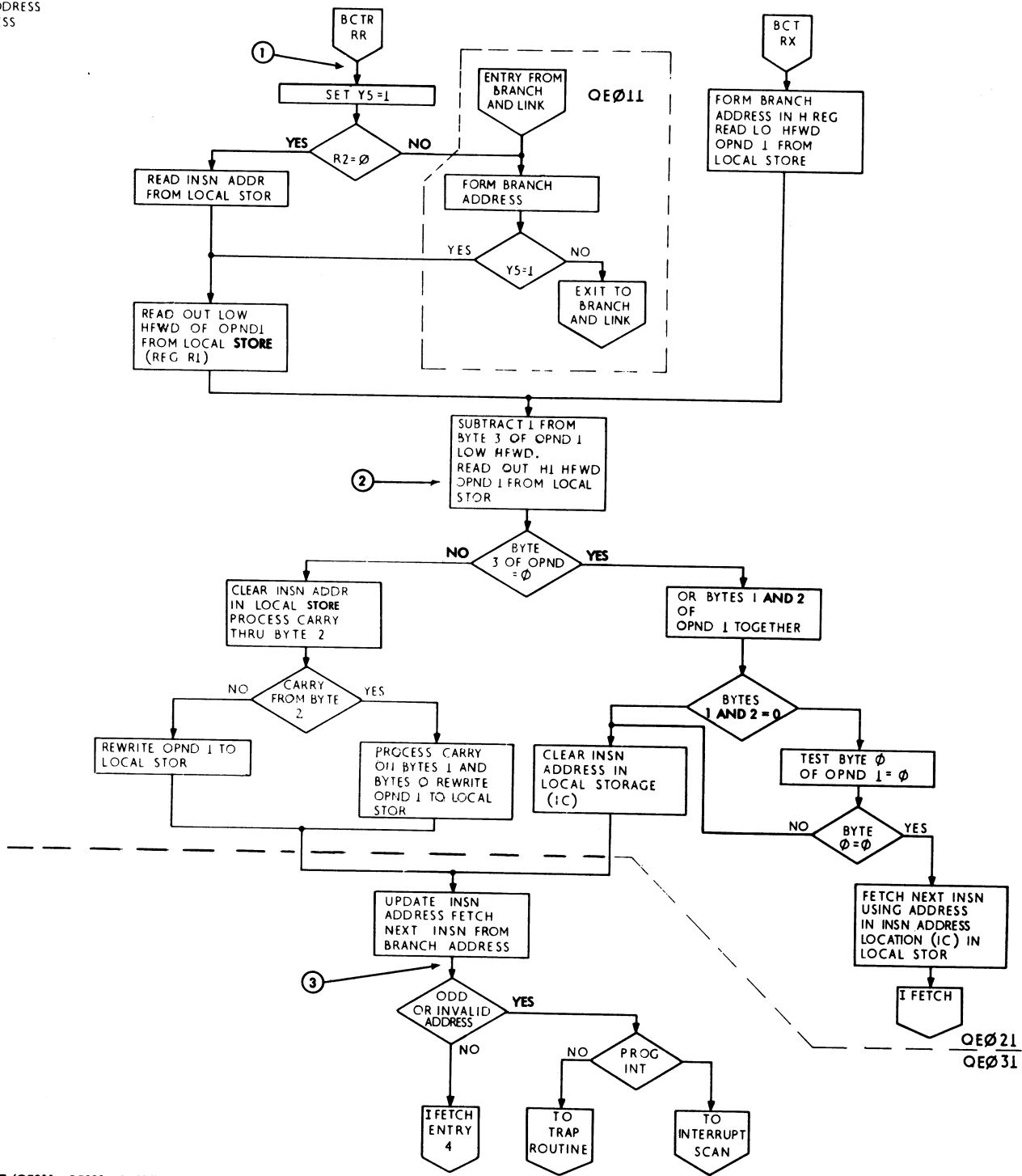
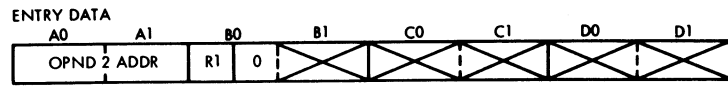
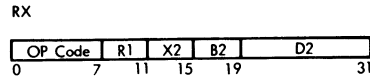
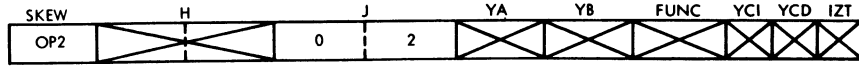


FIGURE 612. BRANCH ON COUNT (QE011, QE021, QE013)



CONVERT TO DECIMAL CVD

ROS	REG 0	DMC	SECTION 167, ROUTINE 12
① 31D	FFA42580	REG 8	A REG B REG C REG D REG
② 2CD	FFA42580	00000010	6AF8 4000 0000 0000
③ 2FE	FFA42580	00000010	6AFD 4000 1000 0010
			6AFE 4000 0000 016C



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

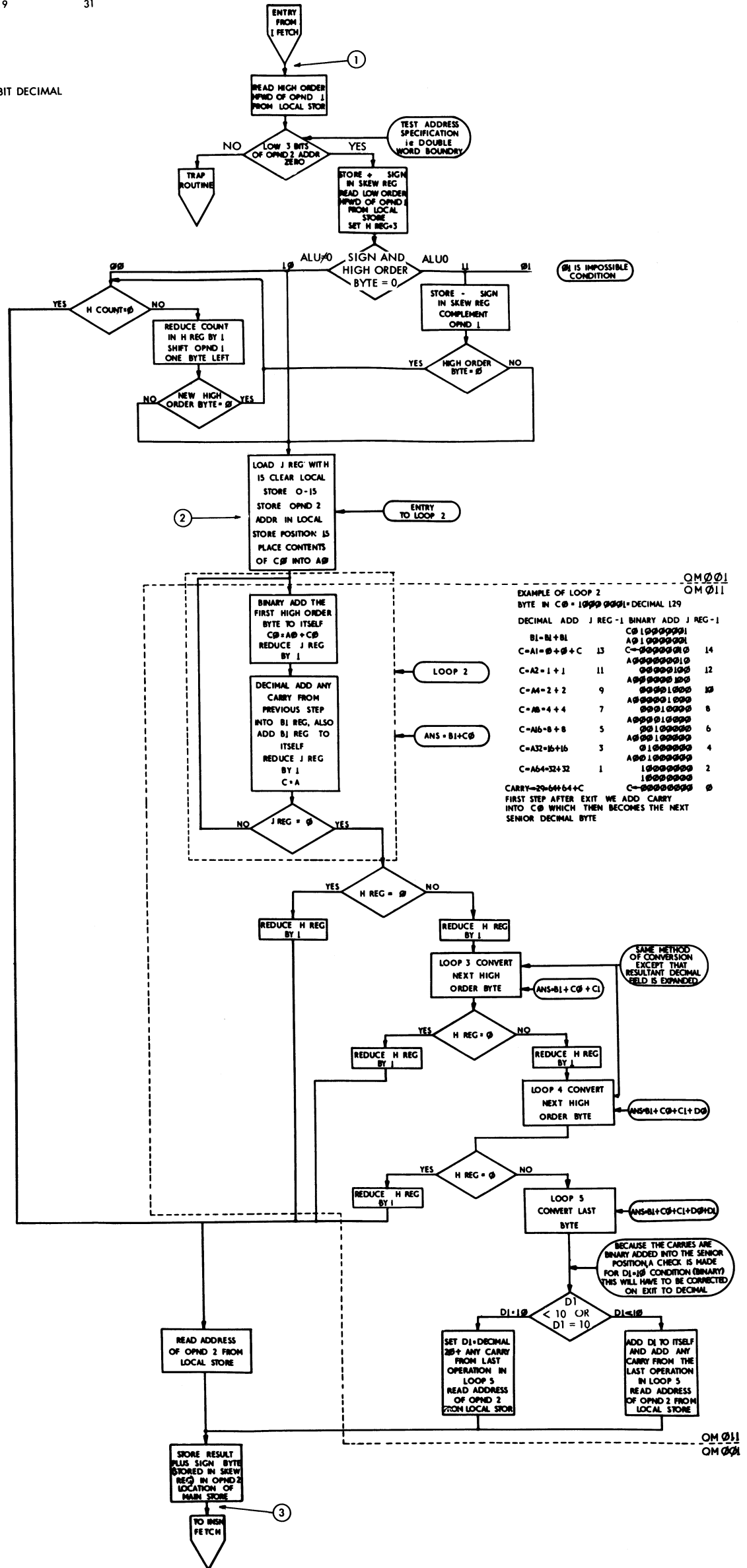
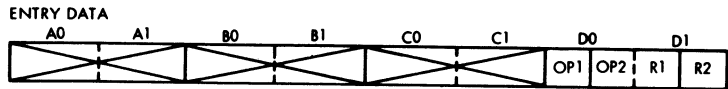
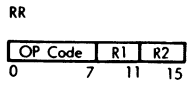
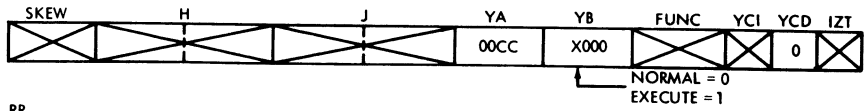


FIGURE 613. CONVERT BINARY TO DECIMAL



SET PROGRAM MASK  
 ROS REG 1  
 388 3F000000  
 3E4 3F000000

DMC SECTION 124, ROUTINE 24 STOP ON MS 6800  
 A REG B REG C REG D REG  
 6800 4010 6802 0410  
 6802 4010 6802 3F00



**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.

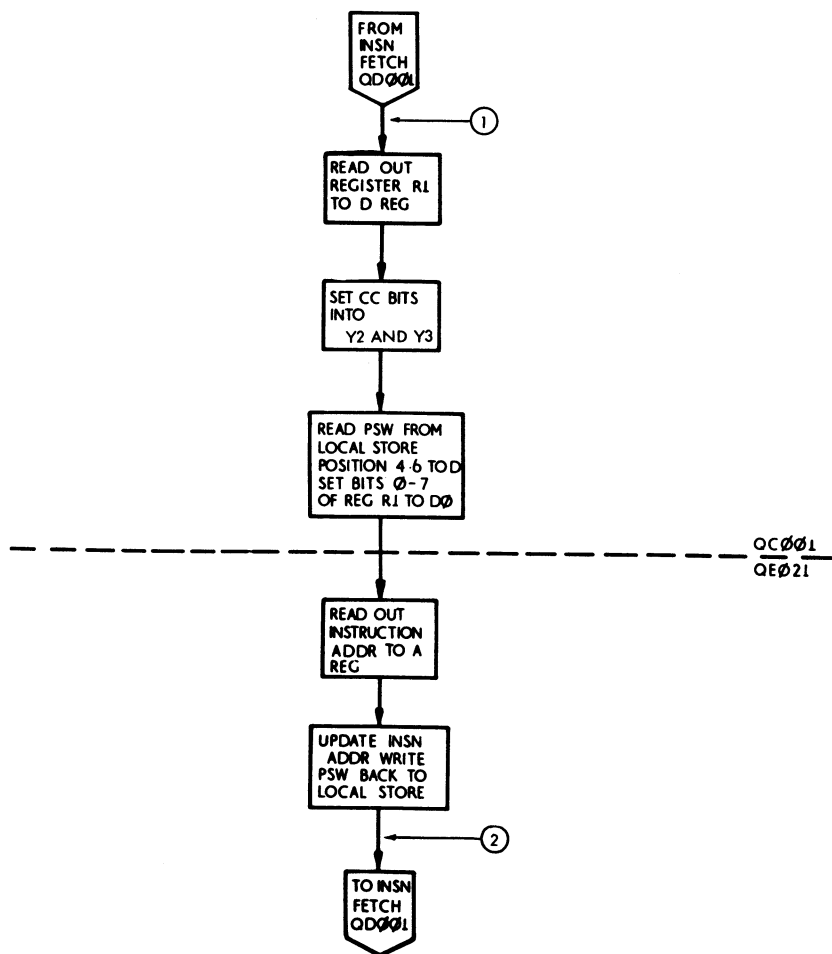
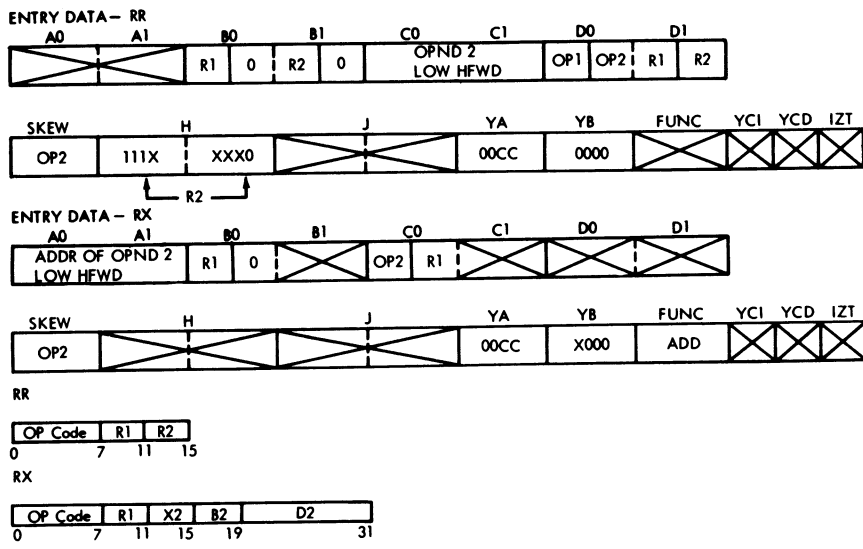


FIGURE 614, SET PROGRAM MASK



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

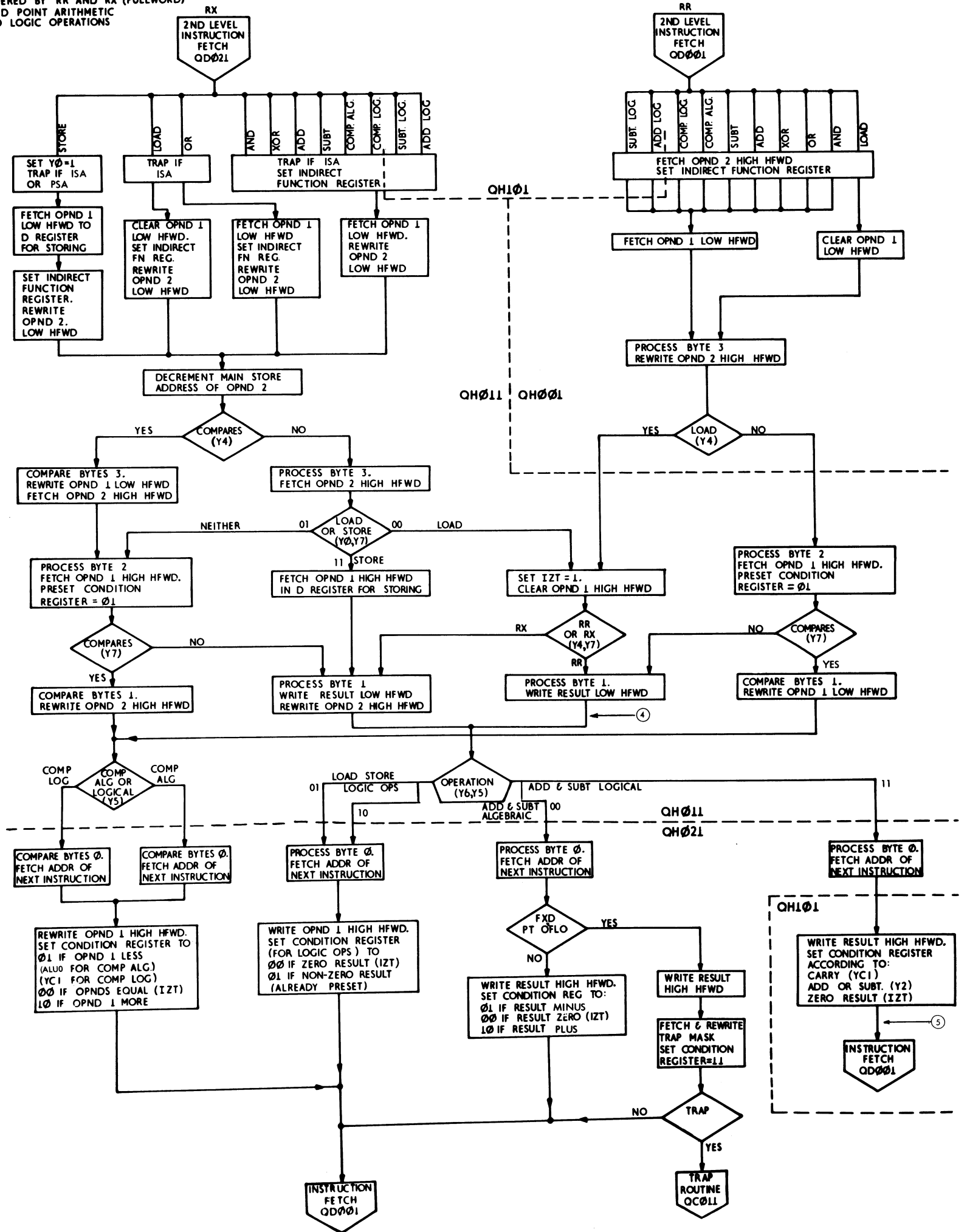
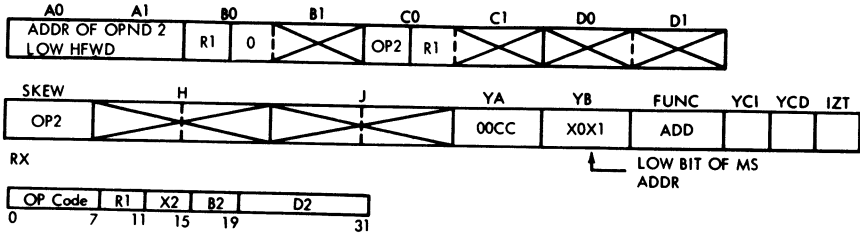


FIGURE 615. RR AND RX FIXED POINT ARITHMETIC AND LOGIC

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	66666666	22222222	6666	700E	2288 2222
③	1E4	66666666	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

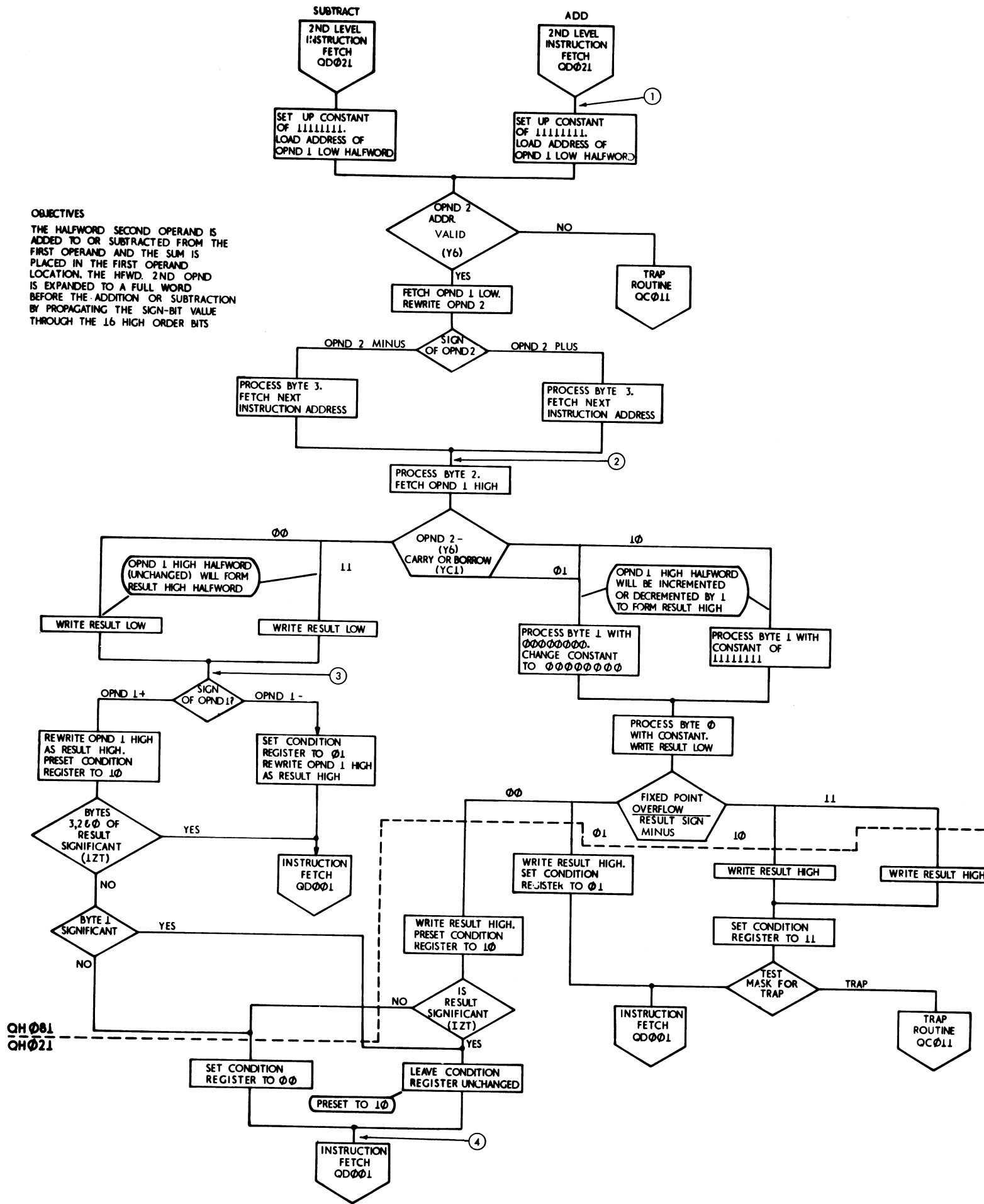
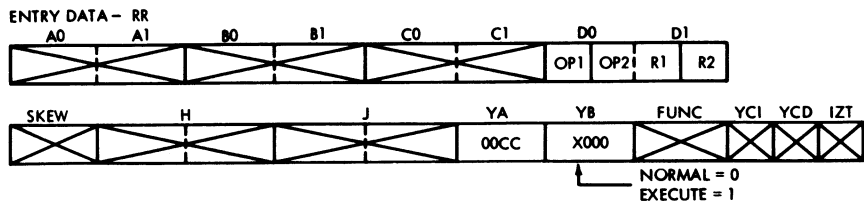


FIGURE 616.RX FIXED POINT ADD AND SUBTRACT

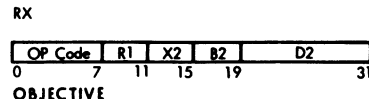
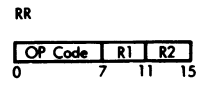
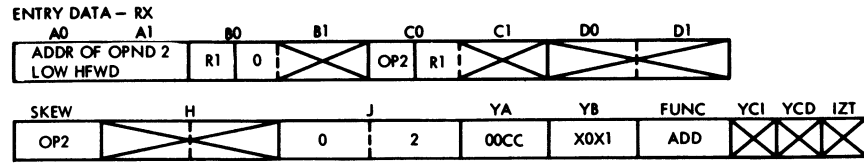




COMPARE DMC SECTION 105, ROUTINE 46 STOP ON MS 618C

①	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



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

**METHOD**  
COMPARISON IS FROM LEFT TO RIGHT, AS FOLLOWS:-

- 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
- 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
- COMPARE BYTES 2 (SEE DIAGRAM); IF NOT EQUAL, SET CONDITION REGISTER ACCORDING TO THE INEQUALITY, AND EXIT (TIME 5 CYCLES); IF EQUAL, THEN
- COMPARE BYTES 3 AND SET THE CONDITION REGISTER ACCORDING TO RESULT (TIME 6 CYCLES).

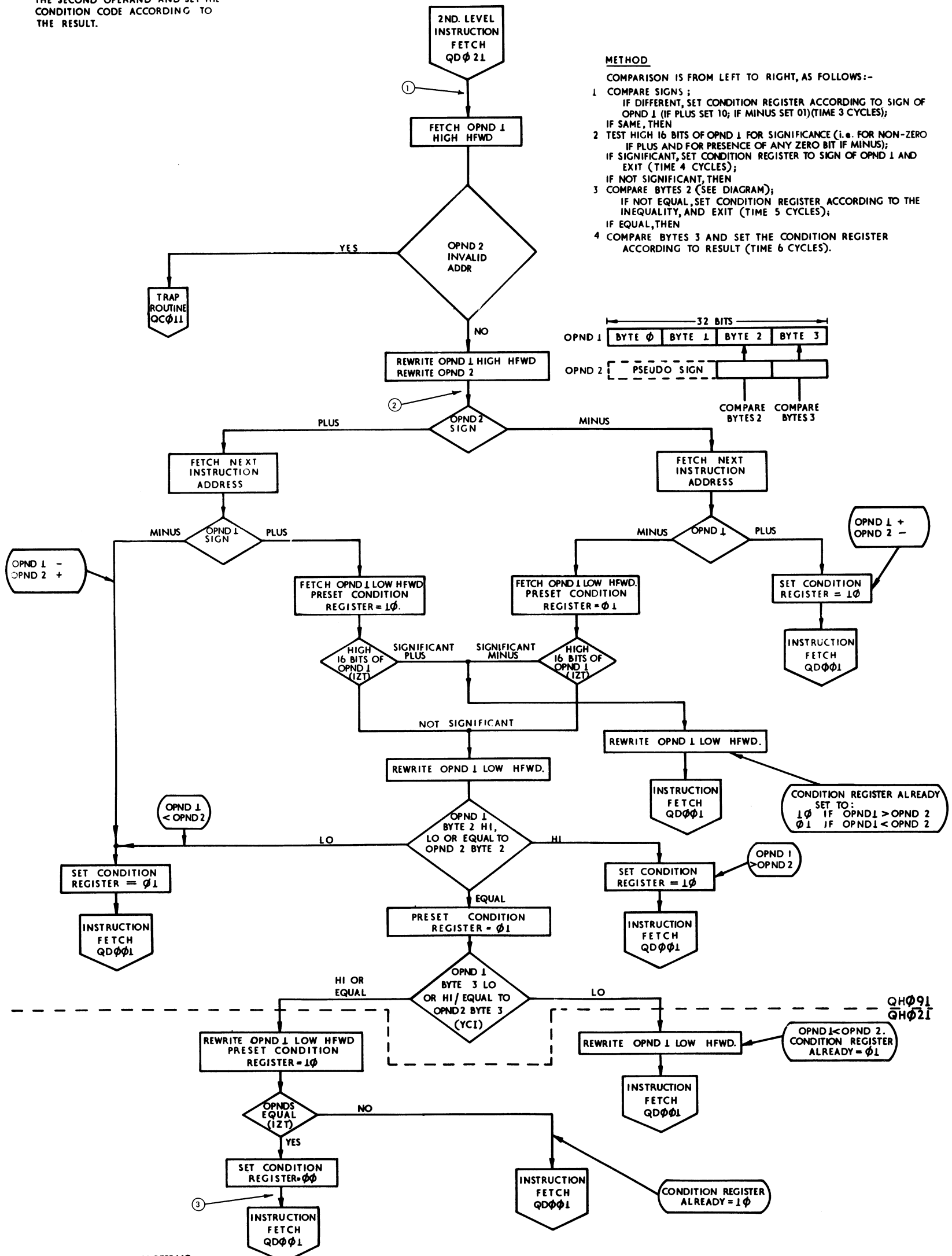
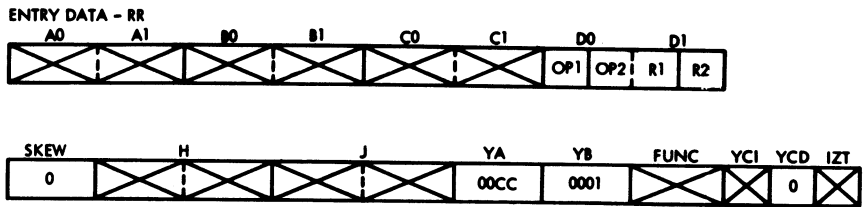
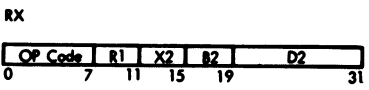
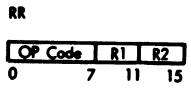
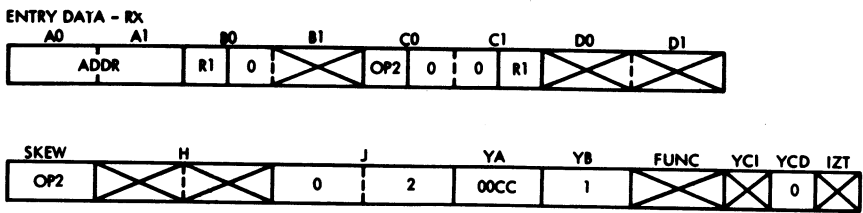


FIGURE 617. RX COMPARE ALGEBRAIC



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

①	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	00006666	6324	0000	631E	07D5



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

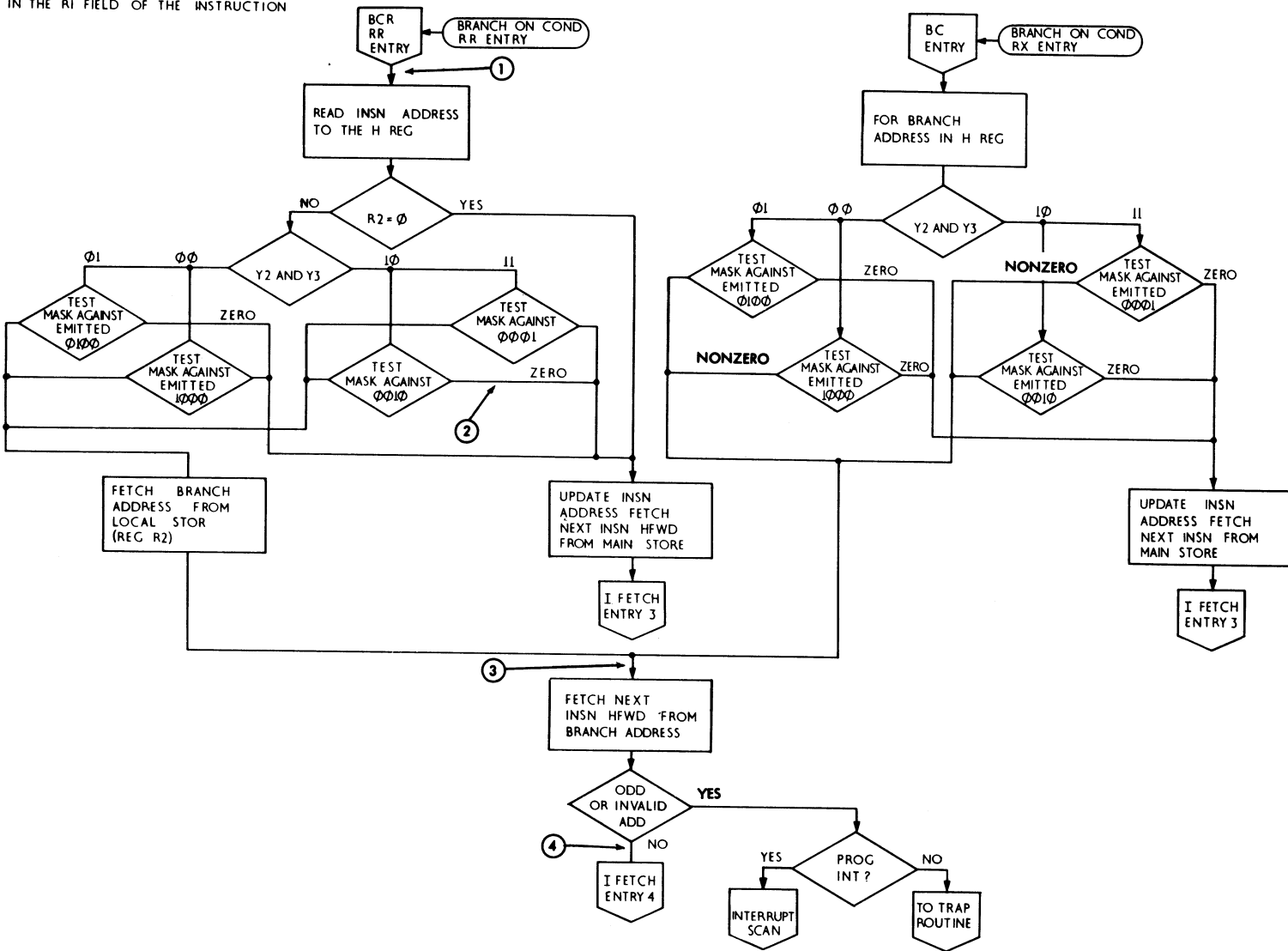
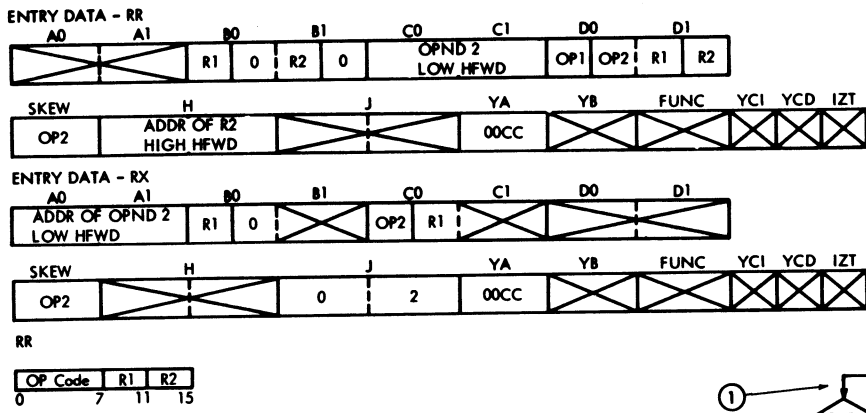


FIGURE 618. BRANCH ON CONDITION RR AND RX (QE031)



MULTIPLY RR FORMAT MR 02, 05 DMC SECTION 16D, ROUTINE 20

ROS	REG 2	REG 3	REG 5	A REG	B REG	C REG	D REG
① 399	0399FFDF	00000030	AAAAAAAA	65E4	0399	AAAA	1C25
② 938	AAAAFFEF	FFFFbbbb	AAAAAAAA	0000	0000	0000	AAAA
③ 880	AAAAFFEF	FFFFbbbb	AAAAAAAA	FF02	EF00	0000	A080
④ 92C	bbbbFFEF	FFFFFE00	AAAAAAAA	FFEF	FFFF	0000	A080

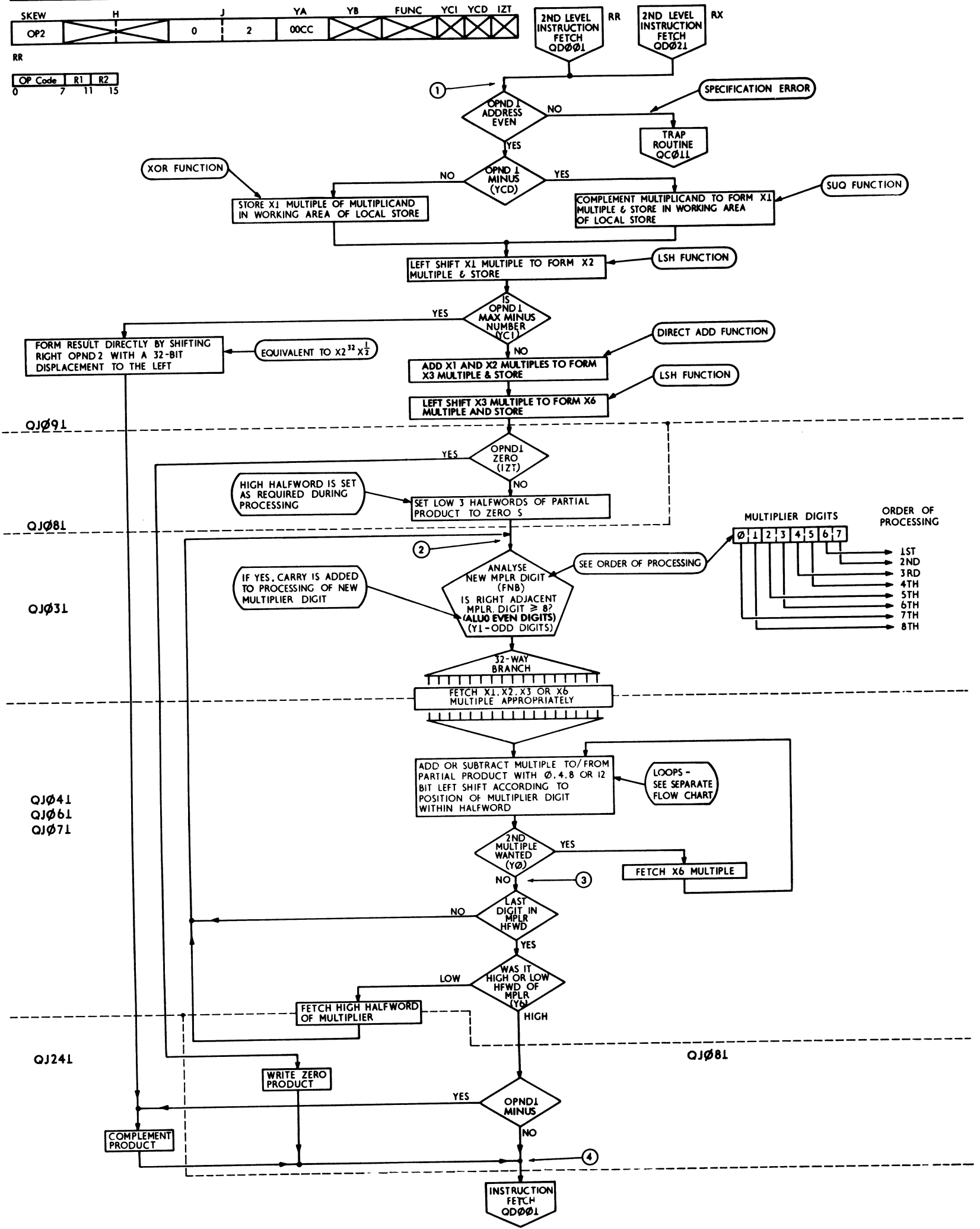
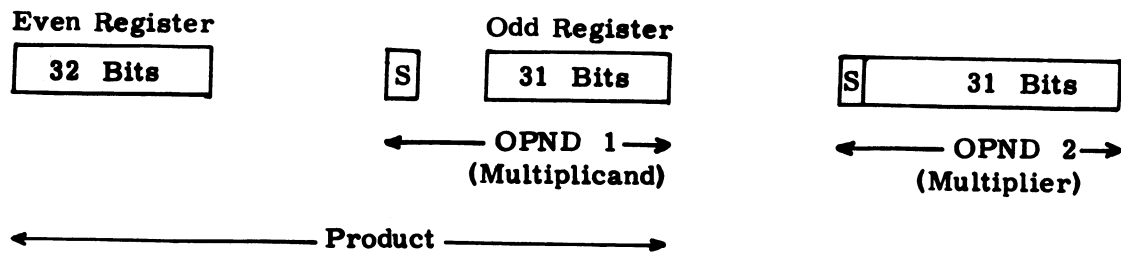


FIGURE 620. RR AND RX FIXED POINT MULTIPLY



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**

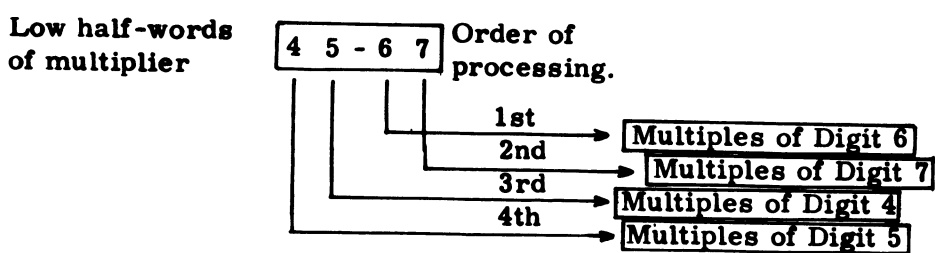
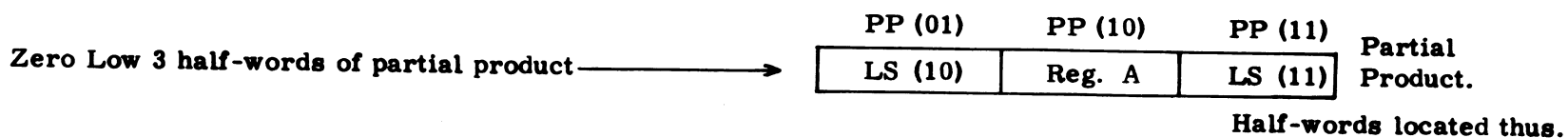
1. The Multiplicand is complemented if it is negative.
2. X1, X2, X3 and X6 multiples of the Multiplicand are formed and held in Local Storage. Each multiple occupies 2 half-words and for X3 and X6 multiples the extended bits of the high order half-word are used to store the overflow bits.
3. The low 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		Multiplier Digit	Multiples Used	
	No Carry In	Carry In		No Carry In	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 from Skew and byte offset appropriate to the position of the multiplier digit within the half-word.

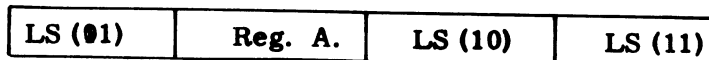
Register H addresses the multiples in Local Storage and during the Add/Subtract loop each half-word of a multiple is read to Register C. Register J addresses the Even/Odd register pair. The partial product is stored as shown in the diagram, half-words being read from the register pair to Reg. B.

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.

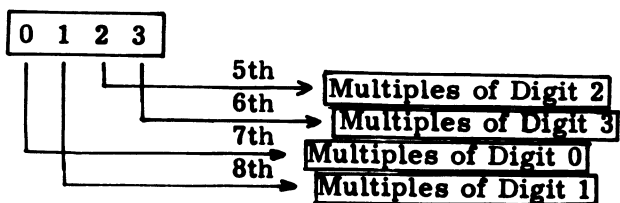


Multiples are added or subtracted with Skew and byte shift appropriate to position of multiplier digit.

This forms partial product

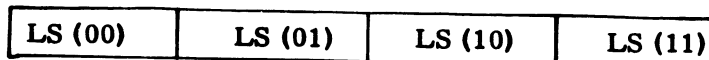


High half-word of multiplier



The numbers in brackets after 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.

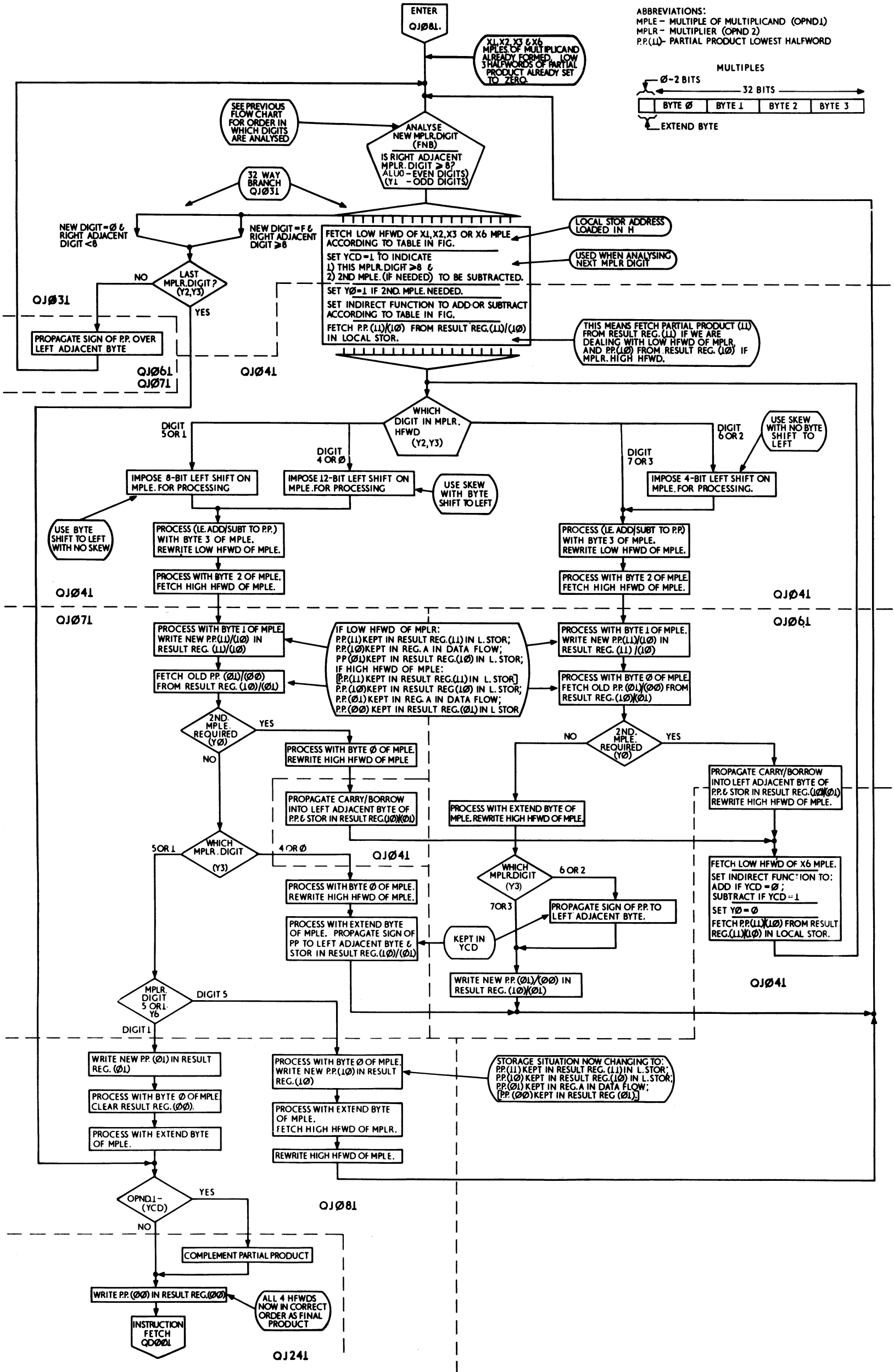
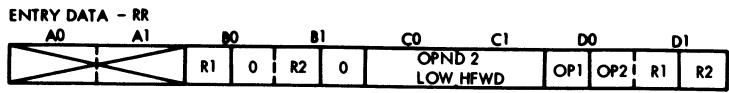
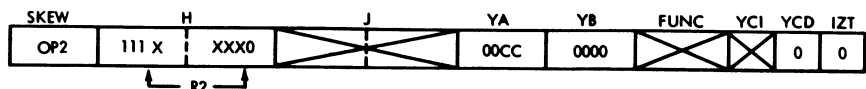


FIGURE 622.RR AND RX FIXED POINT MULTIPLY, DETAIL OF LOOPS

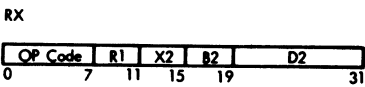
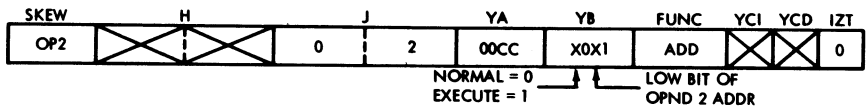
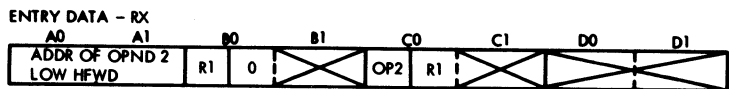


FIGURES 623 AND 624 DR 8,0

DMC Section 170 ROUTINE 20



ROS	REG 0	REG 8	REG 9	A REG	B REG	C REG	D REG	CAS
① 398	80000000	C0000000	80000000	657C	8000	0000	1D80	QJ241
② 973	80000000	3FFFbfff	80000000	0000	0000	8000	FFFF	QJ271
③ 9CD	80000000	bbbbbbbbb	00000000	0000	0000	FFFF	FFFF	QJ221
④ 9F0	80000000	bbbbbbbbb	7FFFbfff	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.

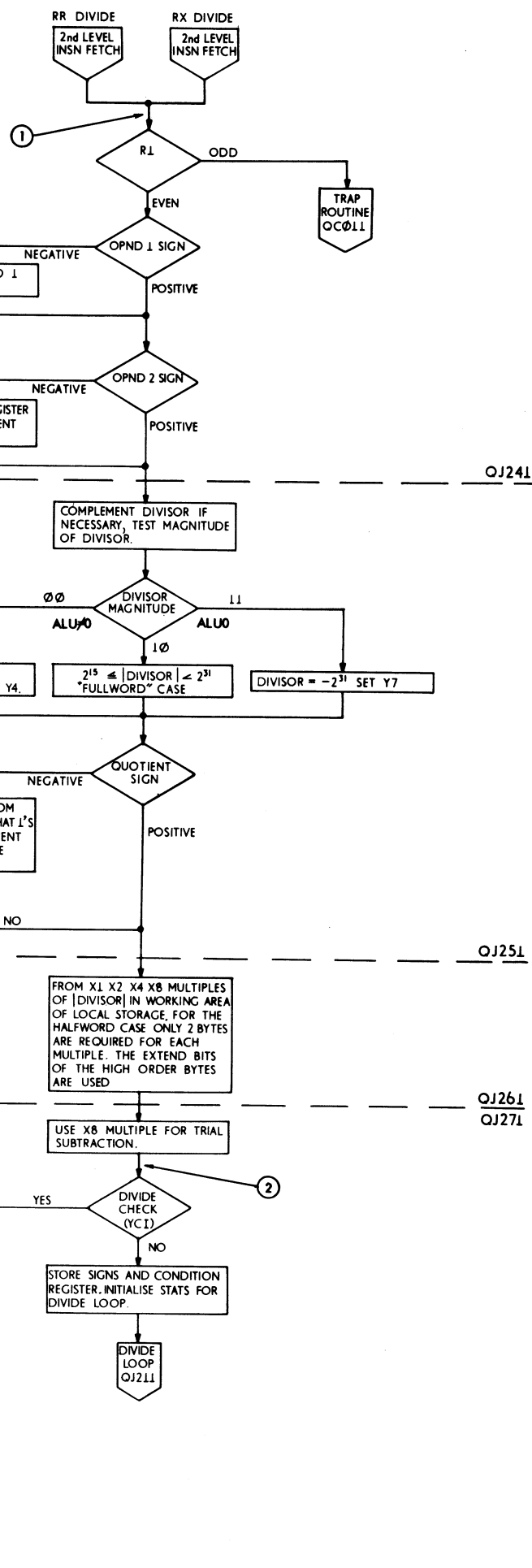


FIGURE 623 RR, RX FIXED POINT DIVIDE INITIALIZATION

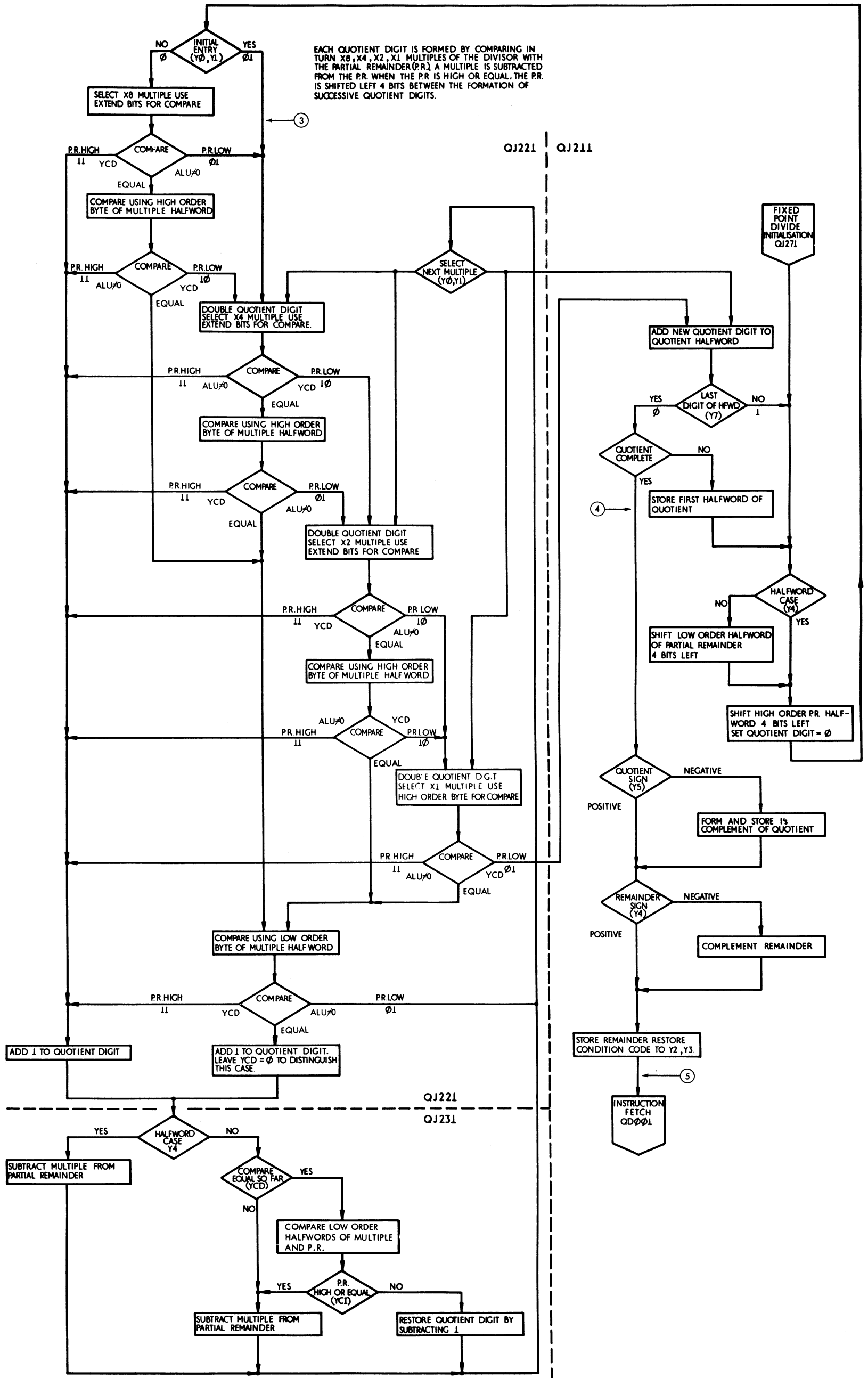
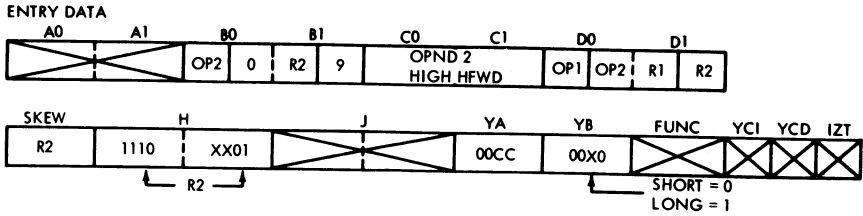
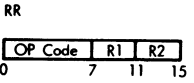


FIGURE 624. FIXED POINT DIVIDE LOOP



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

①	ROS	FP REG 00	FP REG 06	A REG	B REG	C REG	D REG
②	A45	FOFOFOFO	OF1E2D3C	60FA	0069	60FC	3006
③	A2C	FOFOFOFO	OF1Ebbbb	8FFA	2D3C	0F1E	3006
④	BF6	bbbbFOFO	OF1E2D3C	00FA	2D3C	0F1E	3006
	BEF	OF1E2D3C	OF1E2D3C	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 DESCRIPTIVE OF OBJECTIVES OF EACH INSTRUCTION SEE PRINCIPLES OF OPERATIONS MANUAL)

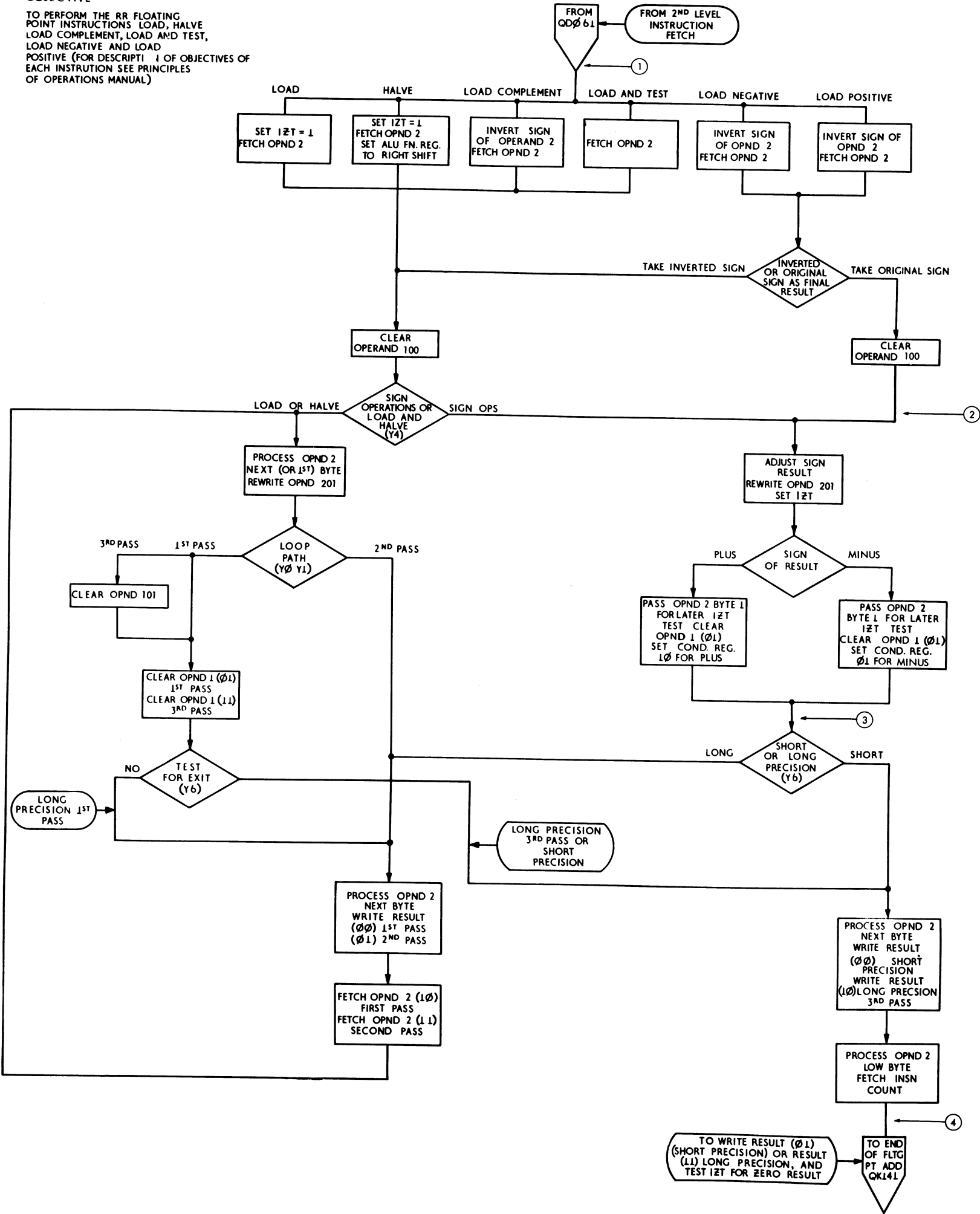
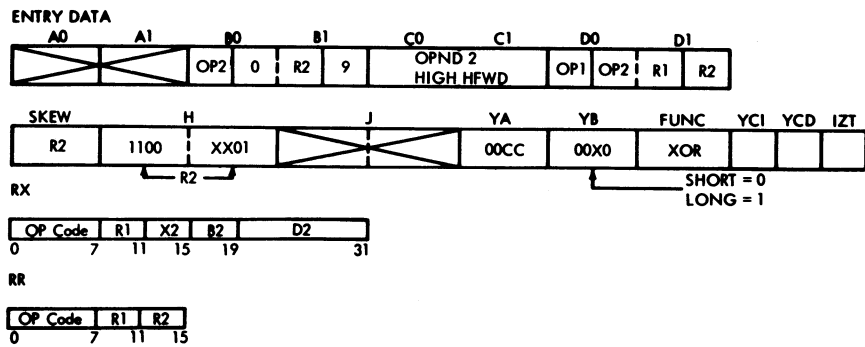


FIGURE 625. RR FLOATING POINT SIGN OPERATIONS





NO EXAMPLE SHOWN.

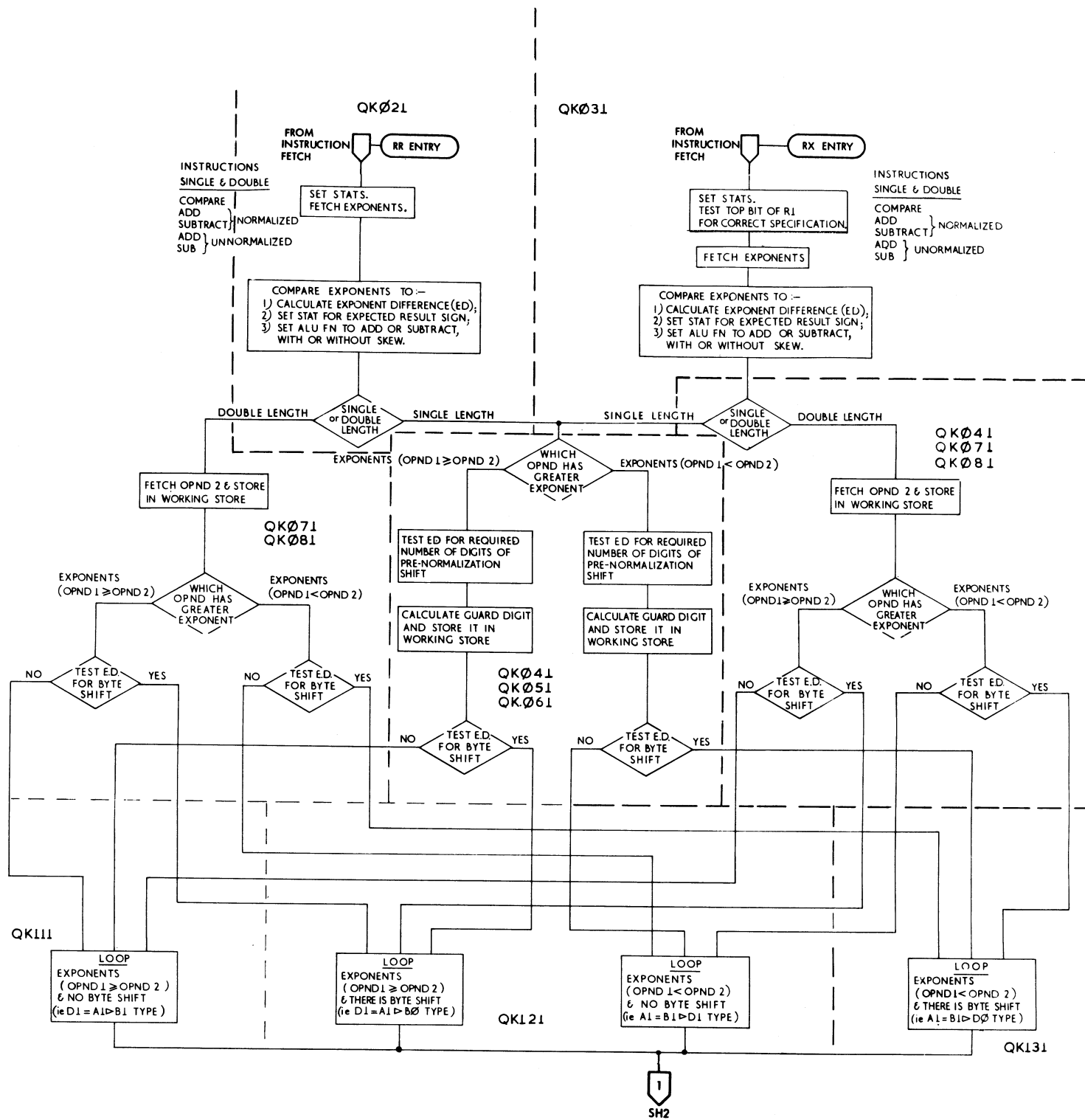


FIGURE 626. FLOATING POINT RR AND RX (QK021, QK171) (SHEET 1 OF 2)

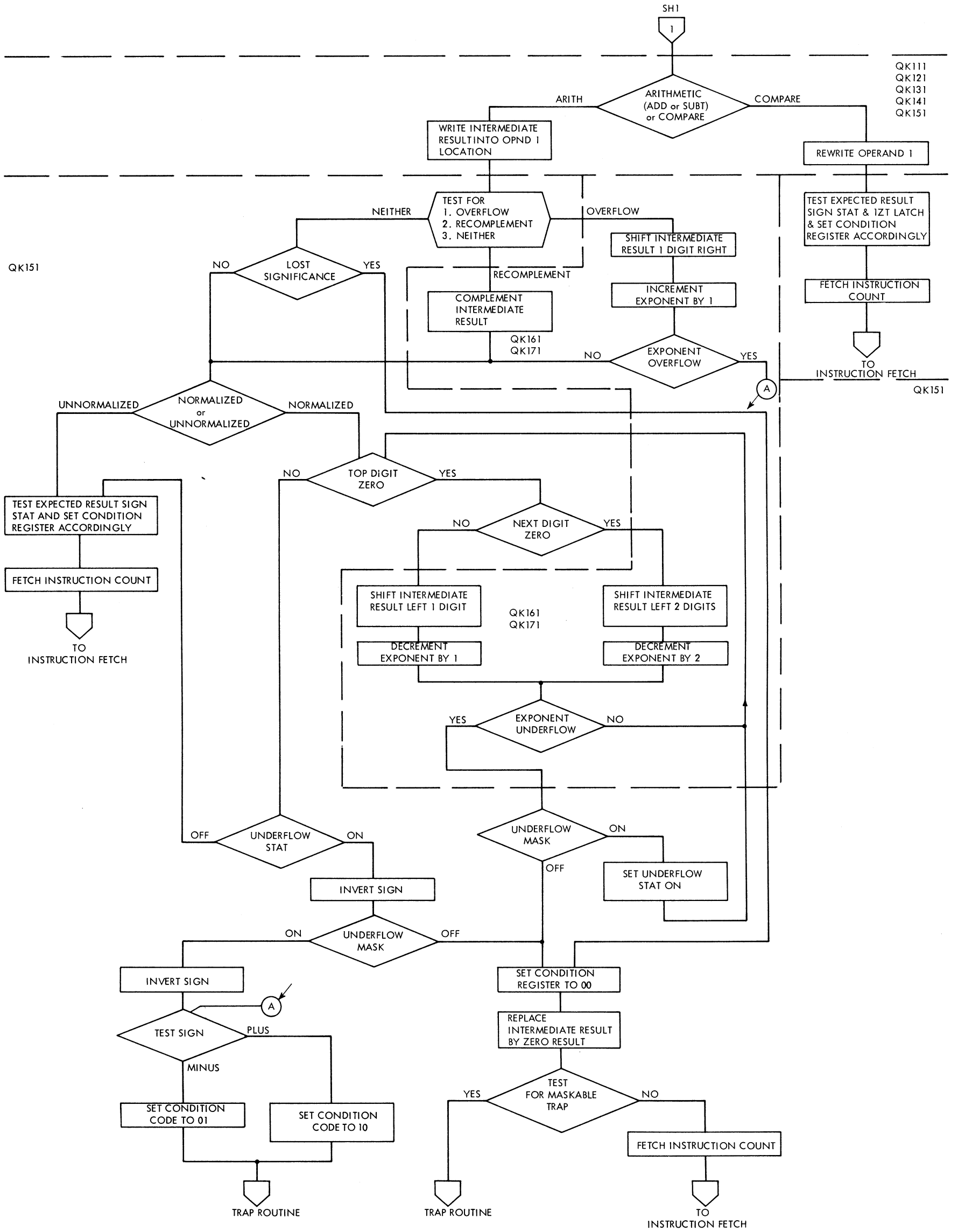
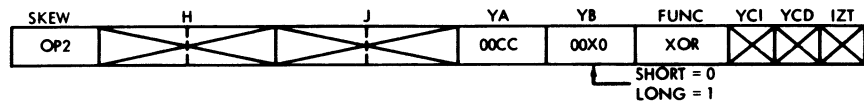
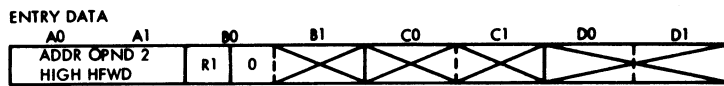
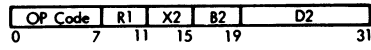


FIGURE 626. FLOATING POINT RR AND RX (QK021, QK171) (SHEET 2 OF 2)



RX



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 ADDR 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

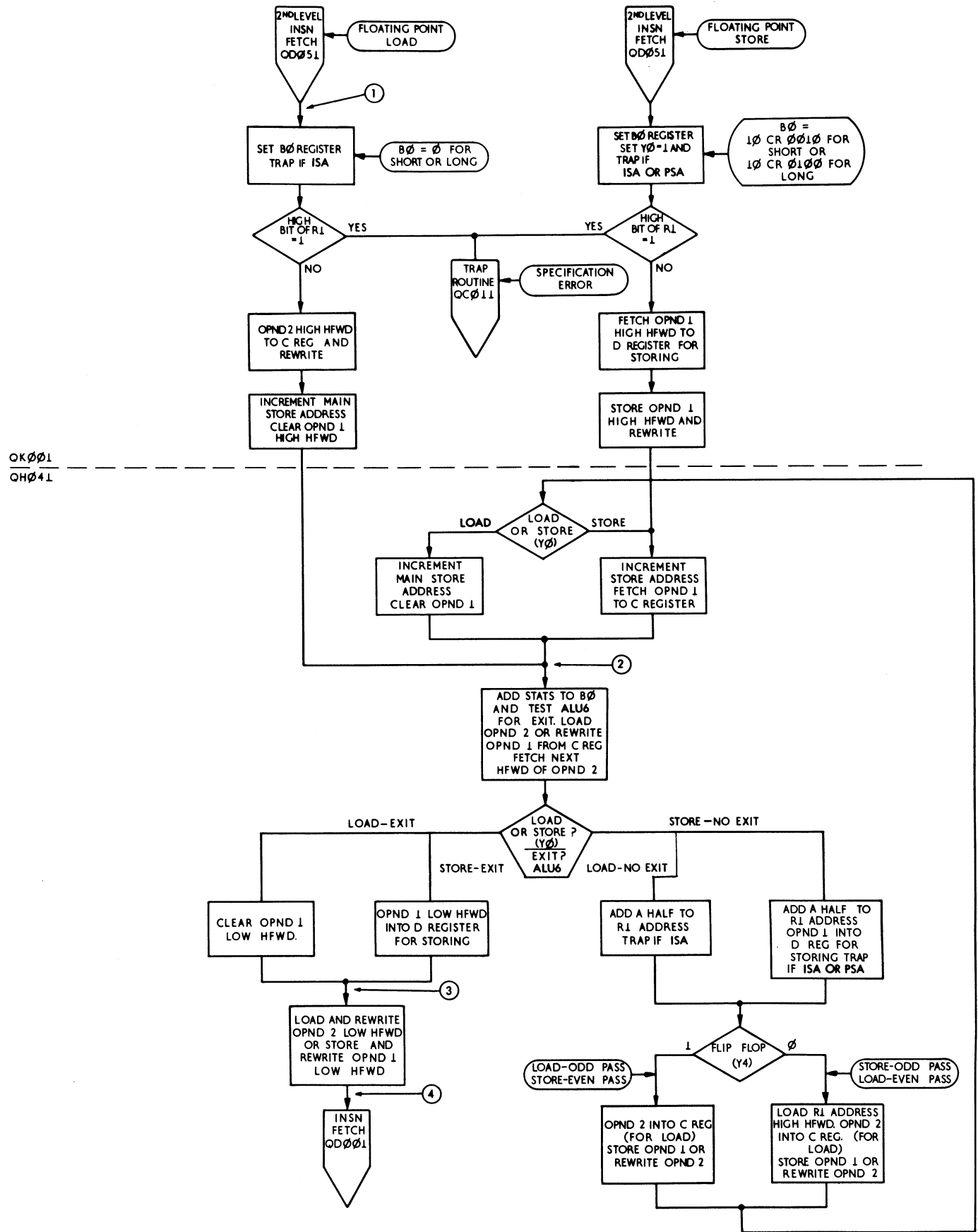
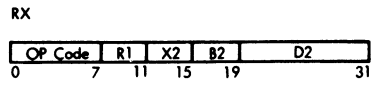
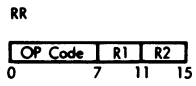
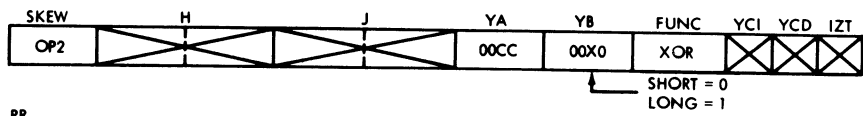
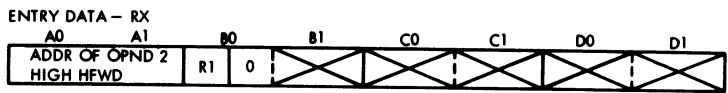
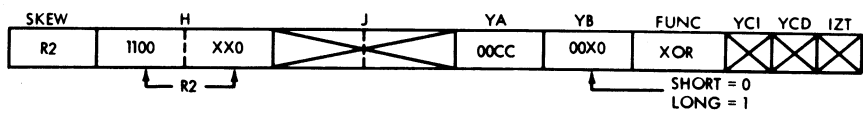
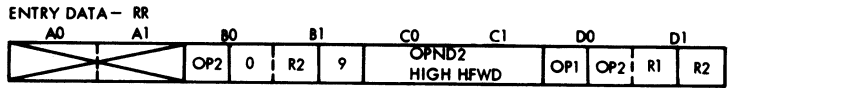


FIGURE 627, FLOATING POINT LOAD AND STORE (SHORT AND LONG)



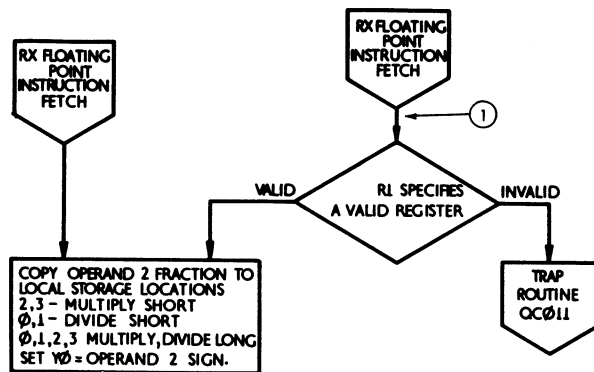
FLOATING POINT MULTIPLY, SHORT NMER DMC SECTION ICC, ROUTINE 01, STOP ON MS 60F2. THIS OPERATION WILL USE FIGURES 628 AND 629.

①	ROS	FP REG 00	FP REG 02	A REG	B REG	C REG	D REG
②	A18	45100000	45100000	60F2	C009	4510	3C20
③	AA6	45100000	45100000	60F4	2040	4510	0000
④	A87	45100000	45100000	60F4	2045	4510	4510
⑤	AD0	45100000	45100000	4AF4	0010	0020	200C
	AF5	45100000	45100000	4AF4	0010	0020	200C

FIGURE 629

⑥	8B7	45100000	45100000	0000	0060	4A10	0000
⑦	8F7	45100000	45100000	0000	0000	4510	0000
⑧	92F	45100000	45100000	0000	0000	4510	0000
⑨	930	45100000	45100000	0000	0000	0000	0000
⑩	93E	45100000	bbbbbbbbb	0000	0000	0000	4500
⑪	84C	45100000	bbbbbbbbb	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



OBJECTIVE  
TO SHOW THE INITIALISATION FOR FLOATING POINT DIVIDE/MULTIPLY

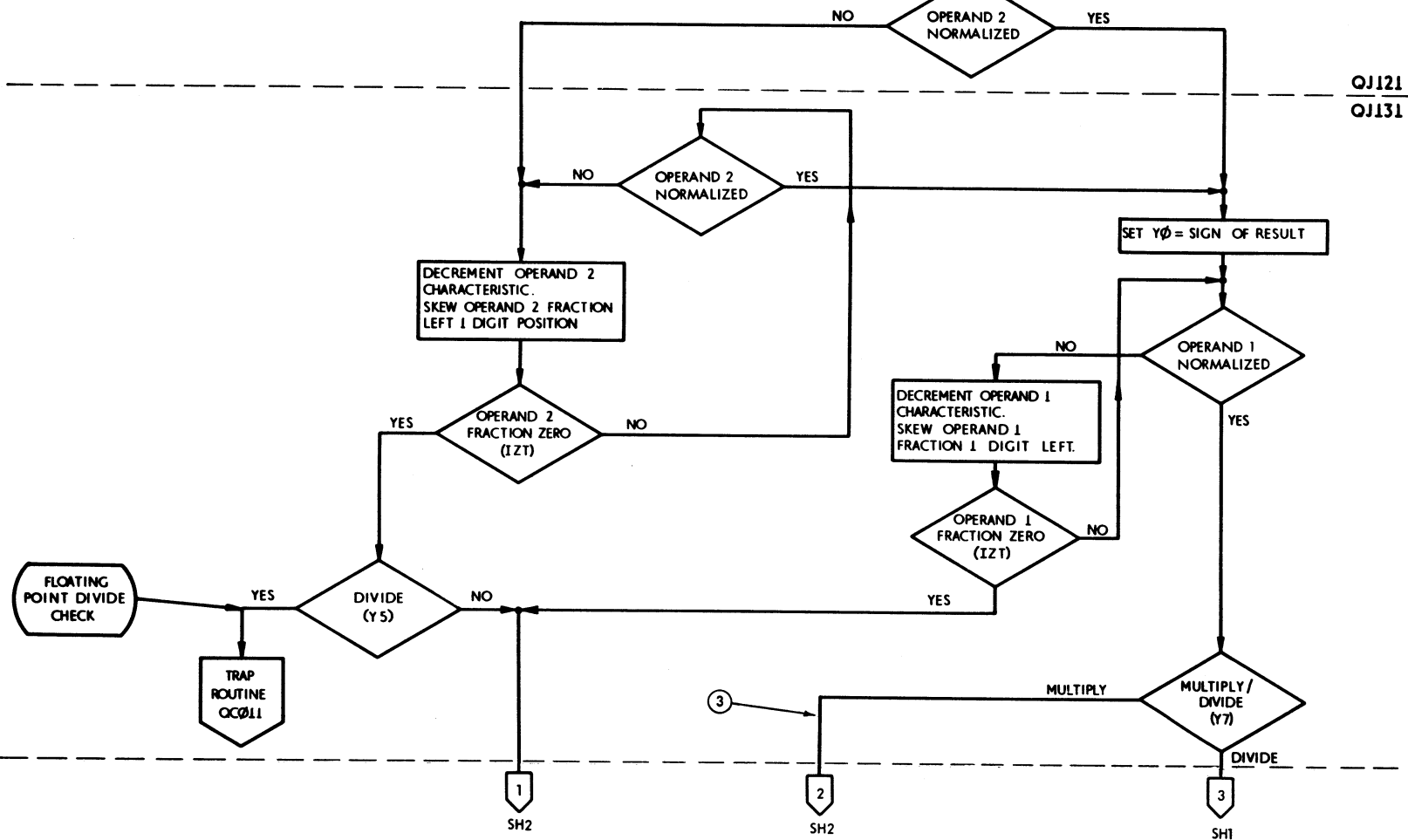


FIGURE 628. FLOATING POINT MULTIPLY/DIVIDE INITIALIZATION (SHEET 1 OF 2)

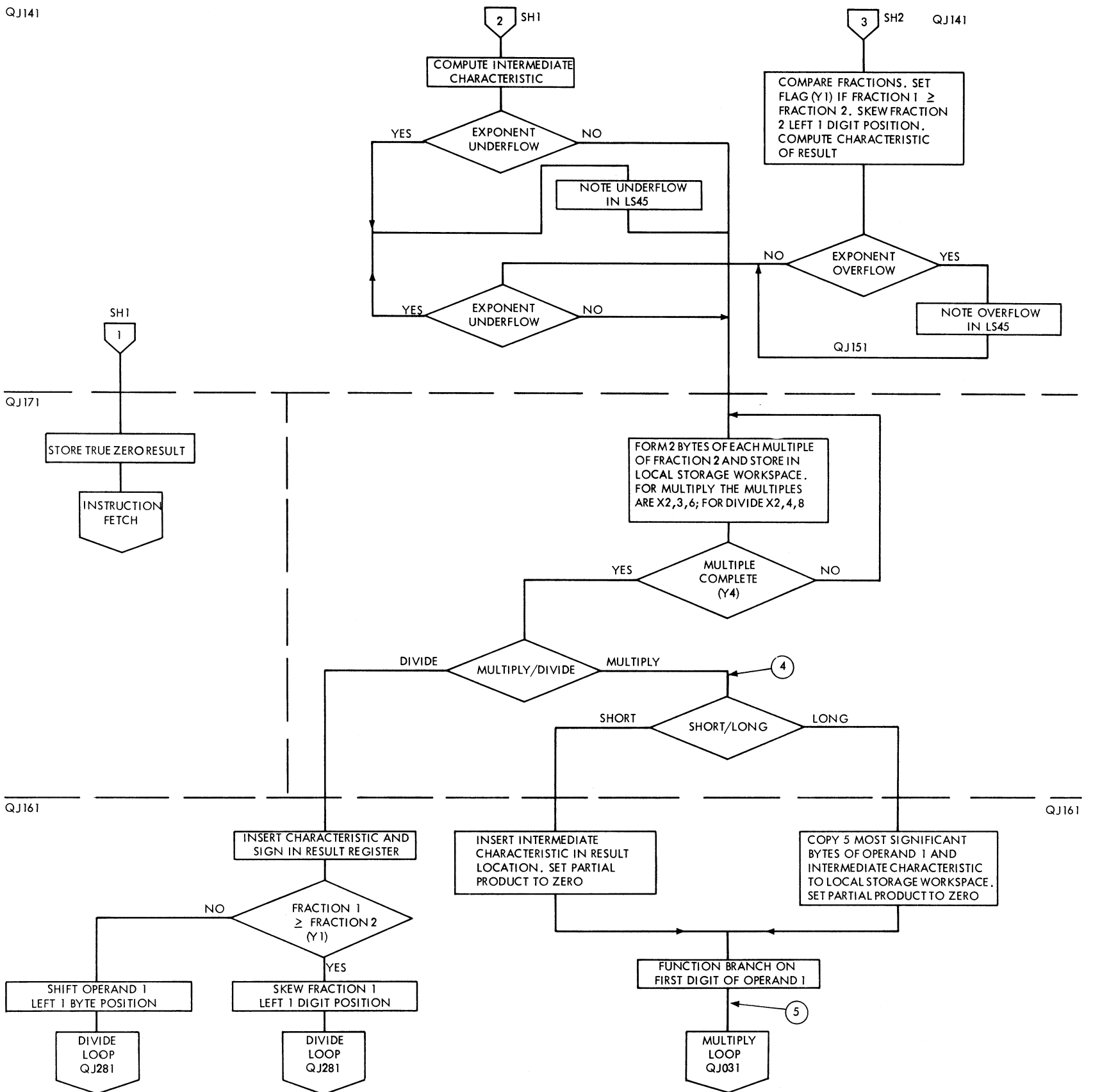


FIGURE 628. FLOATING POINT MULTIPLY/DIVIDE INITIALIZATION (SHEET 2 OF 2)

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.

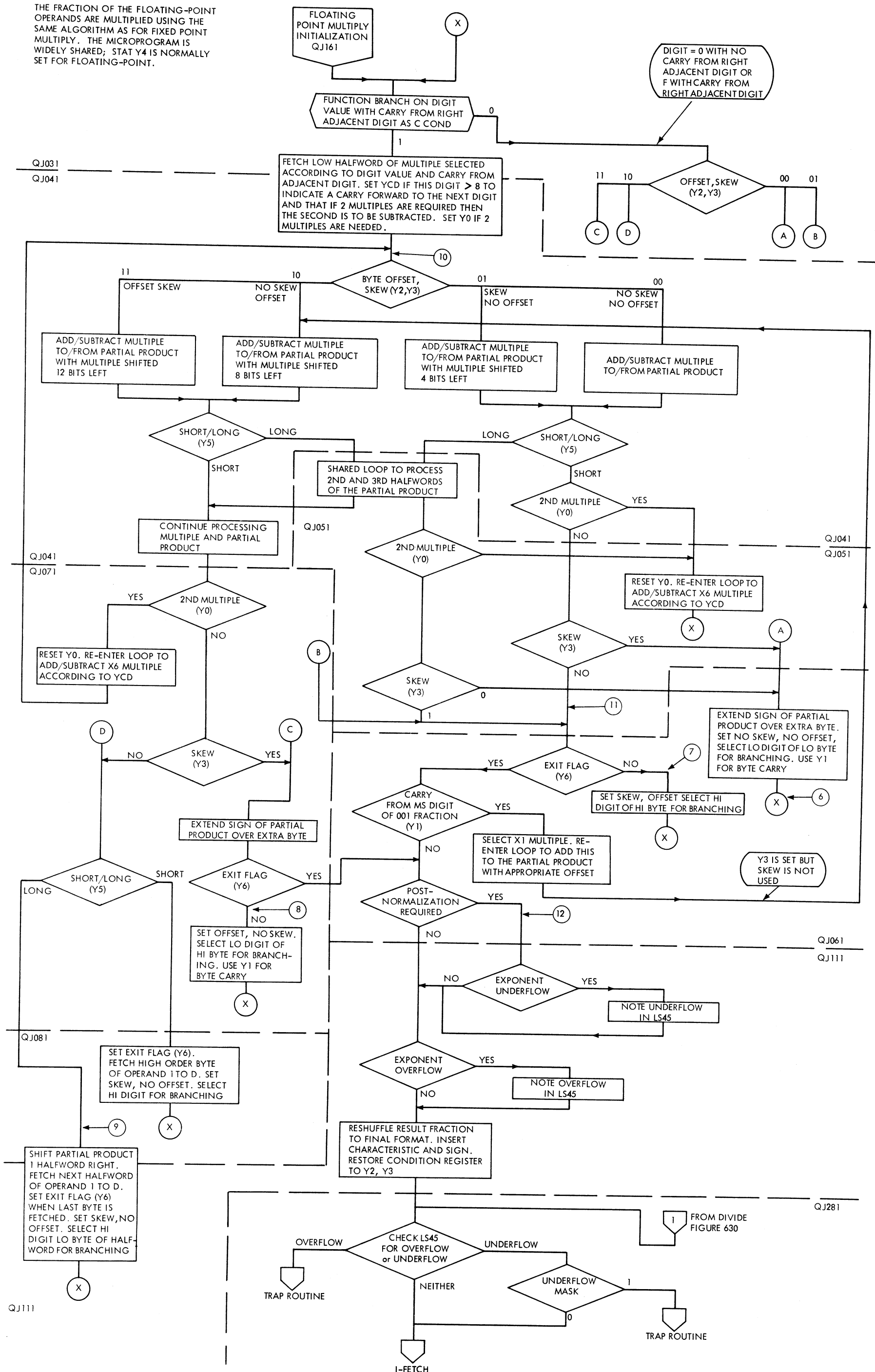


FIGURE 629. FLOATING POINT MULTIPLY LOOP

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.

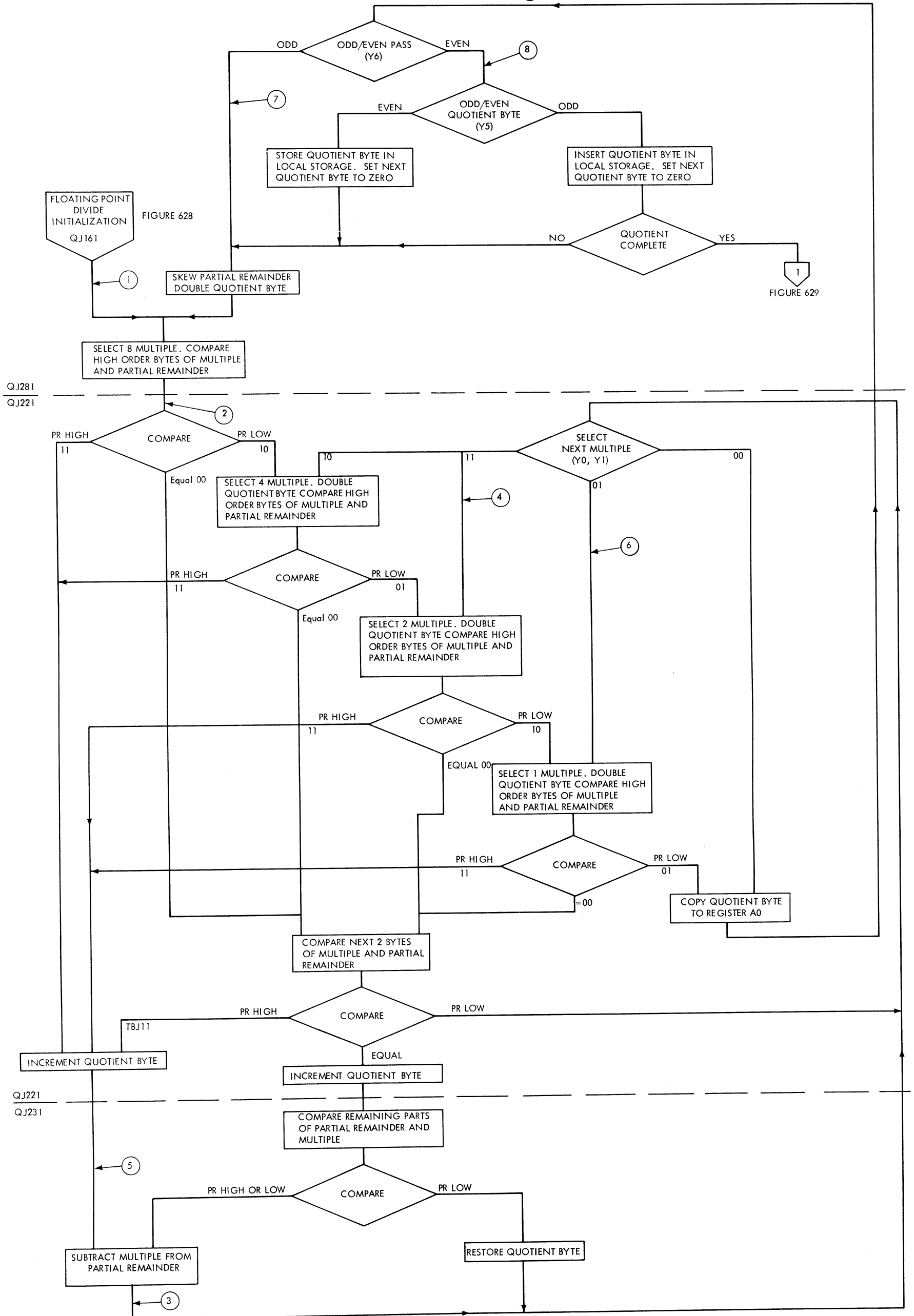
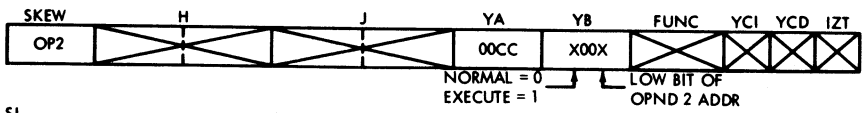
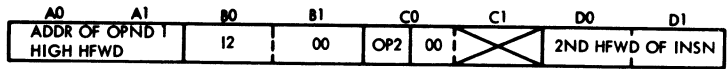
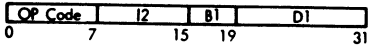


FIGURE 630. FLOATING POINT DIVIDE LOOP

ENTRY DATA



S1



OBJECTIVES

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

TEST UNDER MASK

ROS	REG 6	REG 7
① 343	00FFFFFF	FFFFFFF
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

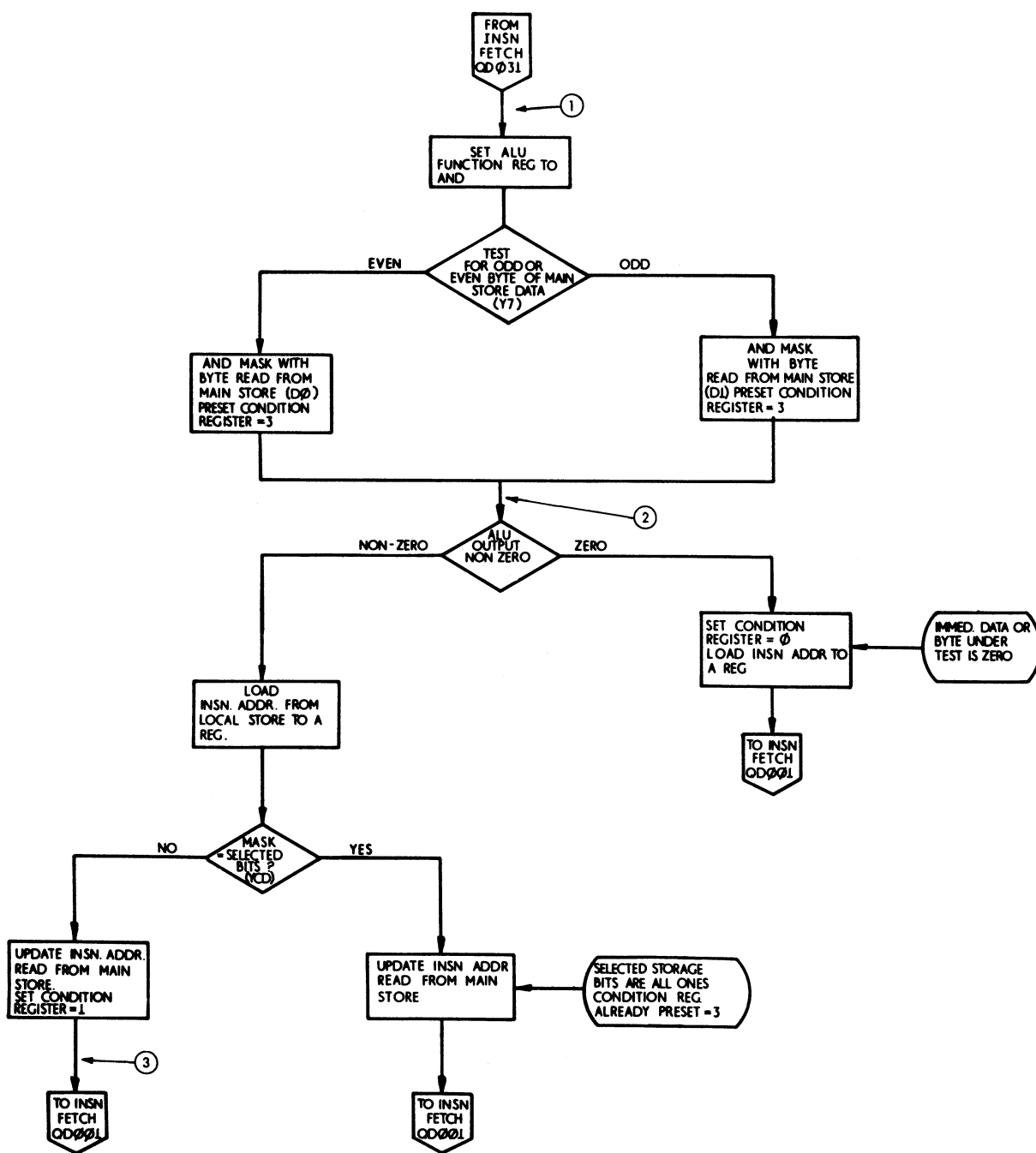
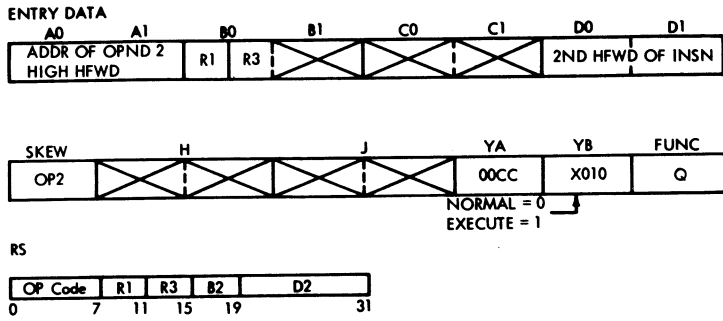


FIGURE 631 .TEST UNDER MASK (QP001)





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	7FFFbfff	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

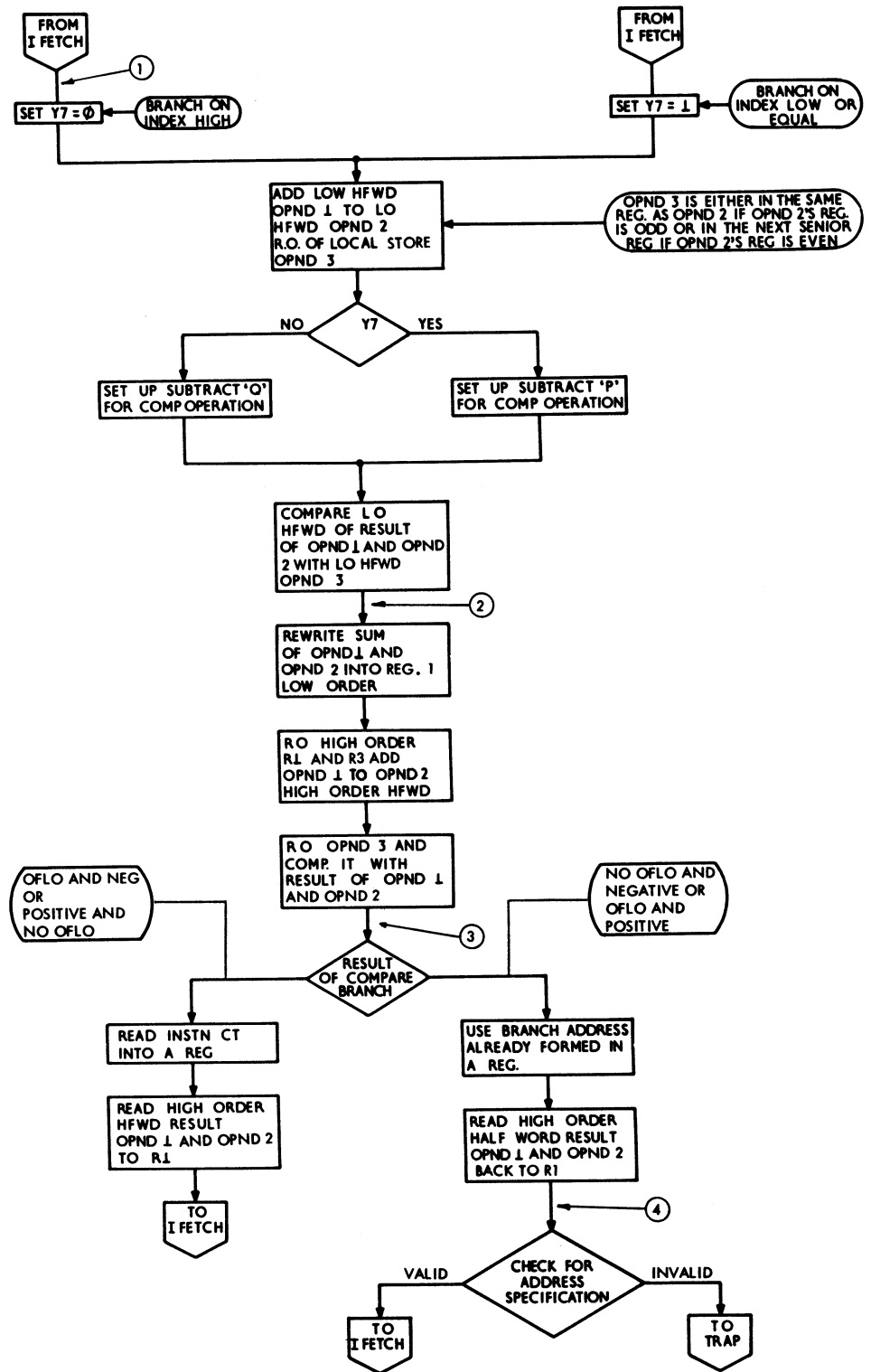
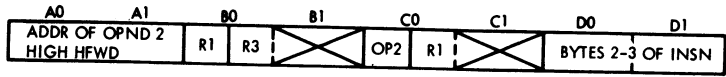


FIGURE 632. BRANCH ON INDEX (QE041)

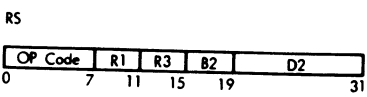
ENTRY DATA



NO EXAMPLE SHOWN.



NORMAL = 0  
EXECUTE = 1



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

- OBJECTIVE
- TO ALTER SYSTEM MASK PSW BITS 0-7
  - READ SECOND OPERAND FROM MAIN STORAGE
  - PLACE CONTENTS OF 2ND OPND INTO PSW BITS 0-7

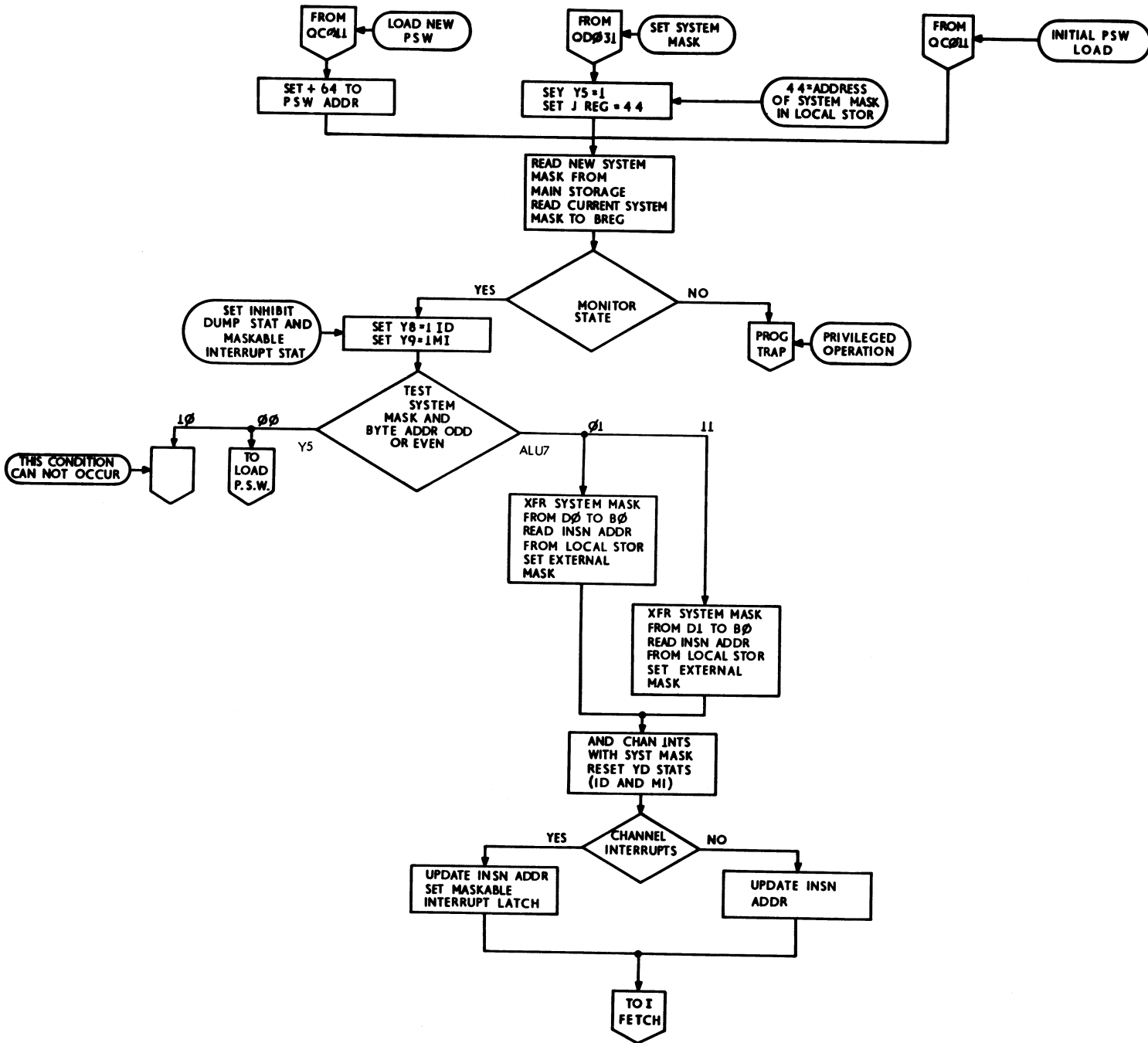
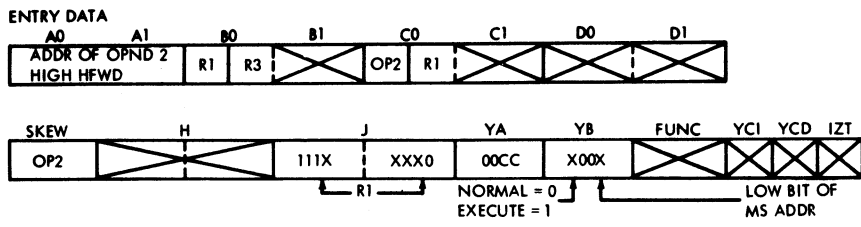
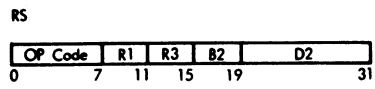


FIGURE 633. SET SYSTEM MASK (QC021)



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

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

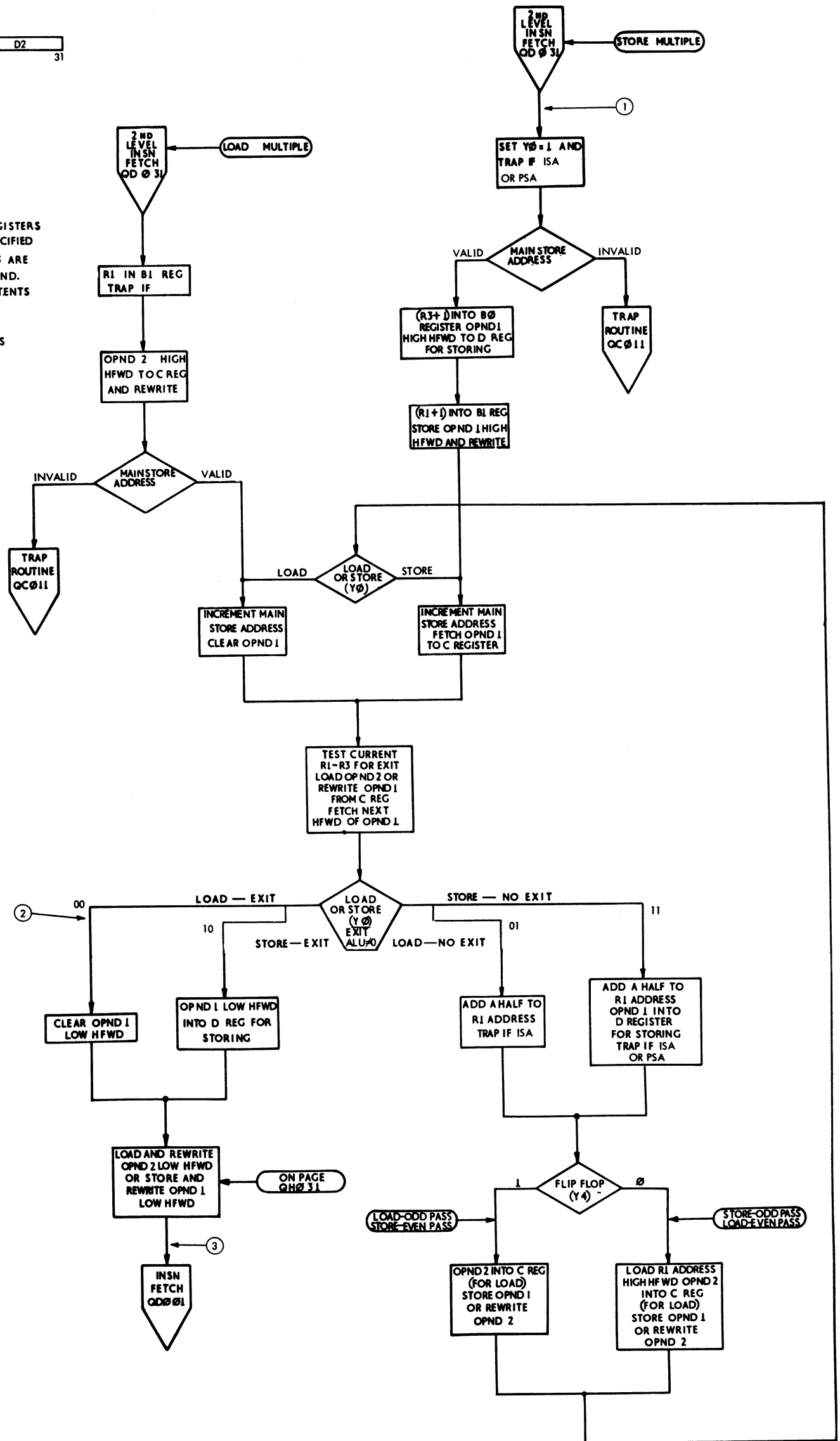
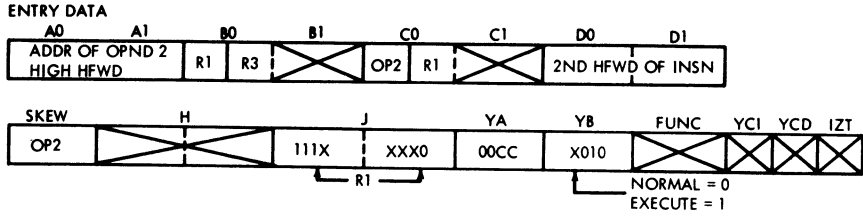
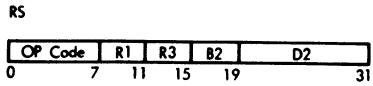


FIGURE 634. RS LOAD AND STORE MULTIPLE (QH041)



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 $\mu$ PROG BY
1L	SHIFT LEFT 1	LEFT SHIFT FUNCTION
4L	SHIFT LEFT 4	PASS 0 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 0 FUNCTION WITH SKEW TOGETHER WITH HALFWORD SHIFT TO RIGHT (LOOP 2), ALL FOLLOWED BY RIGHT SHIFT FUNCTION (LOOP 6)

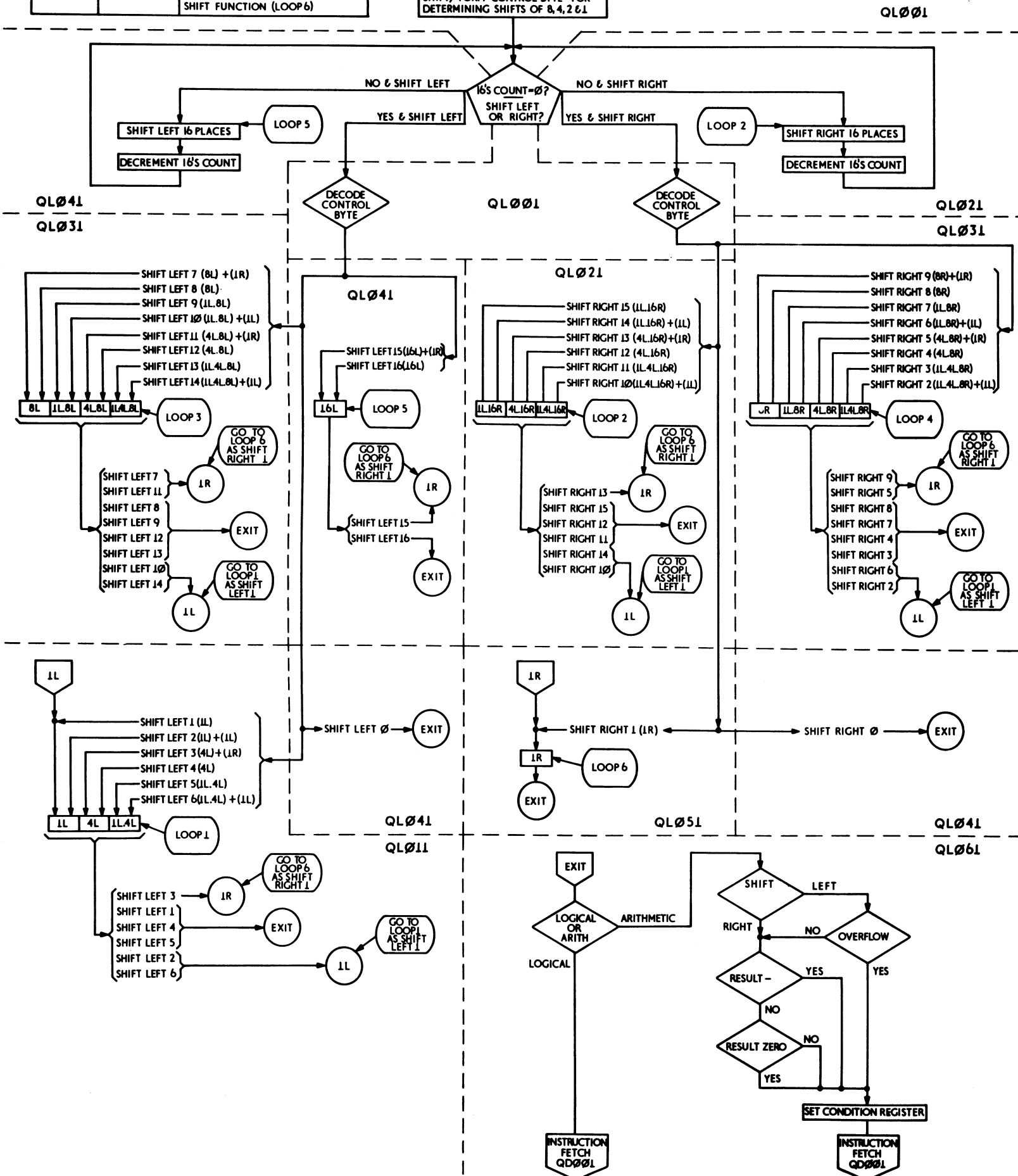
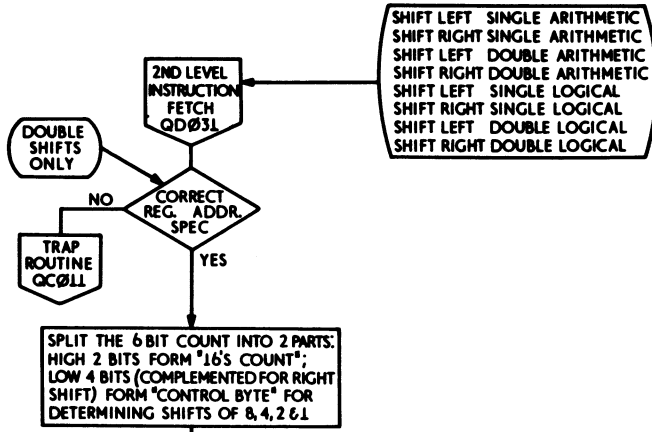
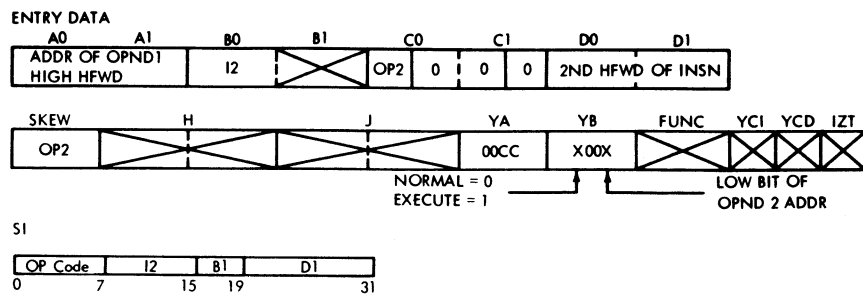


FIGURE 635. SHIFTS



AND NI	DMC	SECTION 137, ROUTINE 01			
①	ROS	A REG	B REG	C REG	D REG
②	349	032C	0000	4003	0000
③	368	032C	00FF	4003	0000
	368	01C0	DFFF	4D01	4400

**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	0CRX010	11111111	00000000	00000000
YCD AT POINT A	0	1	0	0
Y STATS AT POINT B	0CRX010	00000000	00000000	00000000
Y STATS AT POINT C	0CRX010	00010000	00010000	00010000

0 RESULT  
CR = 00

1 RESULT  
CR = 01

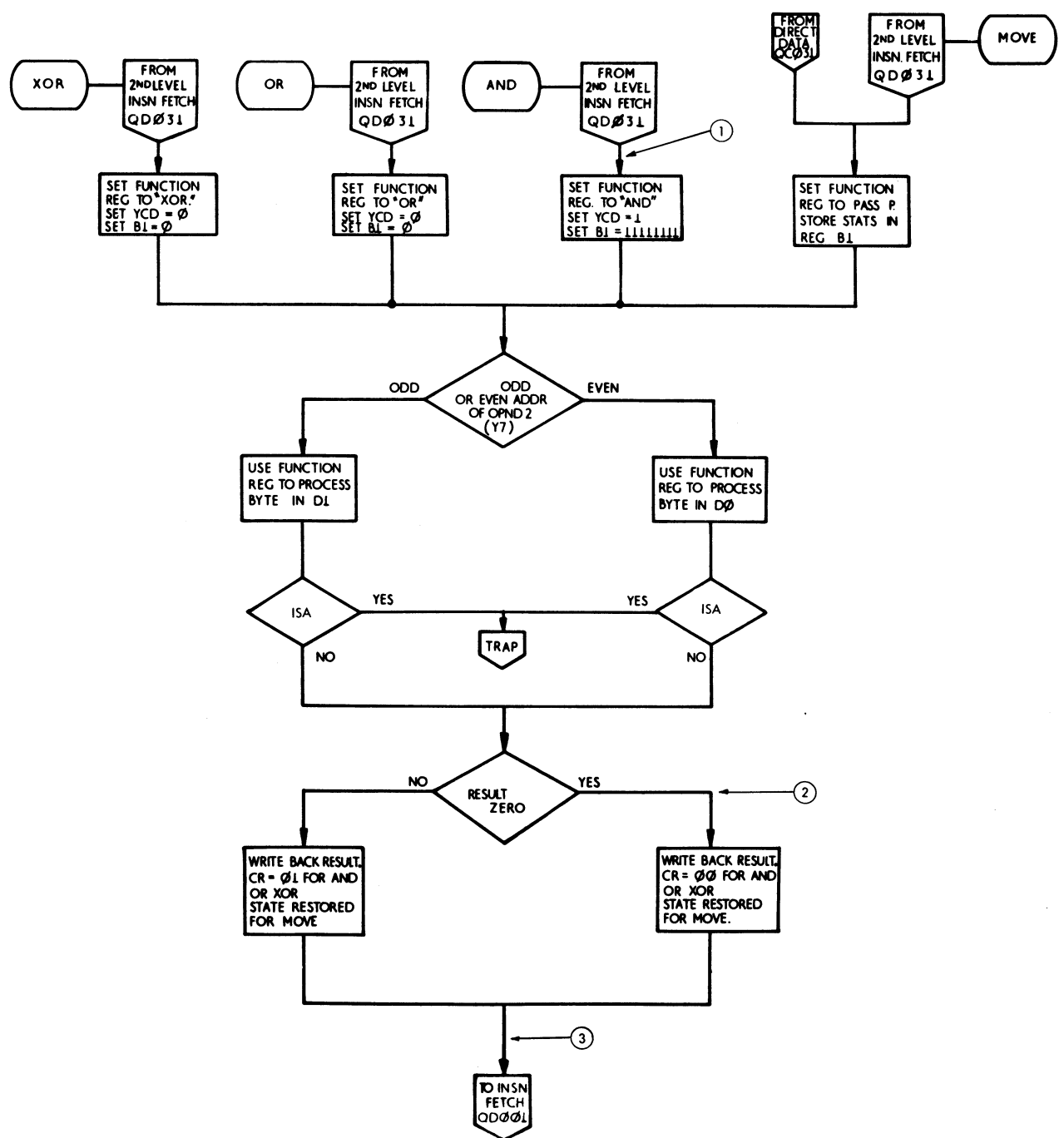
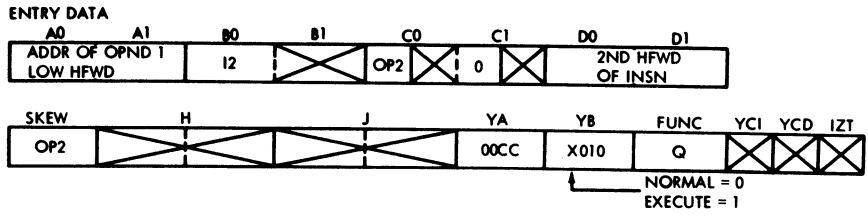
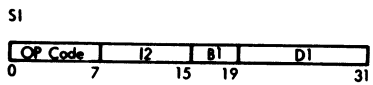


FIGURE 636 SI OPERATIONS, AND, OR, EXOR, MOVE (QP001)



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.

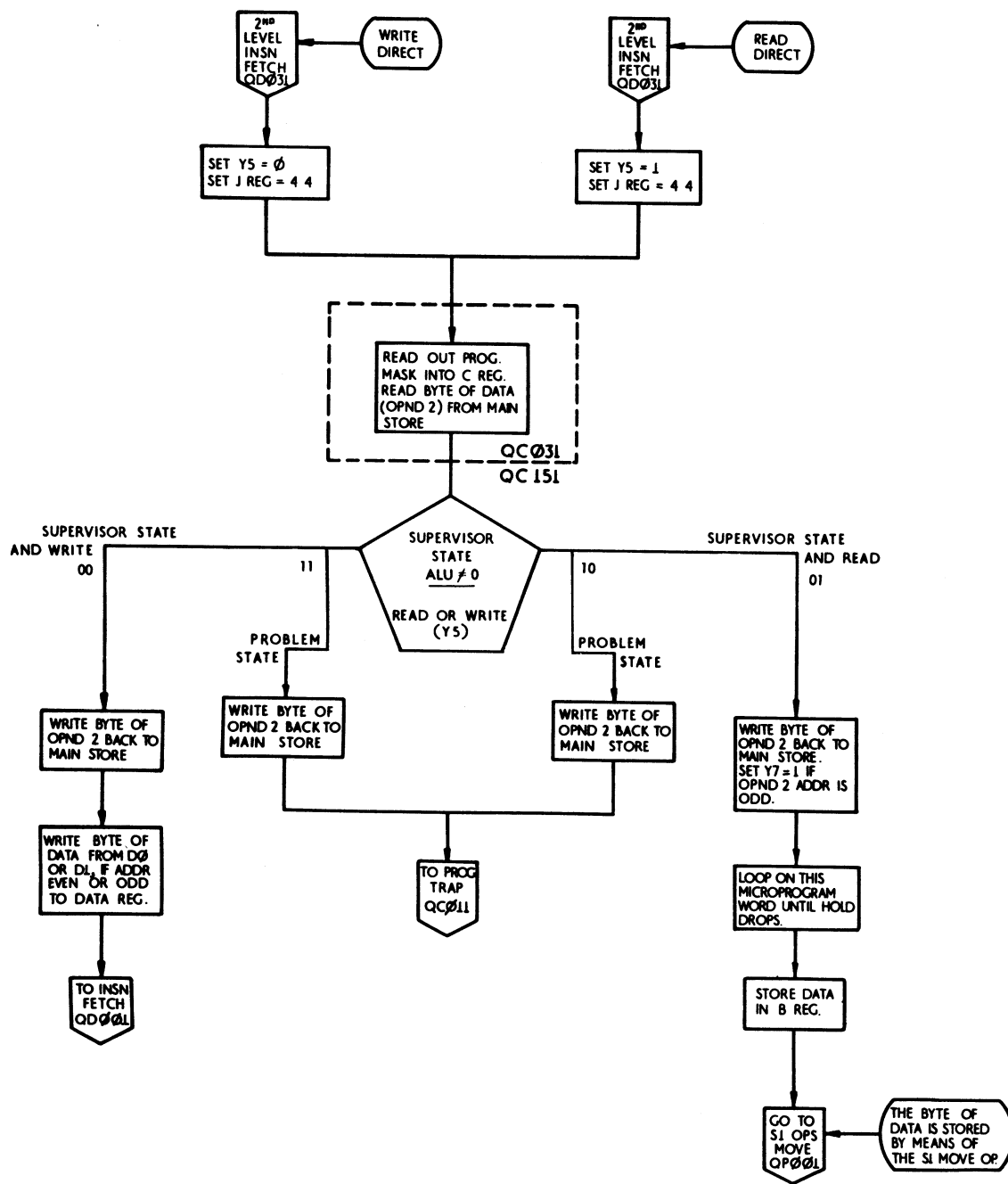
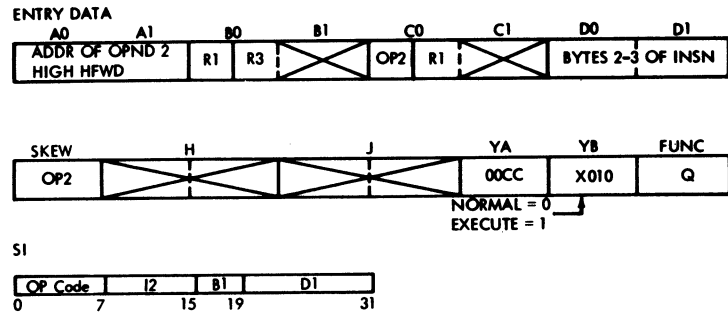


FIGURE 637, READ DIRECT AND WRITE DIRECT (QC031)



LOAD PSW	LPSW	DMC	SECTION 124, STOP AT MS 641E			
			A REG	B REG	C REG	D REG
①	ROS		6050	0000	2000	F050
②	344		65E0	0004	0000	49D8
③	4CC		0834	0004	0000	0834
④	106		0CAC	3000	3004	0CAC
	12C					

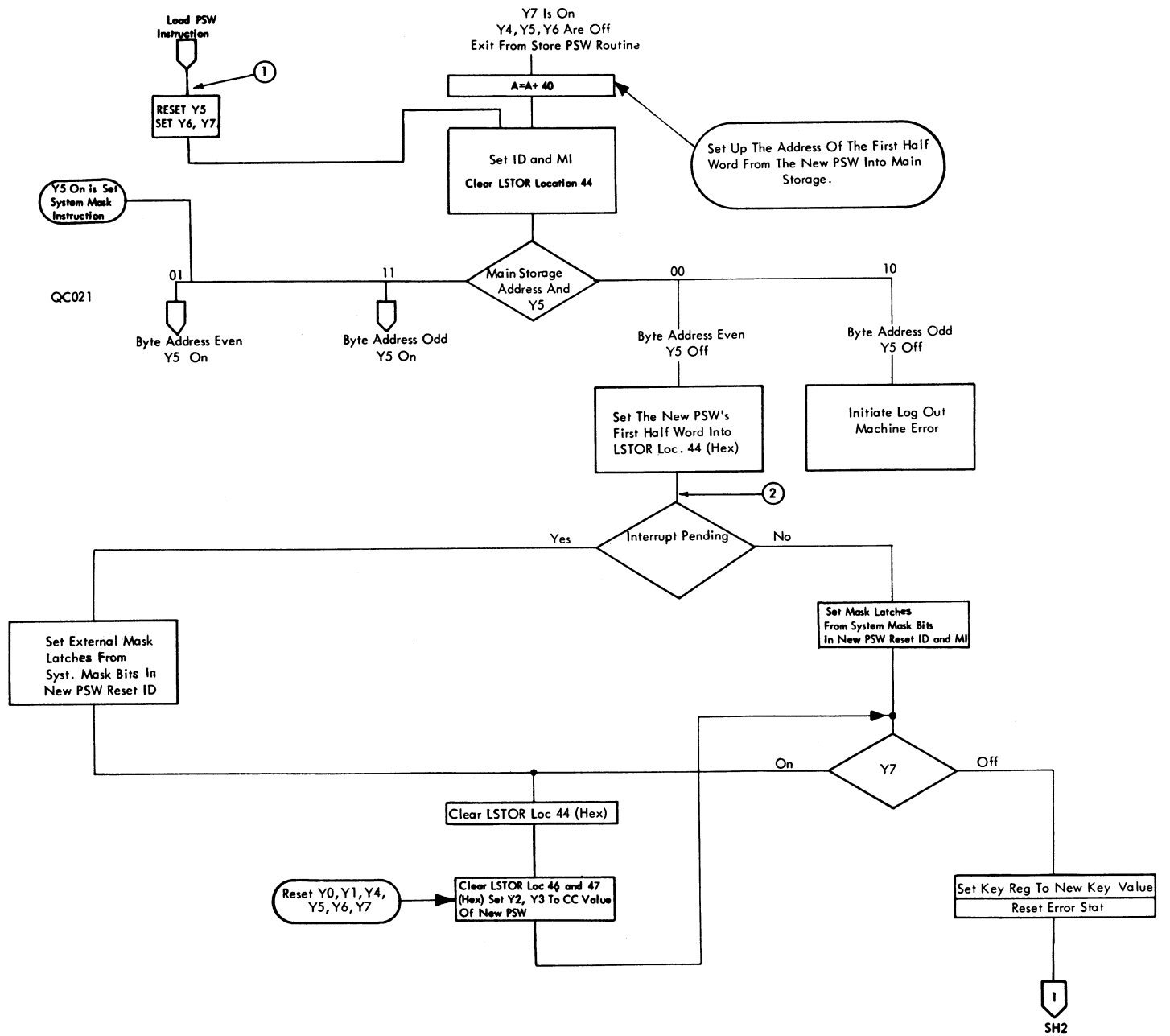


FIGURE 638. LOAD PSW (SHEET 1 OF 2)

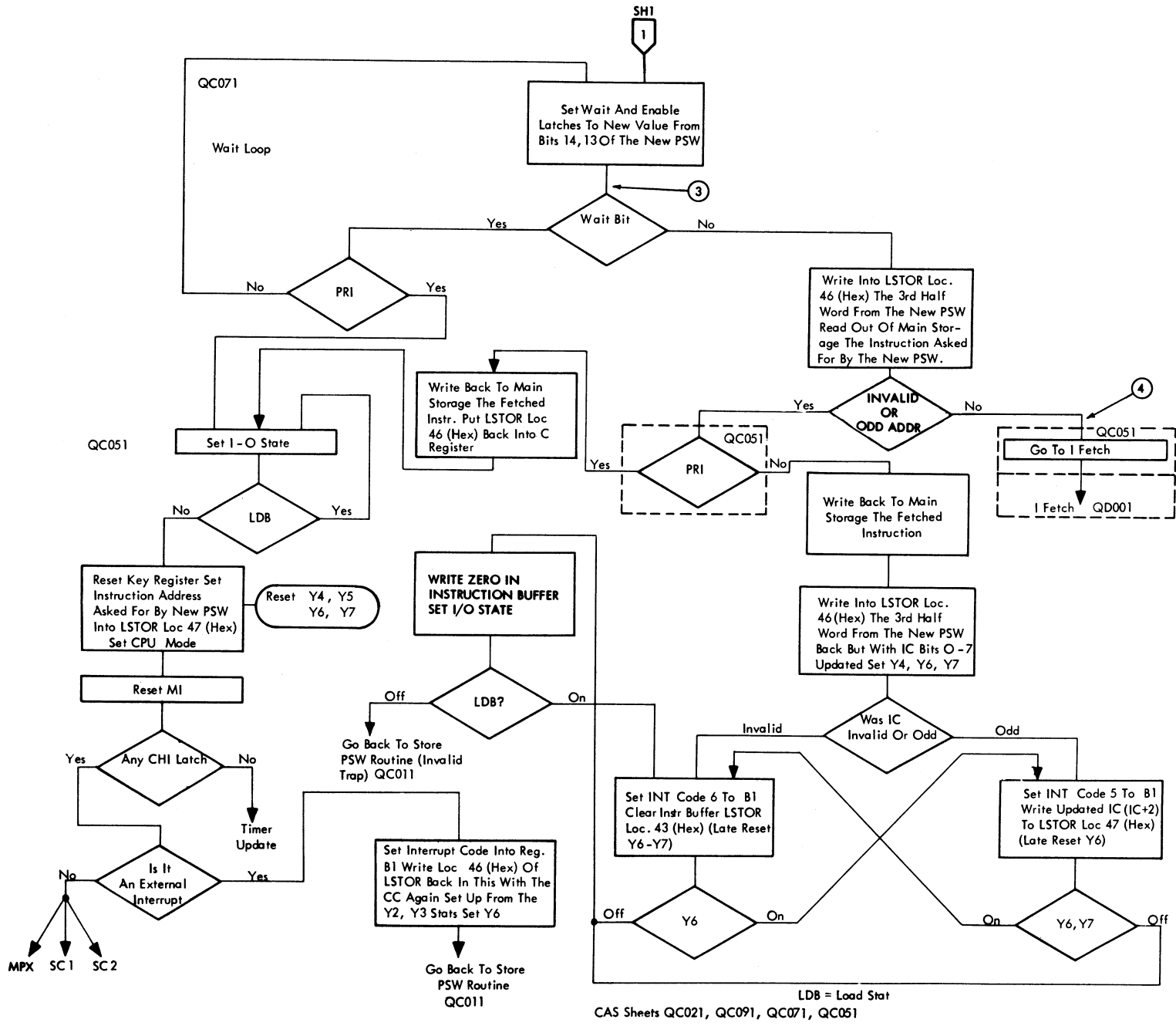
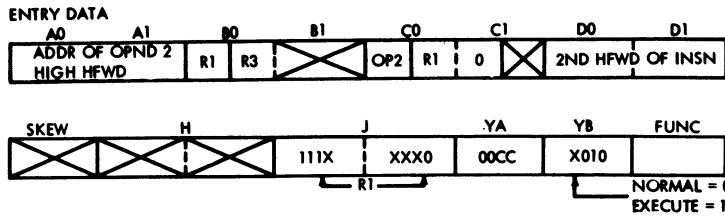
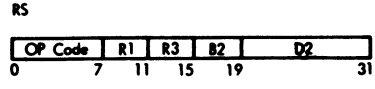


FIGURE 638 LOAD PSW (SHEET 2 OF 2)





NO EXAMPLE SHOWN.



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)	FUNCTION
0000	0001	XFER LOCAL STORE TO LOCAL STORE TO ONE EXTENDED WORD
0000	0011	XFER LOCAL STORE TO LOCAL STORE FROM ONE EXTENDED WORD
0000	1001	CLEAR A LOCAL STORAGE WORD
0100	1000	XFER A LOCAL STORAGE WORD TO BUMP
0100	0000	XFER BUMP TO LOCAL STORE
0100	0010	CYCLE A BUMP WORD
0000	0110	CYCLE SPLS WORD CPU
0100	0110	CYCLE SPLS WORD MPX

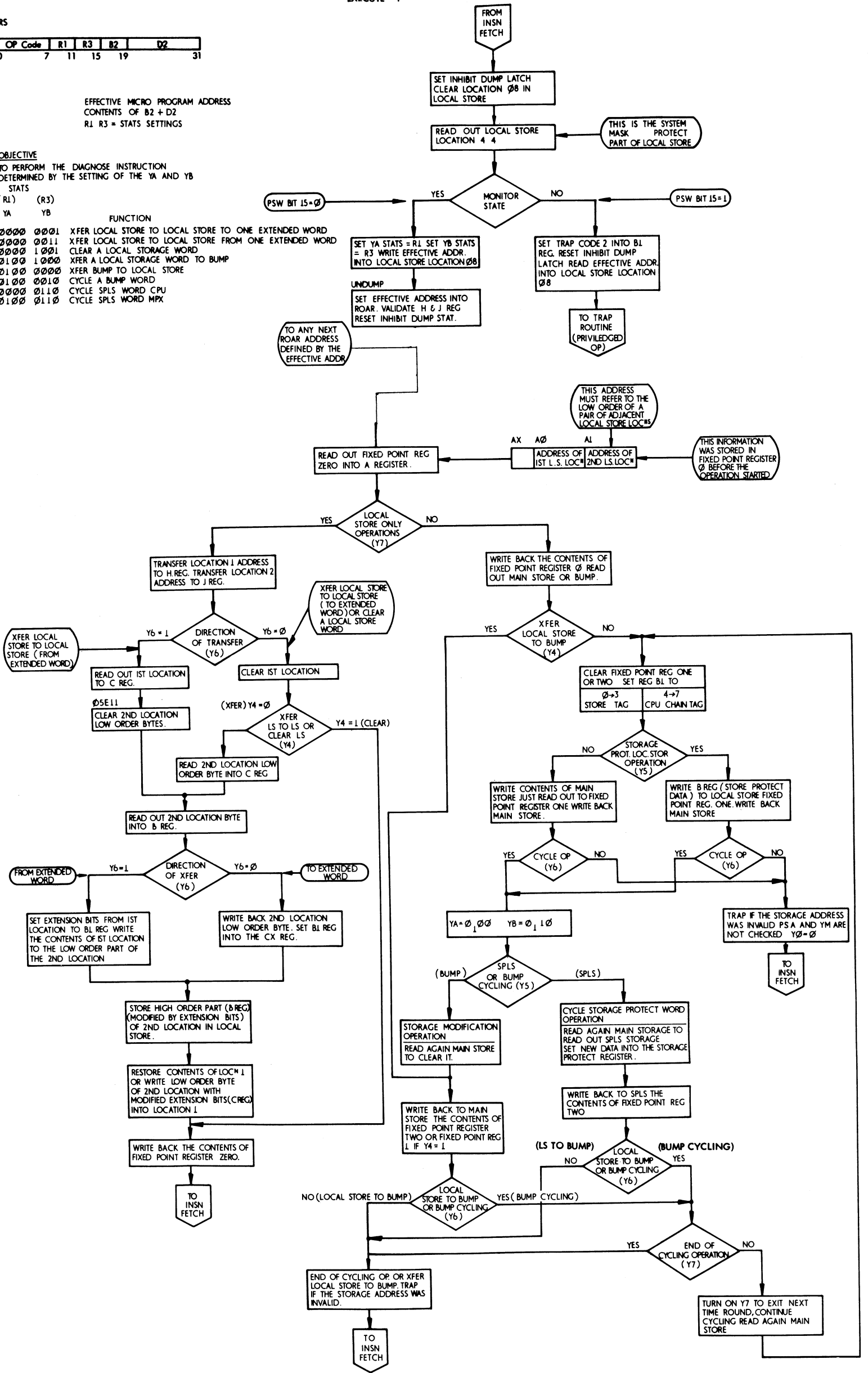
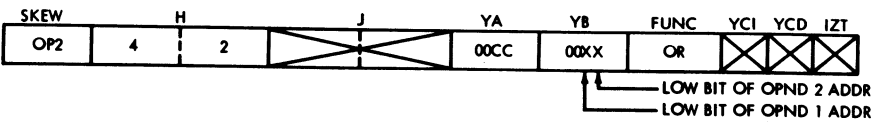
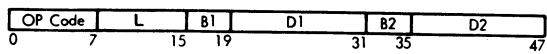


FIGURE 639, DIAGNOSE INSTRUCTION (QS 101)

ENTRY DATA



SS



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

THEN STOP ON MS 617A

①	ROS	A REG	B REG	C REG	D REG
②	C18	6A68	0000	6868	FFFF
③	C70	6868	0068	6768	33FE
④	CD4	6798	0098	3768	33FE
	C7C	6868	FF68	67CC	CCFE

OBJECTIVE

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

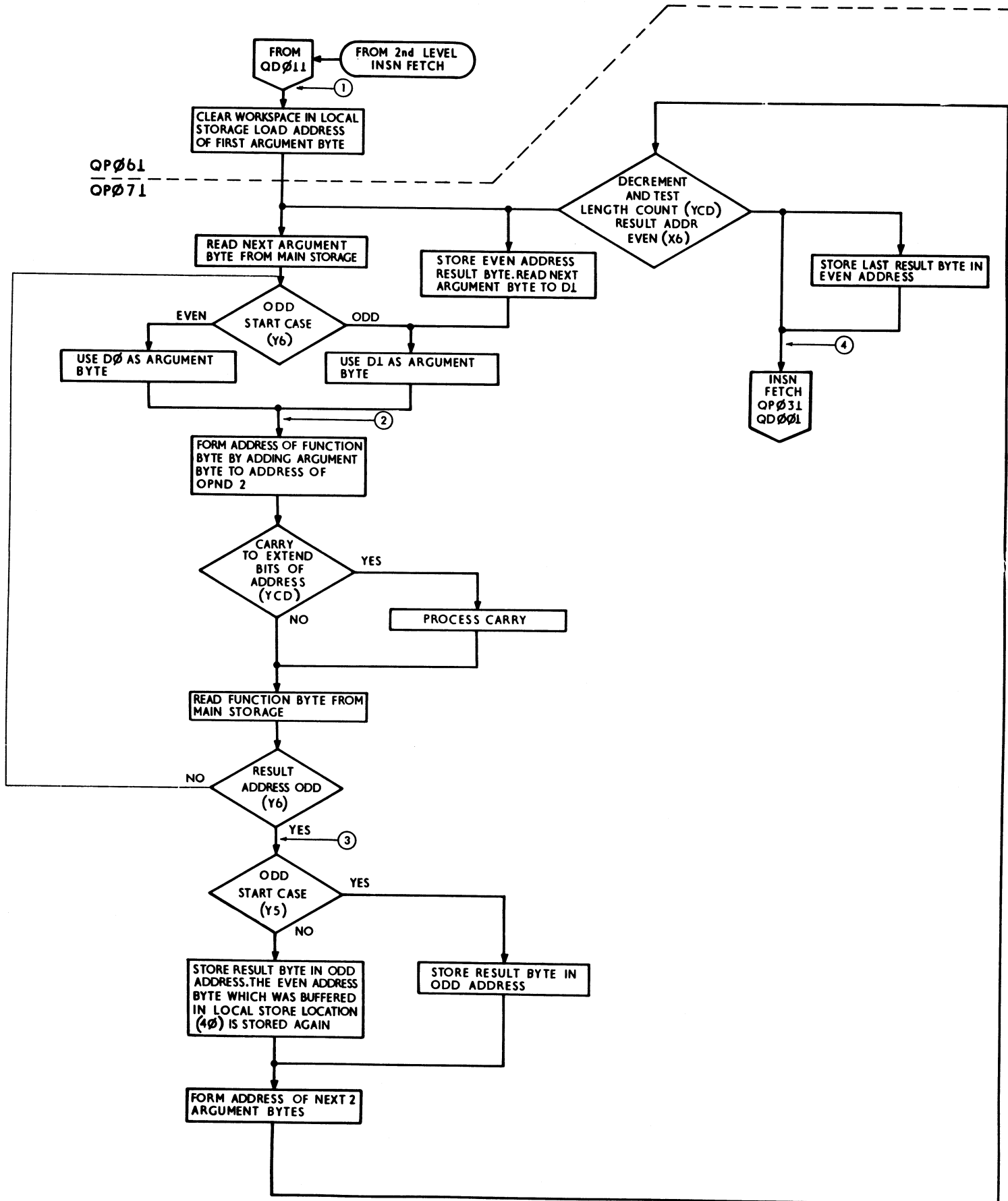
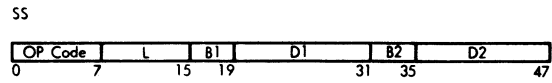
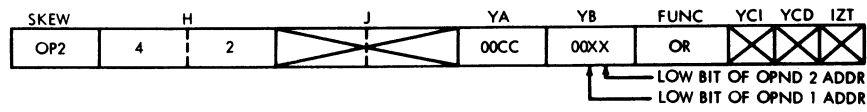
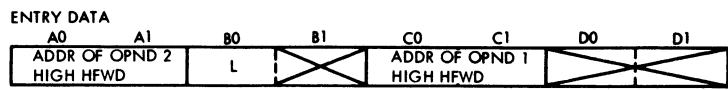


FIGURE 641.SS TRANSLATE



TRANSLATE AND TEST TRT DMC SECTION 185,  
ROUTINE 02 . STOP ON MS 6182

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

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.

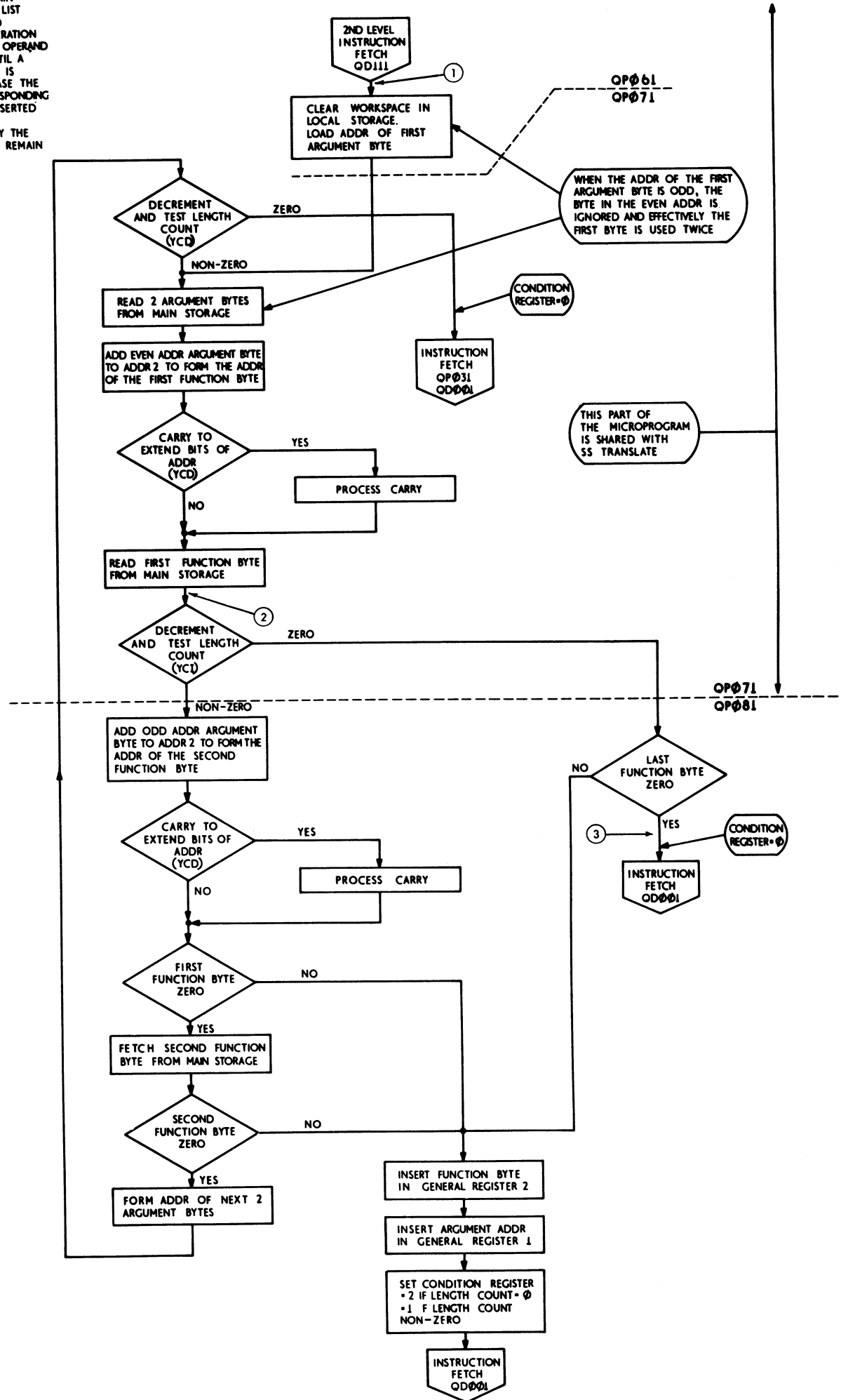
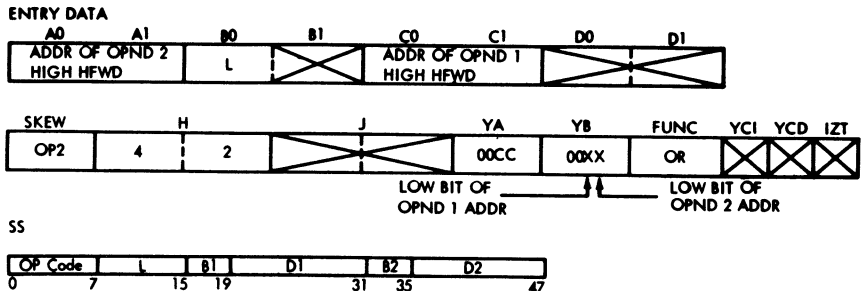


FIGURE 642 SS TRANSLATE AND TEST



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

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

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

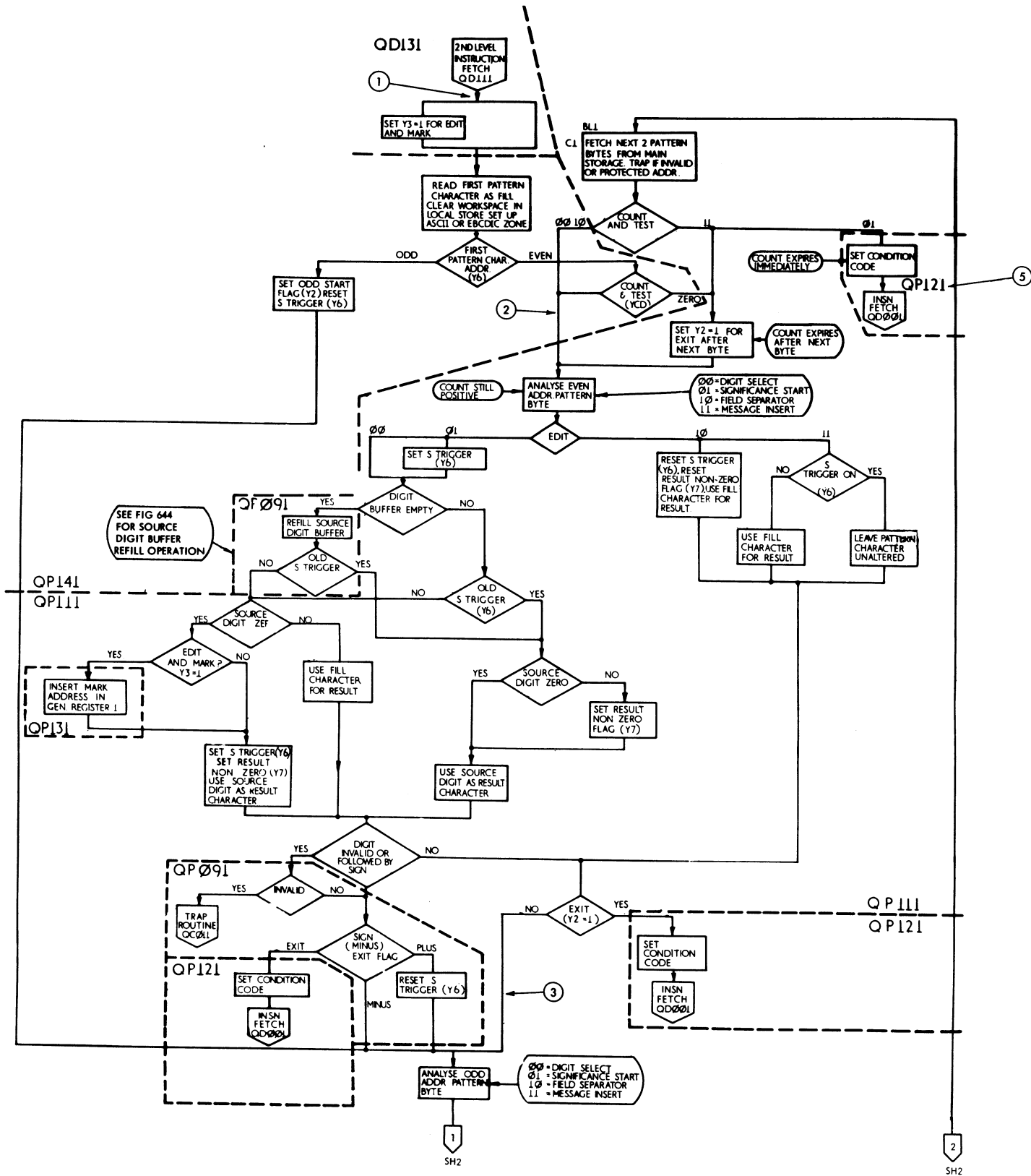


FIGURE 643 EDIT, EDIT AND MARK (QP091, QP101, QP121, QP131, QP141) (SHEET 1 OF 2)

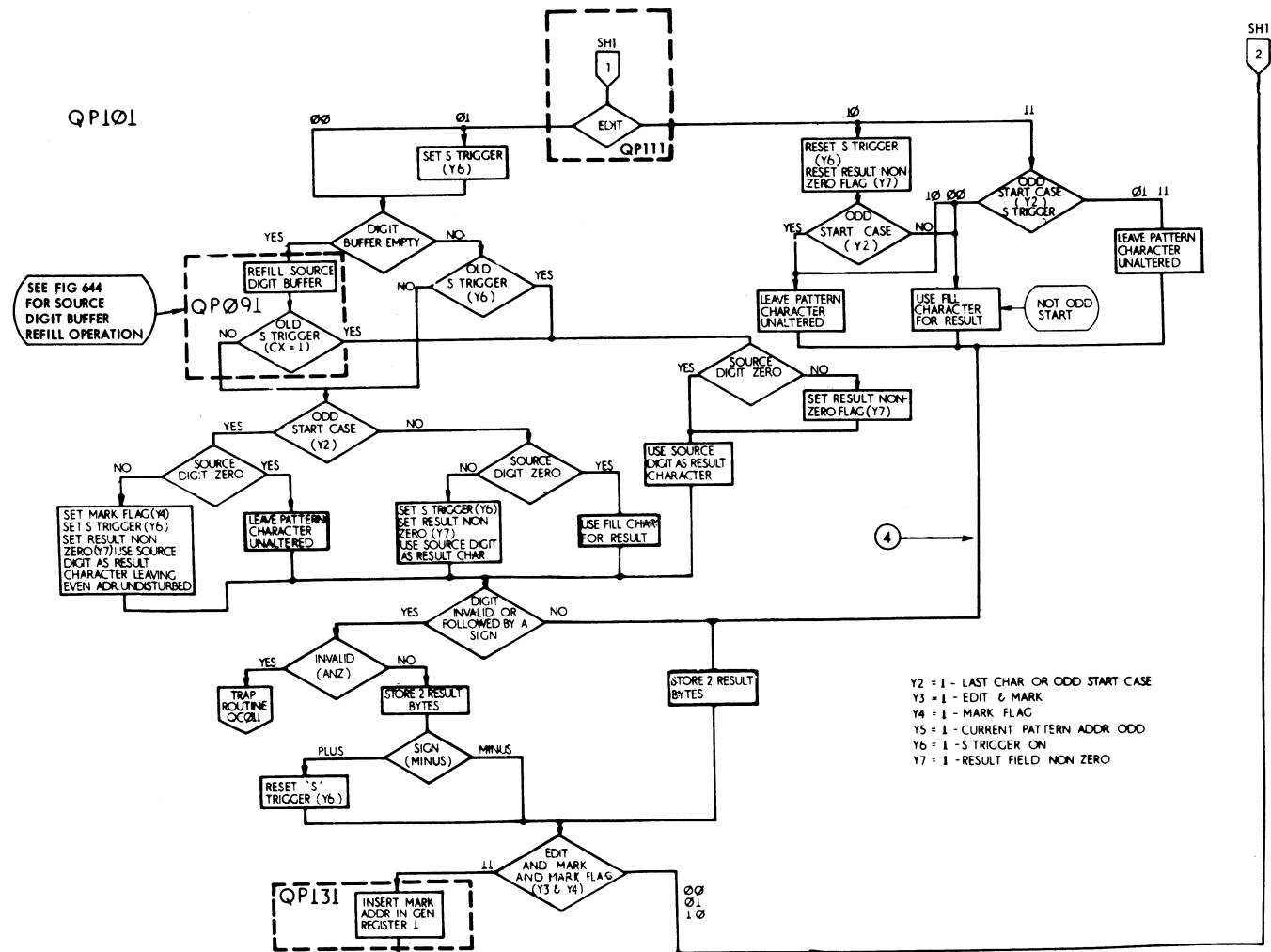


FIGURE 643, EDIT, EDIT AND MARK (QP091, QP101, QP121, QP131, QP141) (SHEET 2 OF 2)

**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.

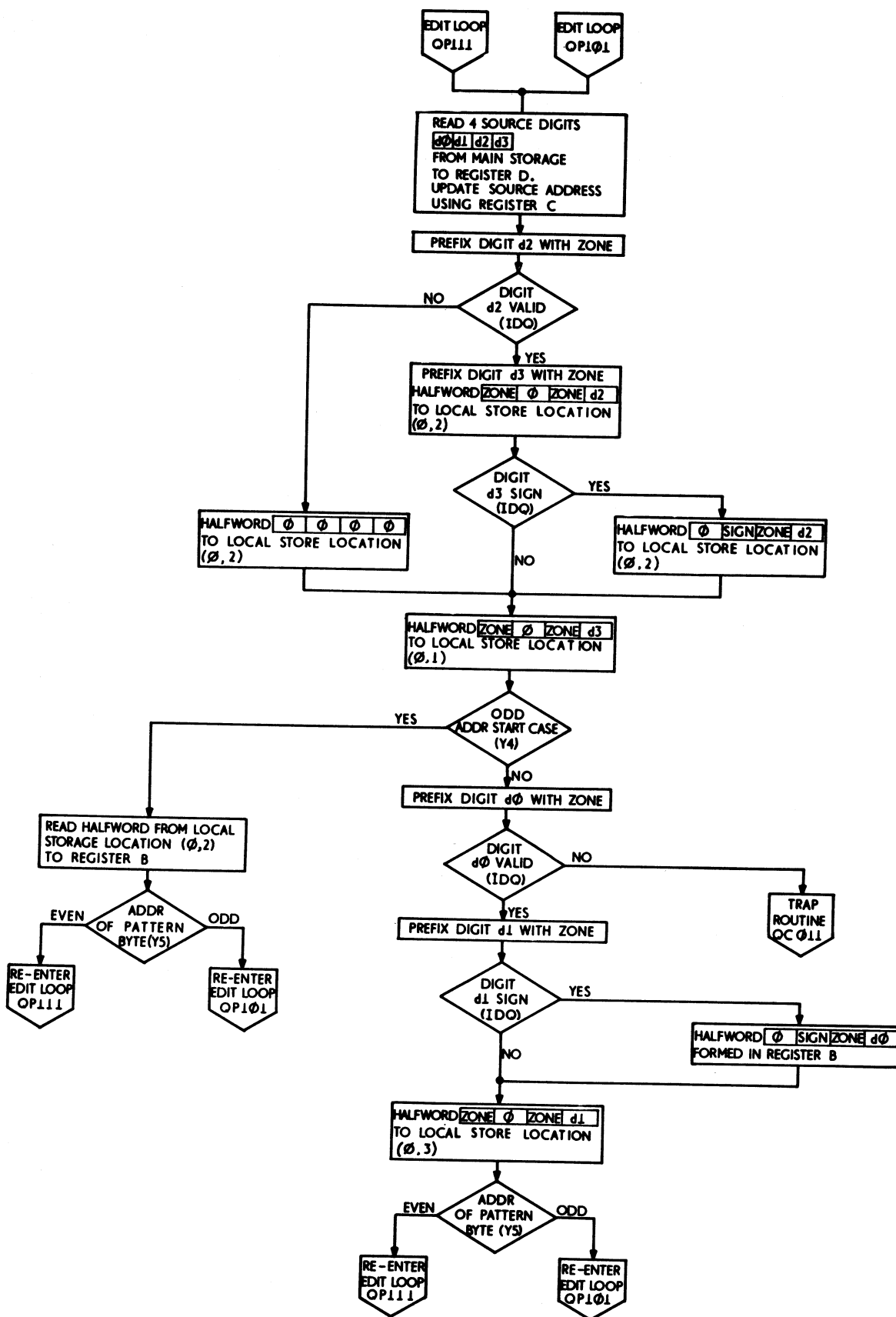
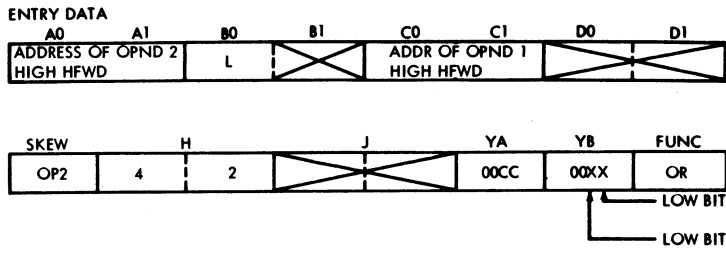


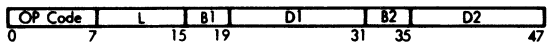
FIGURE 644 SS EDIT, REFILL (QP 091)



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

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

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.

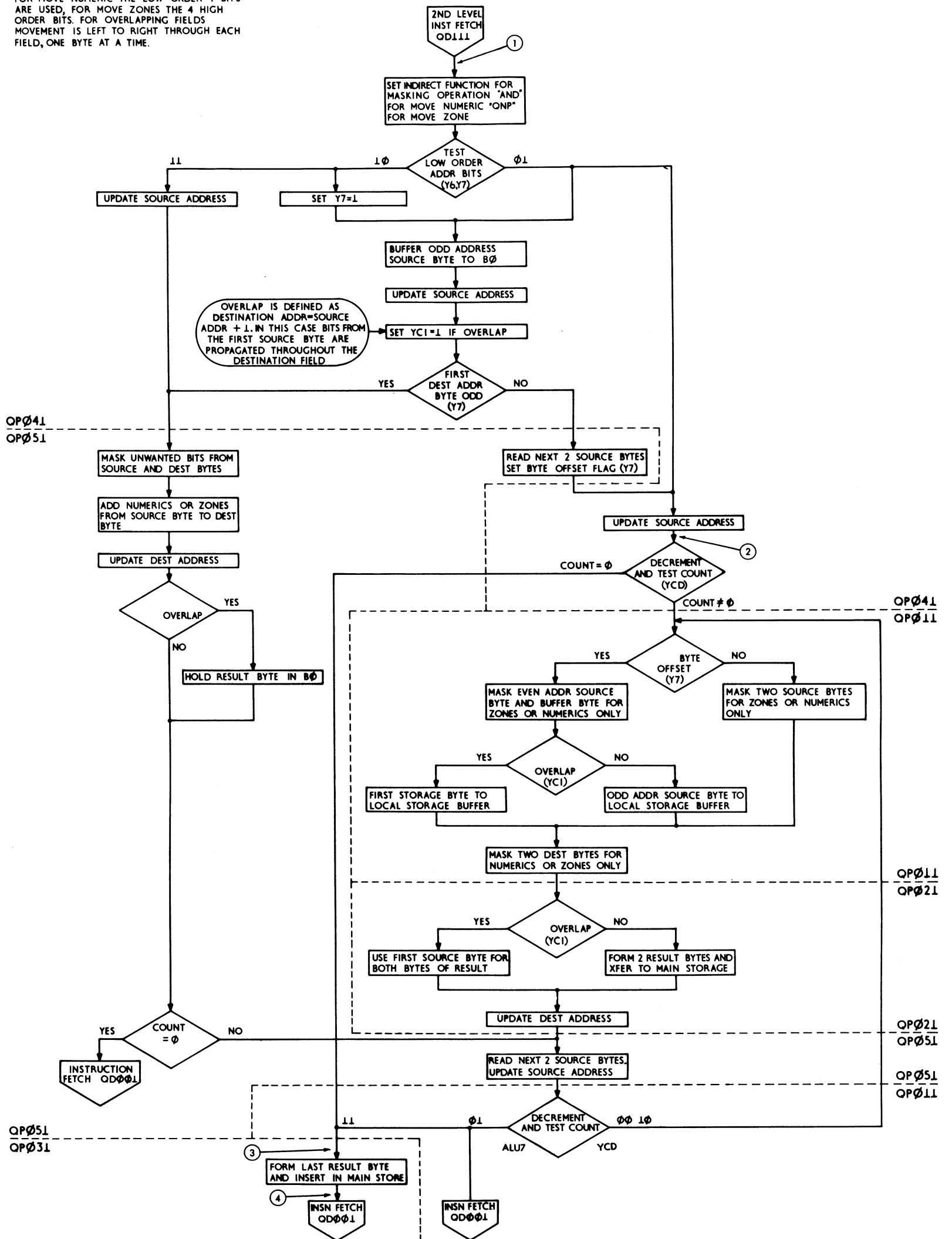
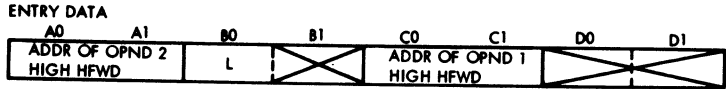
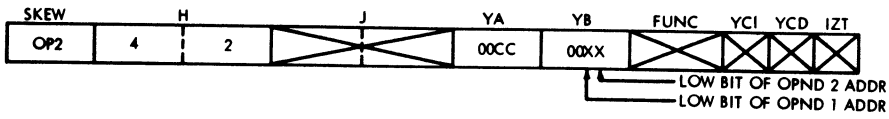


FIGURE 647 SS LOGICAL OPERATIONS - MOVE NUMERIC, MOVE ZONE

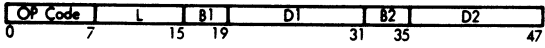


MOVE CHARACTER DMC SECTION 141, ROUTINE 18, STOP ON MS 676C

①	ROS	A REG	B REG	C REG	D REG
②	C04	6CF8	0300	0008	F8F9
③	C80	6CF8	0303	00FA	F8F9
④	C5A	0008	F8F9	6C02	F8F9
	C85	6CFC	6C00	6CFE	FCFD



55



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.

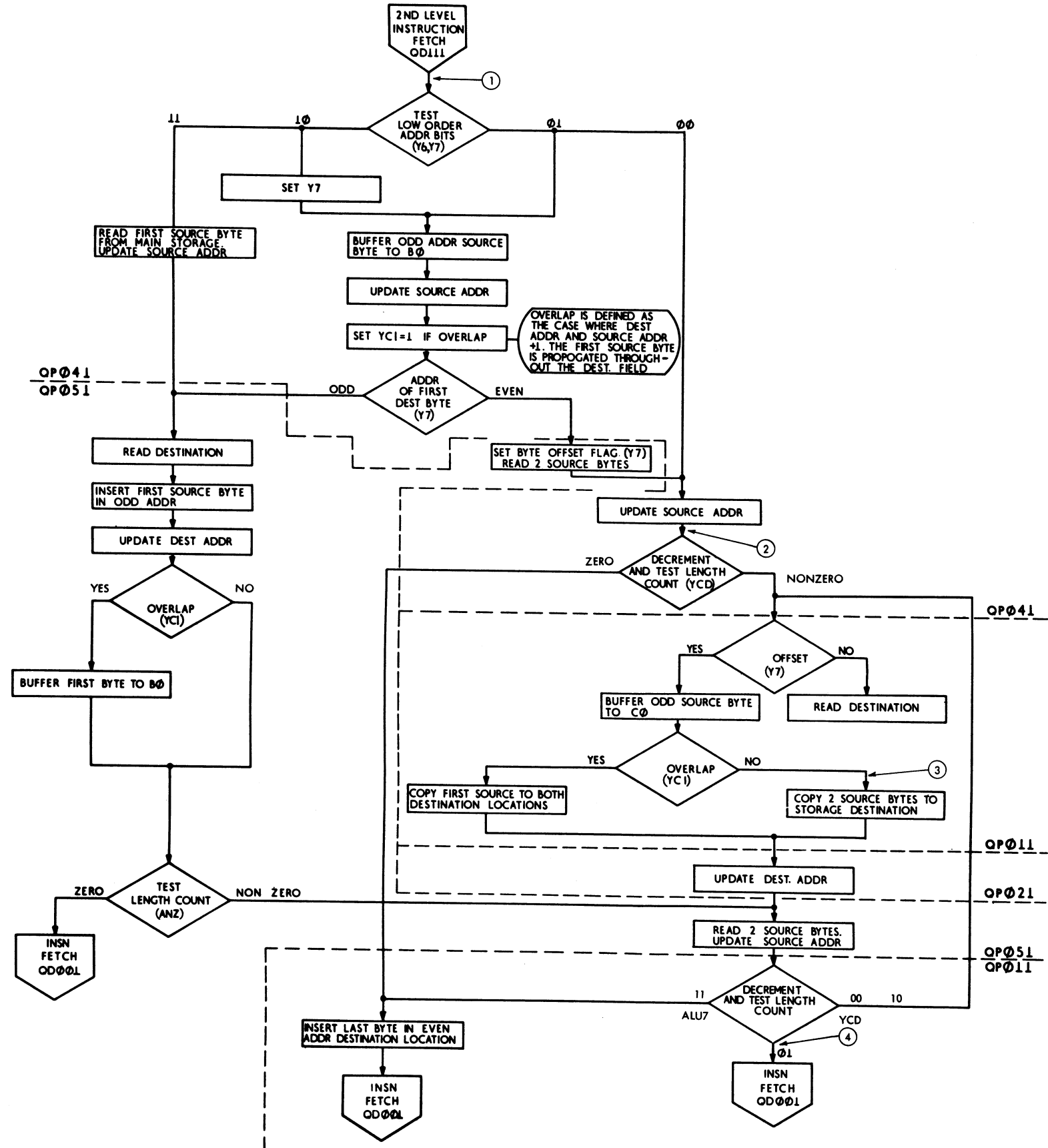


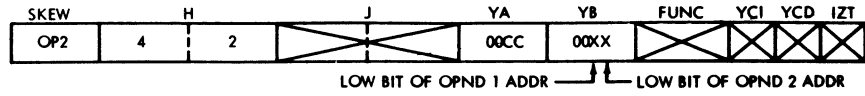
FIGURE 648, 55 LOGICAL OPERATIONS, MOVE COMPLETE



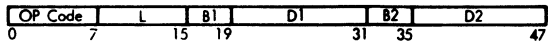


COMPARE LOGICAL CLC DMC SECTION 141, ROUTINE 08, STOP ON MS 631C

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



SS



DEST = OPERAND 1  
SOURCE = OPERAND 2

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.

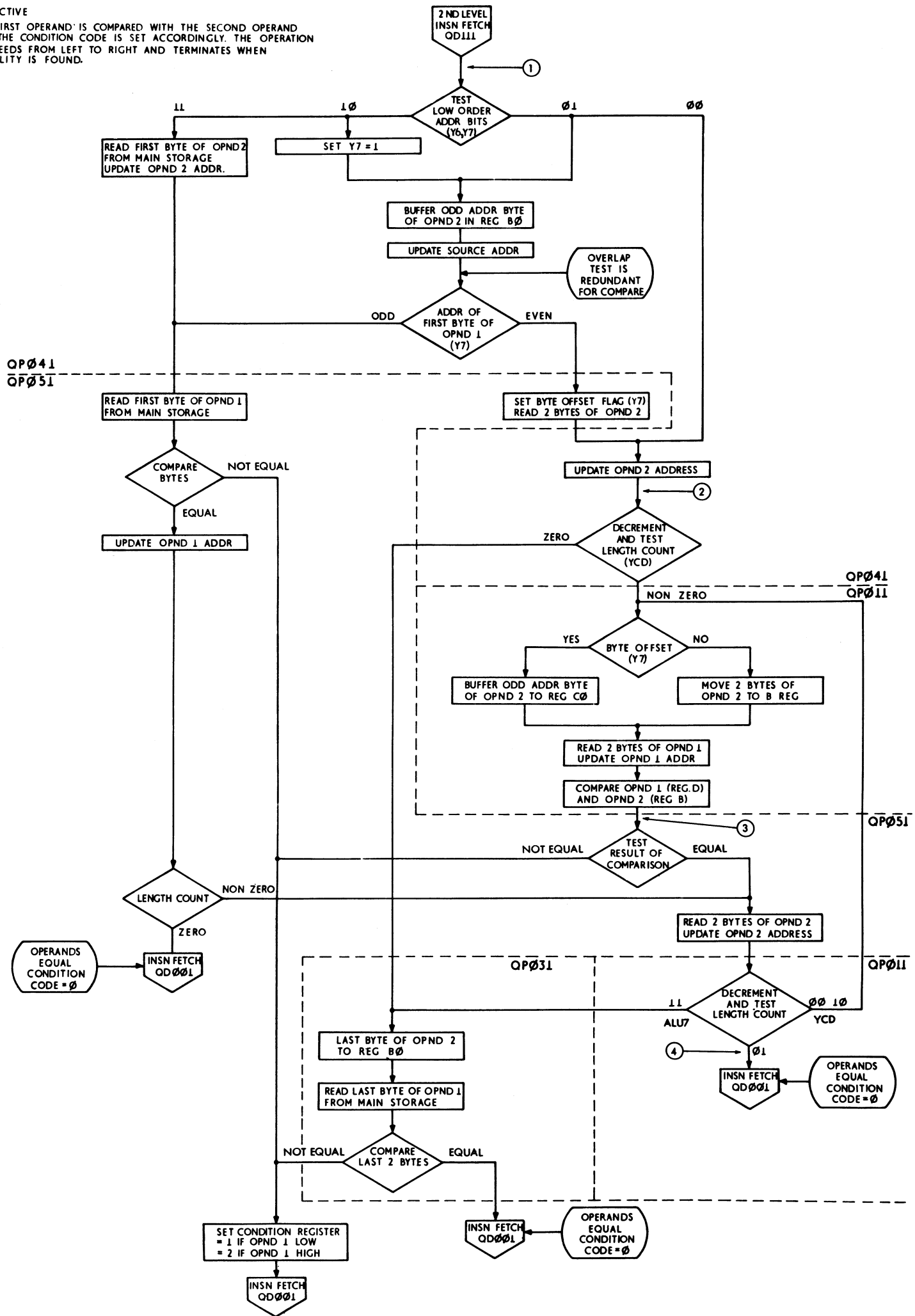
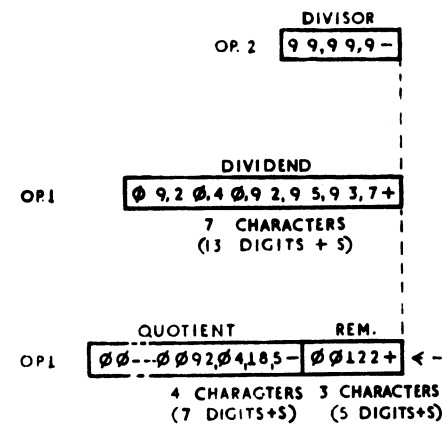
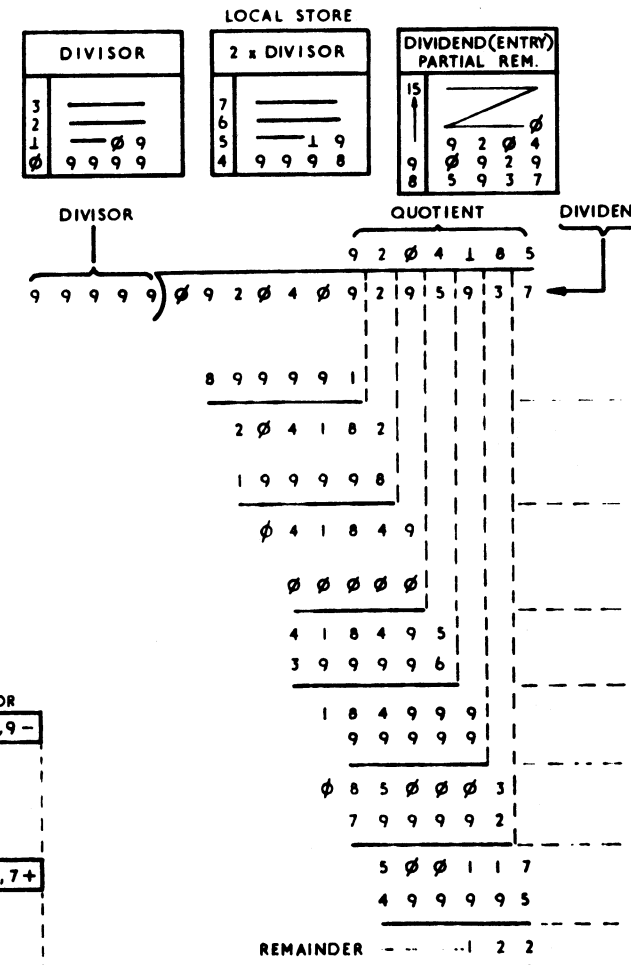


FIGURE 649.SS LOGICAL OPERATIONS, COMPARE

FIGURE 651. DECIMAL DIVIDE EXAMPLE

CALCULATE  $0920.409.295.937(+)$   $\div$   $9999(-)$

PRE-LOOP: LOAD BOTH OPERANDS FROM M. STORE INTO L. STORE.  
LOAD 2x DIVISOR IN L. STORE  
COMPUTE RESULTANT SIGNS AND STORE IN Y6, Y7  
Q SIGN =  $\frac{+}{-} = -$  (Y6) REM. SIGN = DIV SIGN  $\div$  (+) (Y7)



PARTIAL REM. IN L. STORE (BYTES ADDR. UNDERLINED)	SKEW &/OR OFFSET	SEQUENCE	MULTIPLES USED TO FORM QUOTIENT DIGIT
9 2 0 4 0 9 2 9 5 9 3 7	SO	TRIAL SUBTRACT	
9 2 0 4 0 9 2 9 5 9 3 7	SO	GENERATE Q DIGIT 9	-2 -2 -2 -2 +1
0 2 0 4 1 8 2 9 5 9 3 7	SO	GENERATE Q DIGIT 2 L. STOR BYTE 9,2	-2 -2 +1 +1
0 0 0 4 1 8 4 9 5 9 3 7	SO	GENERATE Q DIGIT 0	-2 +1 +1
0 0 0 4 1 8 4 9 5 9 3 7	SO	GENERATE Q DIGIT 4 L. STOR BYTE 04	-2 -2 -2 +1 +1
1 8 4 9 9 9 3 7	SO	GENERATE Q DIGIT 1	-2 +1
0 8 5 0 0 0 3 7	SO	GENERATE Q DIGIT 8 L. STOR BYTE 18	-2 -2 -2 -2 +1 +1
0 0 5 0 0 1 1 7	SO	GENERATE Q DIGIT 5 L. STOR Q BYTE 50	-2 -2 -2 +1
0 1 2 2	NO OP.	QUOTIENT COMPLETED	EXIT TO QR161

EXIT: ATTACH SIGNS TO Q AND REM,  
THEN MAIN STORE Q AND REM.

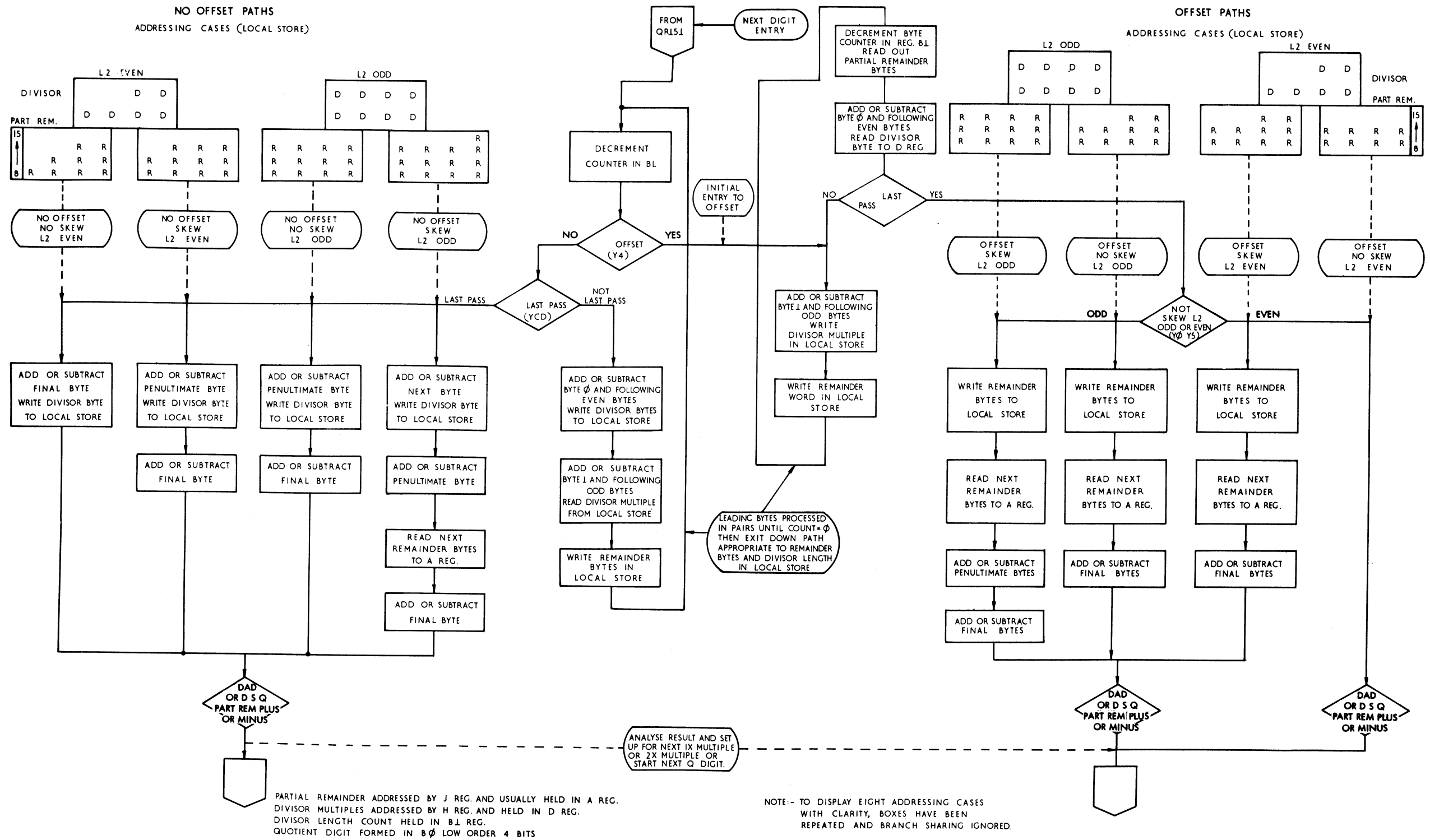
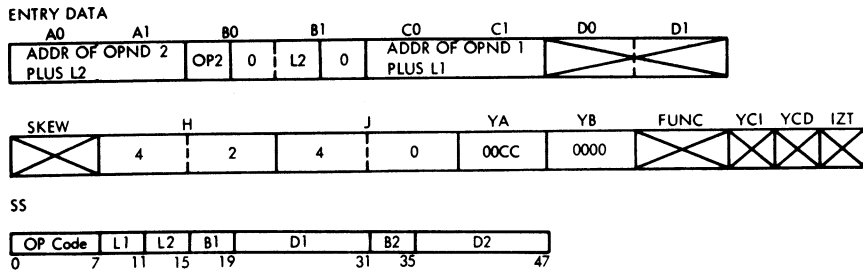


FIGURE 652 DECIMAL DIVIDE ADD/ SUBTRACT PATHS (QR141)

PARTIAL REMAINDER ADDRESSED BY J REG. AND USUALLY HELD IN A REG.  
 DIVISOR MULTIPLES ADDRESSED BY H REG. AND HELD IN D REG.  
 DIVISOR LENGTH COUNT HELD IN B1 REG.  
 QUOTIENT DIGIT FORMED IN B0 LOW ORDER 4 BITS

NOTE:- TO DISPLAY EIGHT ADDRESSING CASES WITH CLARITY, BOXES HAVE BEEN REPEATED AND BRANCH SHARING IGNORED.



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

①	ROS	A REG	B REG	C REG	D REG
②	E02	6790	C000	6771	F790
③	E06	678E	000C	670C	5C01
④	E38	6787	0020	F00C	0000
⑤	E47	6771	0300	10C2	0060
⑥	E62	6771	5C00	0700	005C
⑦	EB2	6771	0700	0020	5C00
⑧	E72	6771	0200	07FB	5CE0
	4F7	6771	025C	025C	025C

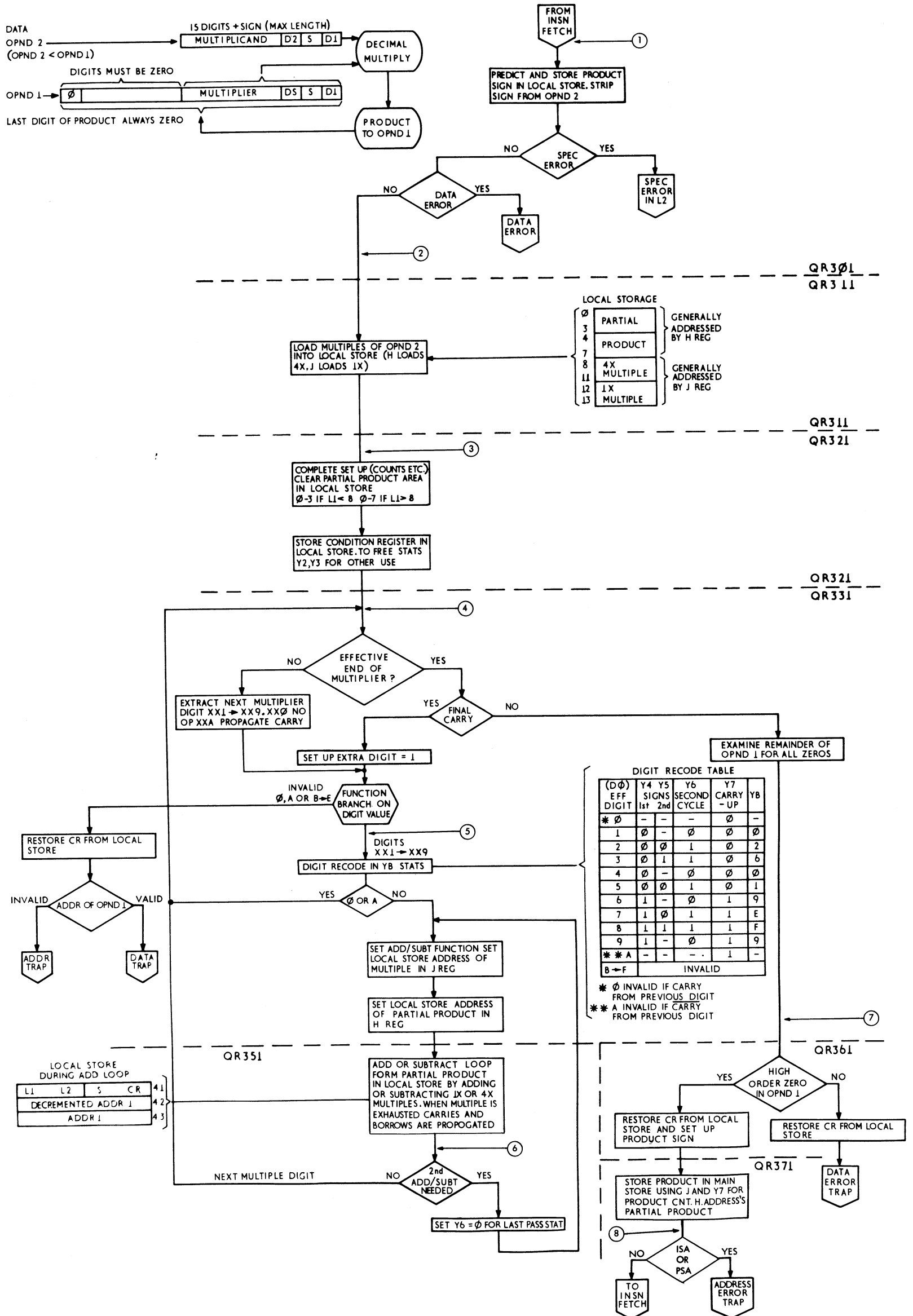
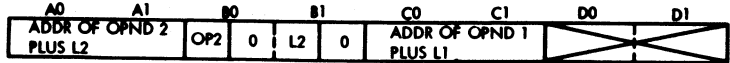
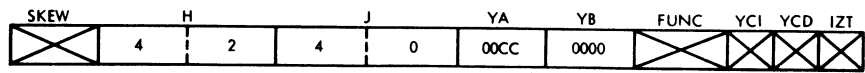


FIGURE 653 DECIMAL MULTIPLY

ENTRY DATA

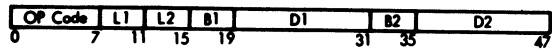


PACK DMC SECTION 181, ROUTINE 02, STOP ON MS 622C, THEN STOP ON MS 62D2. THIS WILL INSURE THAT THIS IS THE FIRST PASS THROUGH THE LOOP.



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

SS



OBJECTIVE

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

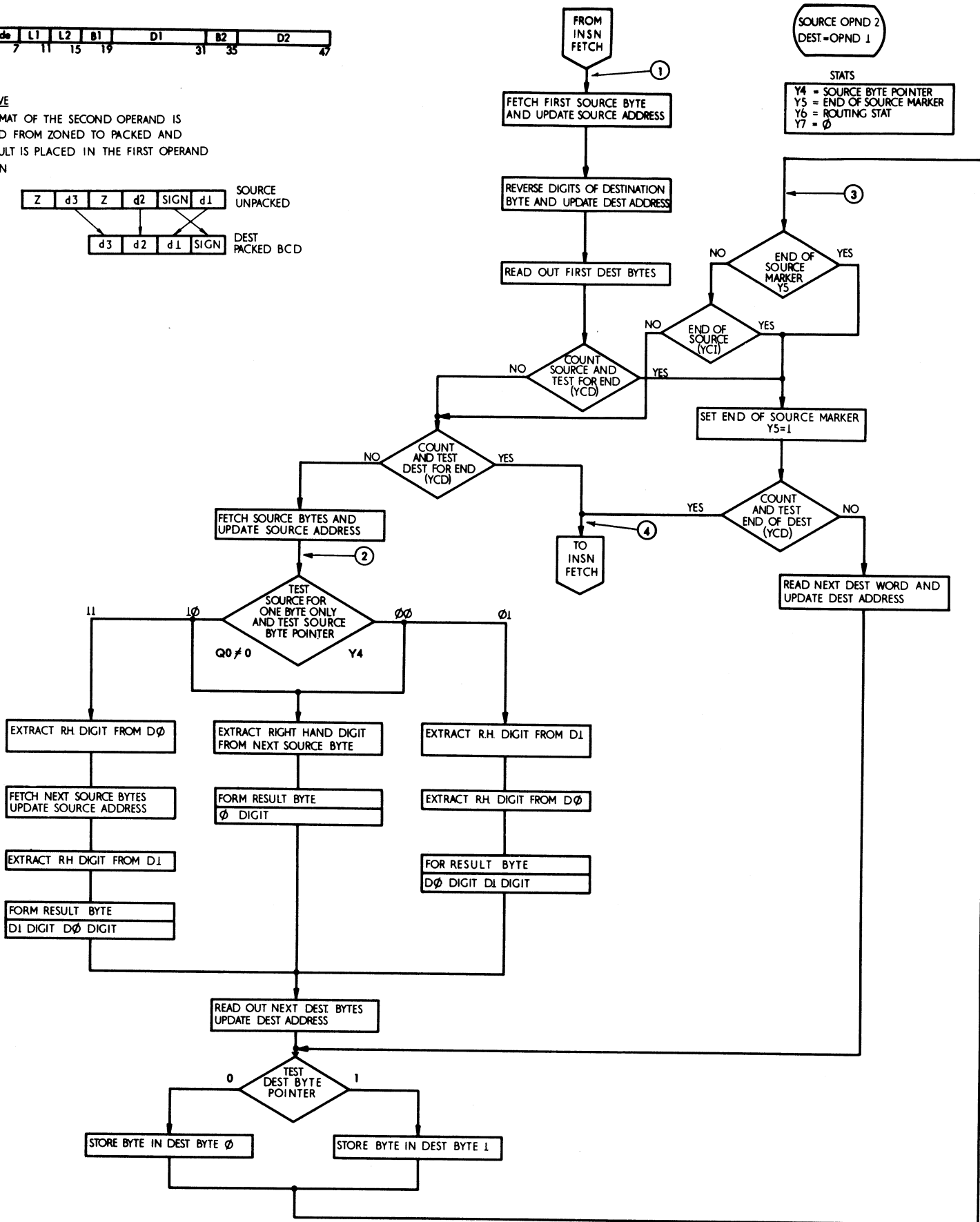
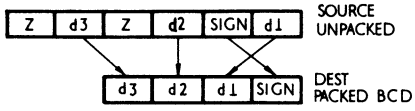
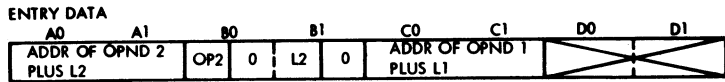
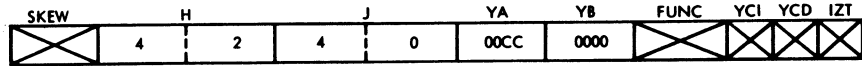


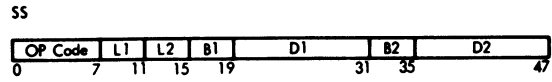
FIGURE 654 DECIMAL PACK (QR101, QR111)



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.



①	ROS	A REG	B REG	C REG	D REG
②	C07	65CE	3010	65D0	2000
③	440	65D0	1000	65CF	EF00
④	474	65CD	0000	65CC	0000
	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 0101 FOR ASCII  
 DATA FORMAT EXAMPLE

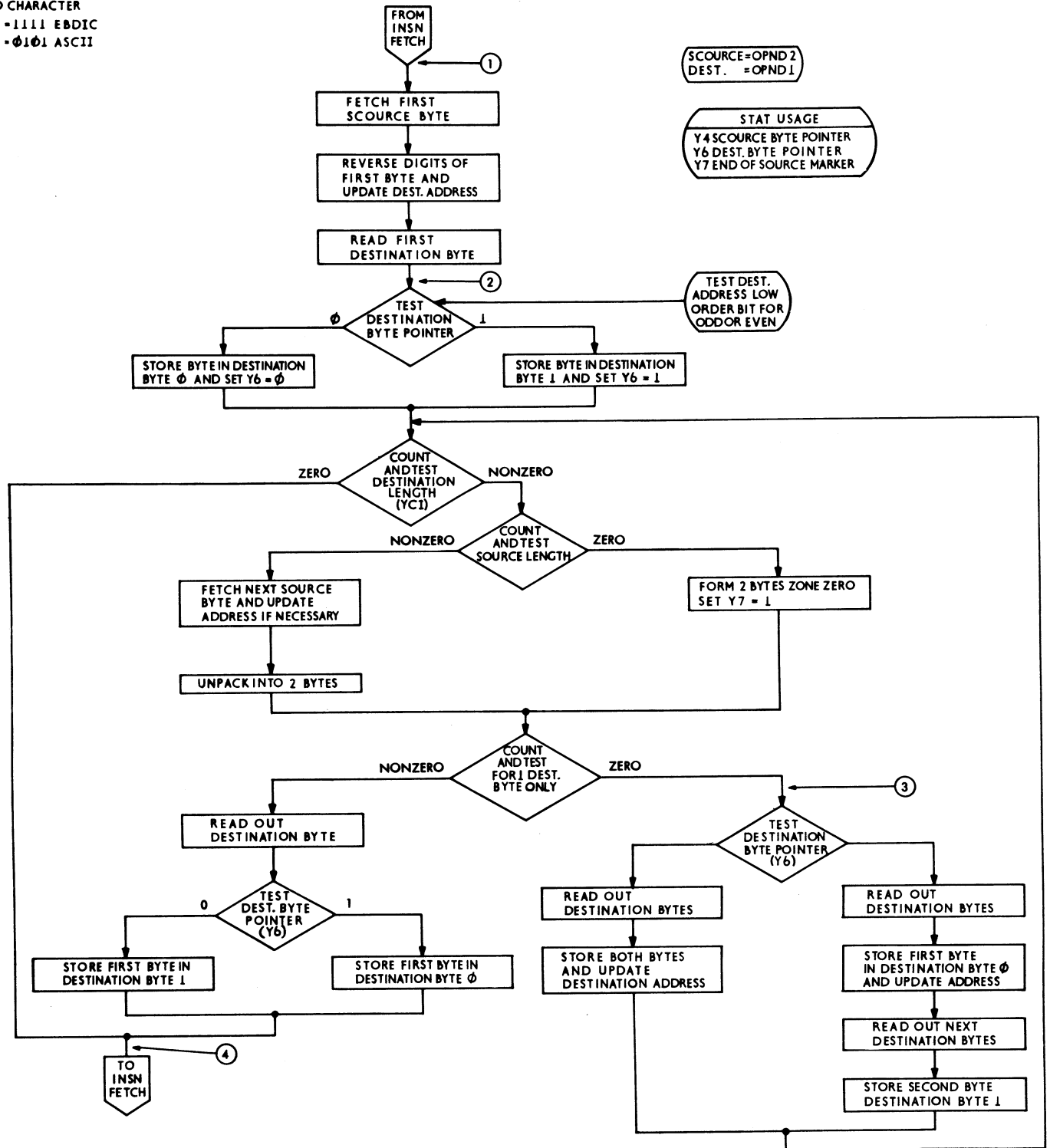
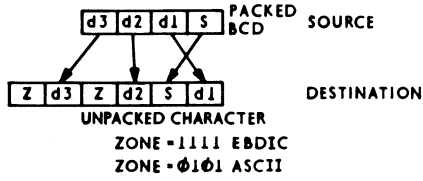
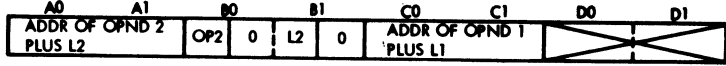
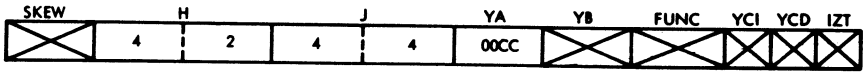


FIGURE 655 DECIMAL UNPACK (QR101, QR121)

ENTRY DATA



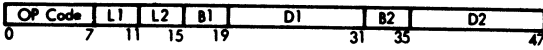
MOVE WITH OFFSET MVO DMC SECTION 17C, ROUTINE 10, STOP ON MS 658E



- ① ROS
- ② C03
- ③ 855
- ④ 868
- 841

A REG	B REG	C REG	D REG
68C2	1010	699B	F8C1
699B	2010	699A	0000
68C1	0000	68C0	0011
6594	00FF	6901	1220

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

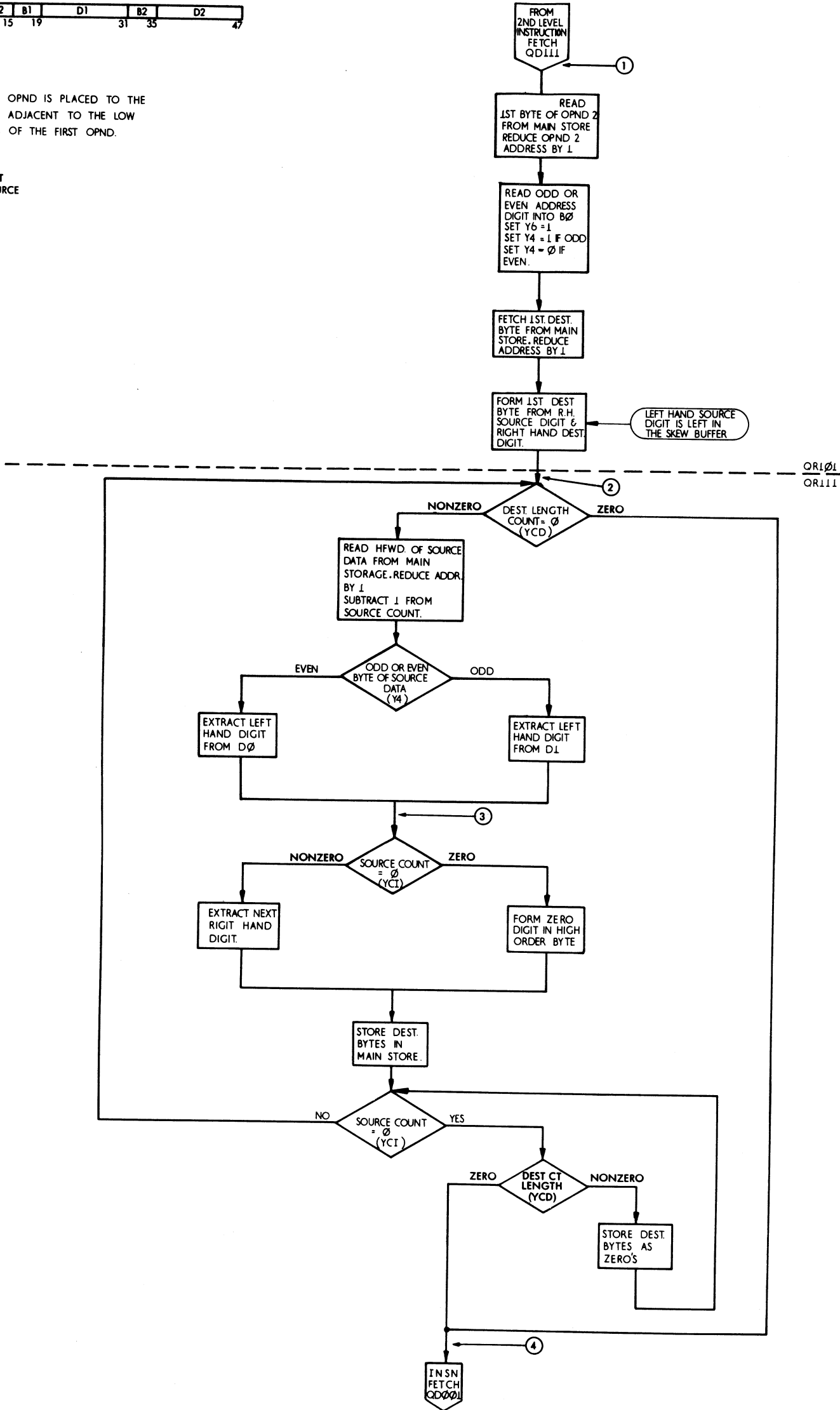
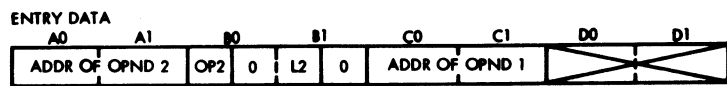
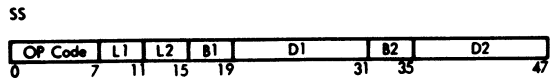
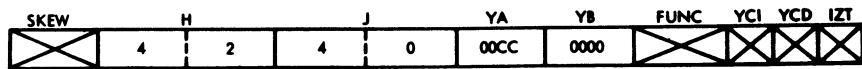


FIGURE 656 DECIMAL MOVE WITH OFFSET



NO EXAMPLE SHOWN.



**OBJECTIVE**

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

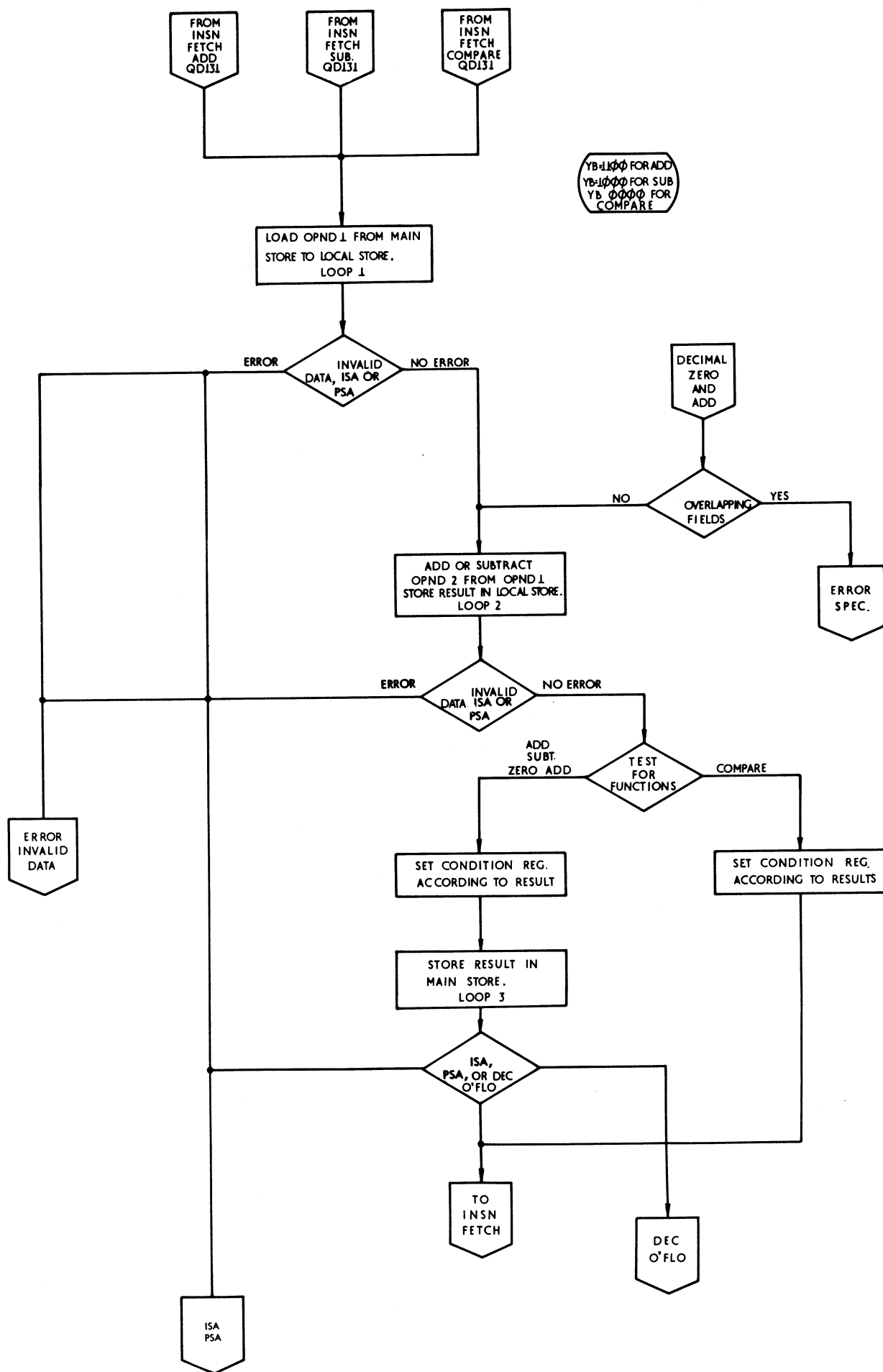
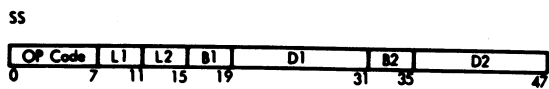
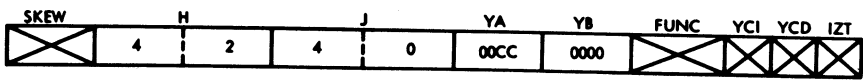
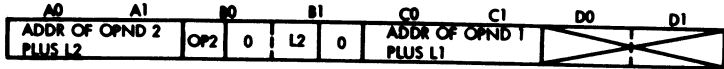


FIGURE 657 GENERAL FLOW CHART FOR DECIMAL ADD, SUBTRACT, COMPARE AND ZERO ADD (QR001, QR021, QR011, QR031)



ENTRY DATA



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

- ① ROS
- ② C15
- ③ D6A
- ④ D4A
- ⑤ D4B

A REG	B REG	C REG	D REG
6C67	A060	69AE	FC61
69AE	051C	69AC	1CFF
69AB	051C	6910	1010
6C67	6005	0265	000C

FIGURE 660

- ⑥ D53
- ⑦ D23
- ⑧ D16
- ⑨ D38

6C67	6F10	0265	000C
6C67	6F10	024F	0000
6C61	6910	10EF	0000
6C61	0C10	10EF	0000

FIGURE 661

- ⑩ D7D
- ⑪ D46
- ⑫ D32
- ⑬ D5C

6C61	0C10	10EF	0000
69AE	0C0D	104F	1CFF
69AB	1010	10A8	1010
69AB	1000	4000	1010

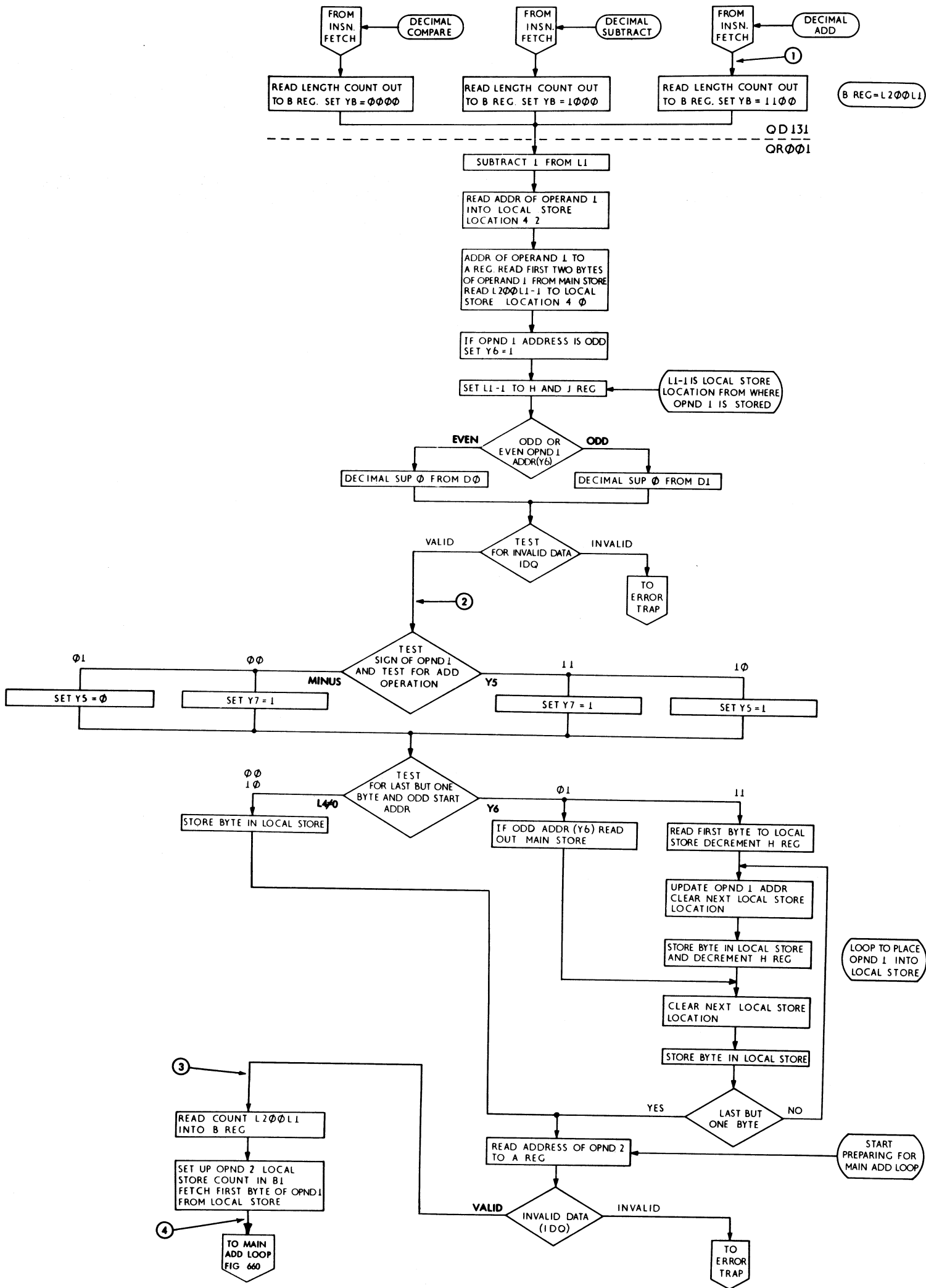
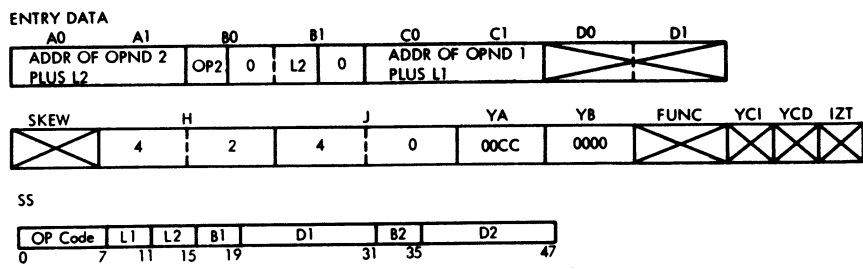


FIGURE 658 SS DECIMAL LOAD OPND 1 (ADD, SUBTRACT, COMPARE) (QD131, QR001)



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

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

OPND 1 = DEST  
OPND 2 = SOURCE

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

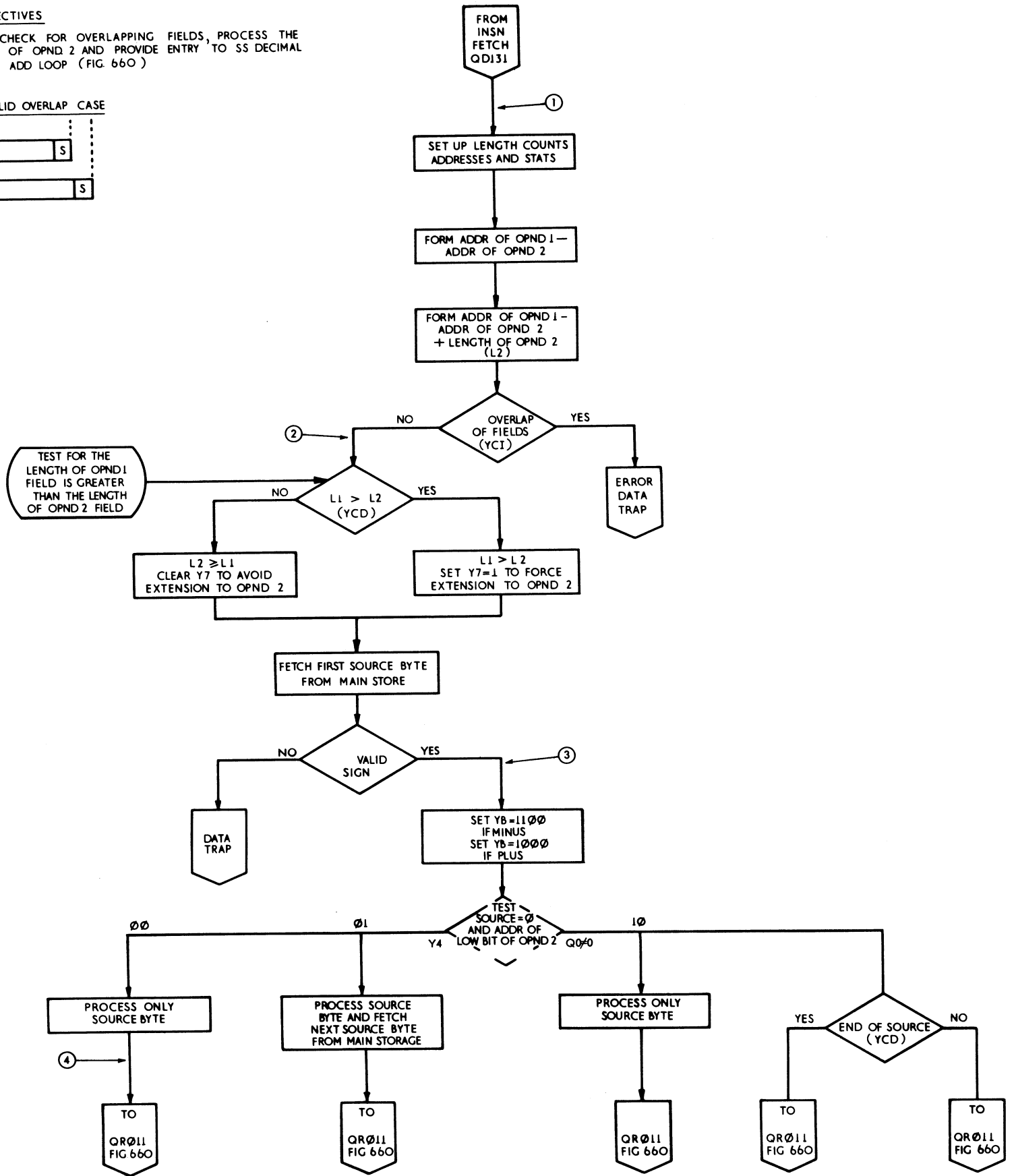
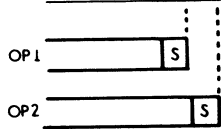


FIGURE 659 SS DECIMAL LOAD ZERO AND ADD ENTRY (QR041)

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

1100 FOR ADD  
 1000 FOR SUBT  
 0000 FOR COMP.

OPND 1 = DEST  
 OPND 2 = SOURCE

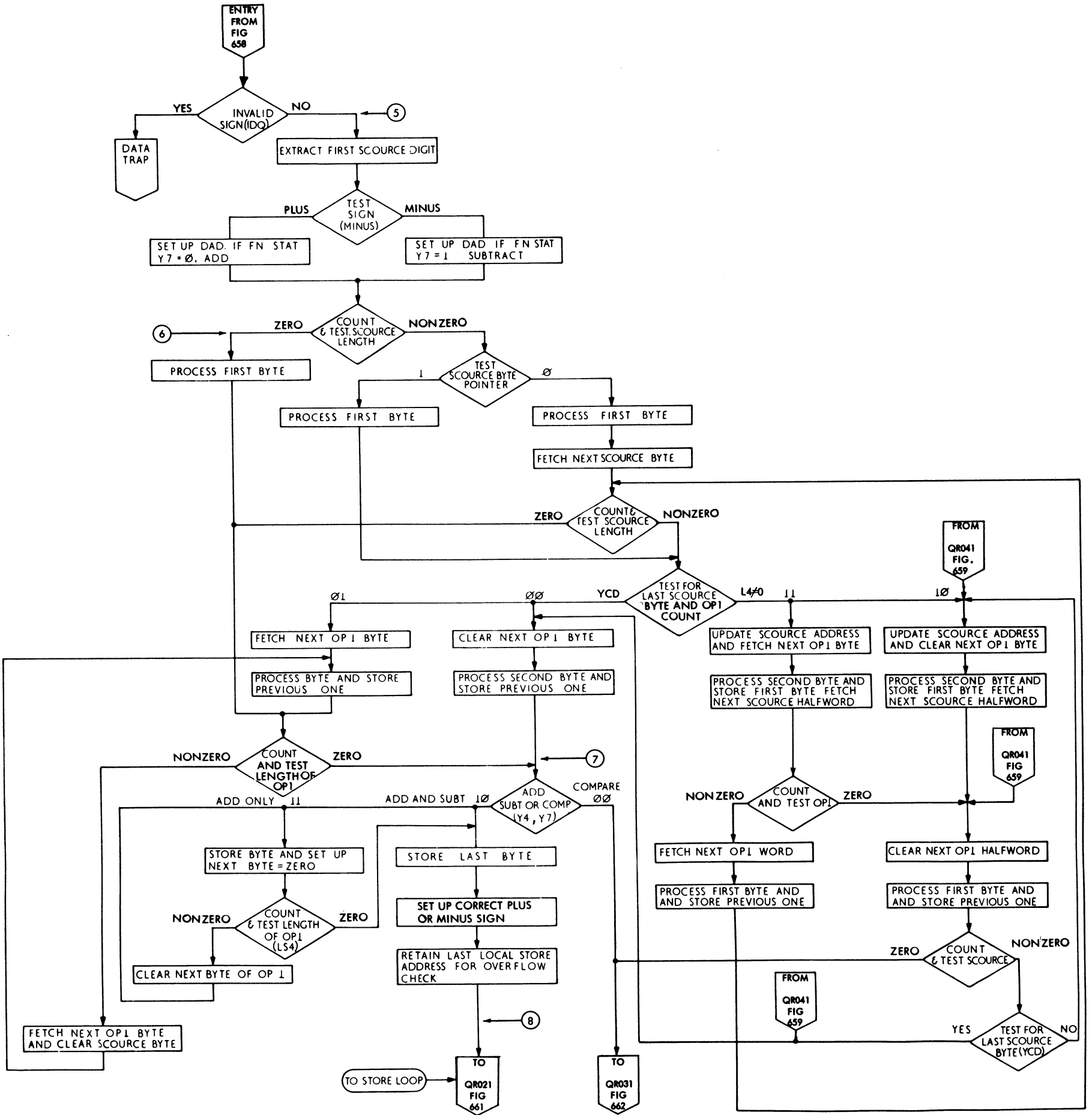


FIGURE 660.55 DECIMAL LOAD OPND 2 AND PROCESS (ADD, SUBT, COMP, ZERO AND ADD) (QR011)

**OBJECTIVE**

TO STORE RESULT OF THE DECIMAL ADD, SUBTRACT OR ZERO ADD IN THE MAIN STORE LOCATION OF OPND.1 AND SET CONDITION REGISTER ACCORDING TO RESULT.

(See Figure 658 for program)

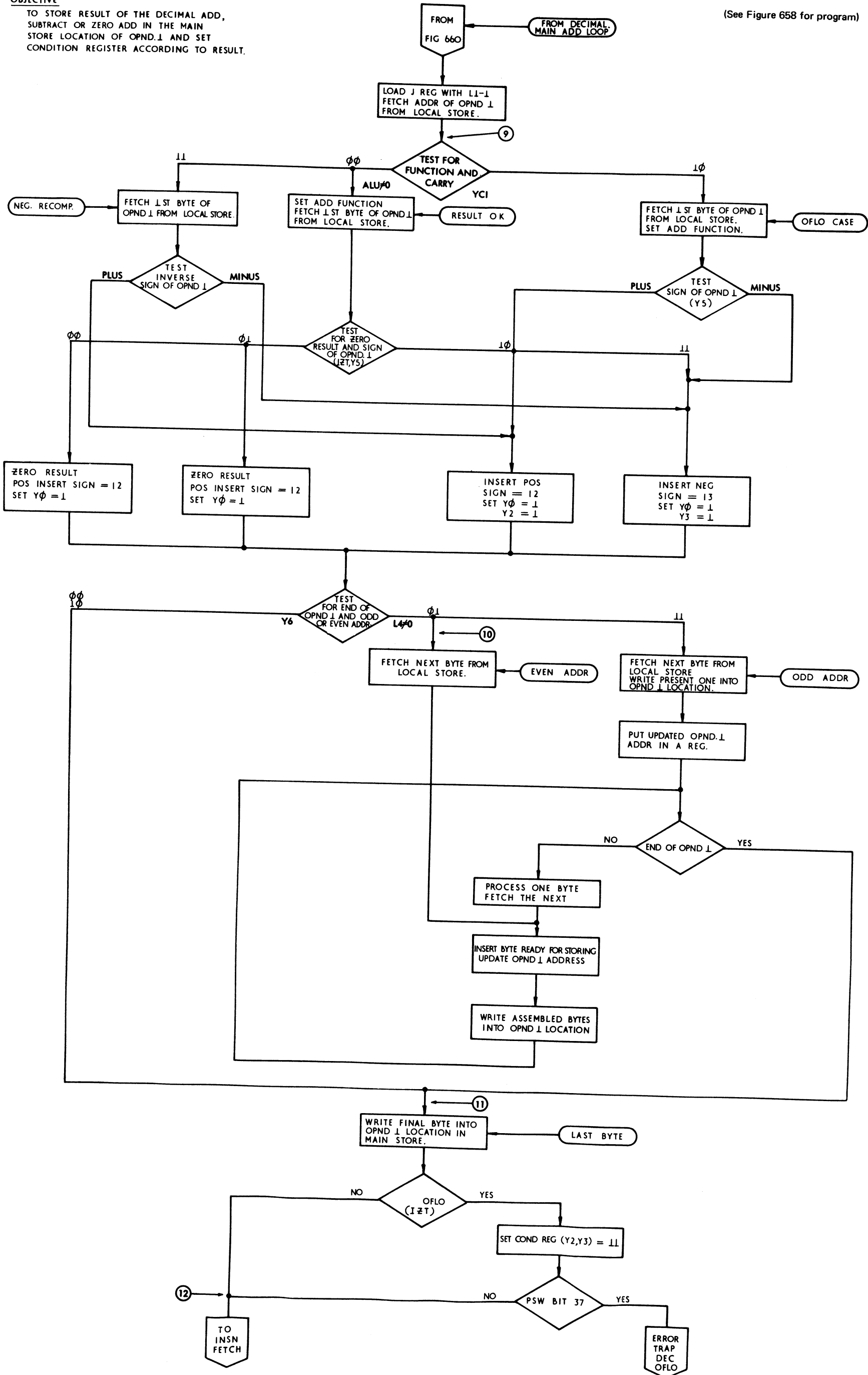


FIGURE 661.55 DECIMAL TERMINATE (ADD SUBTRACT AND ZERO AND ADD) (QR021)

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

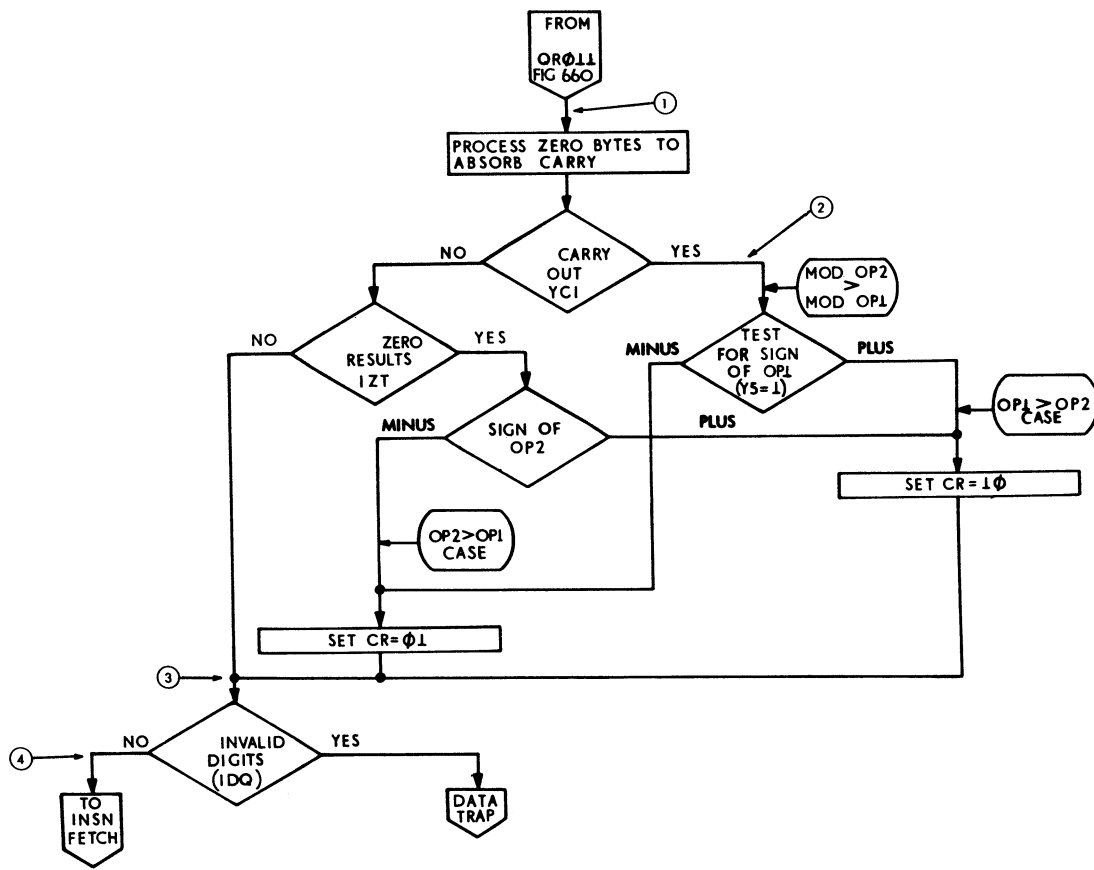


FIGURE 662 SS DECIMAL COMPARE (QR001)

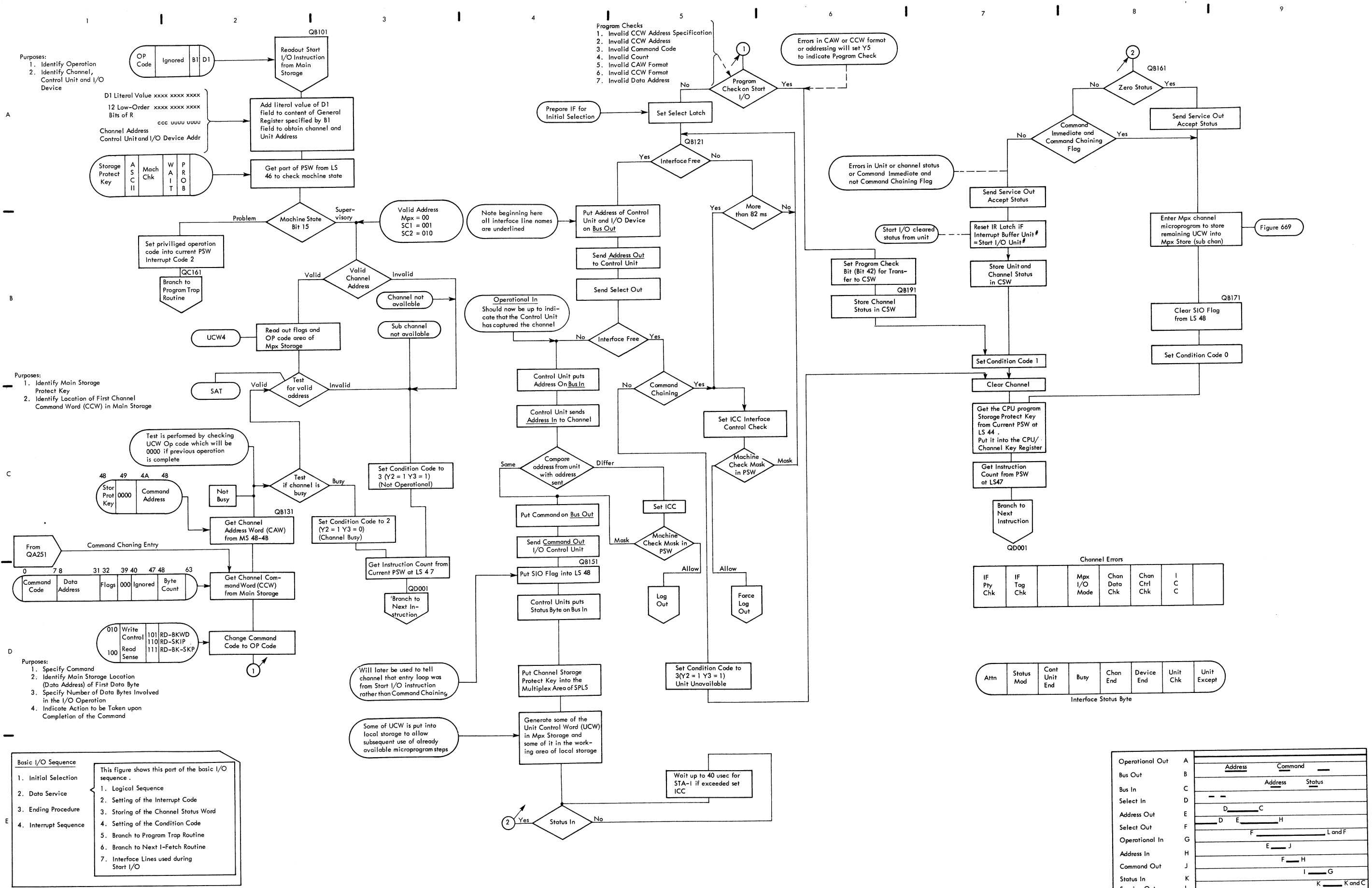


FIGURE 665. START I/O INSTRUCTION (MULTIPLEX CHANNEL) SEE FIG 667 FOR DETAIL

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

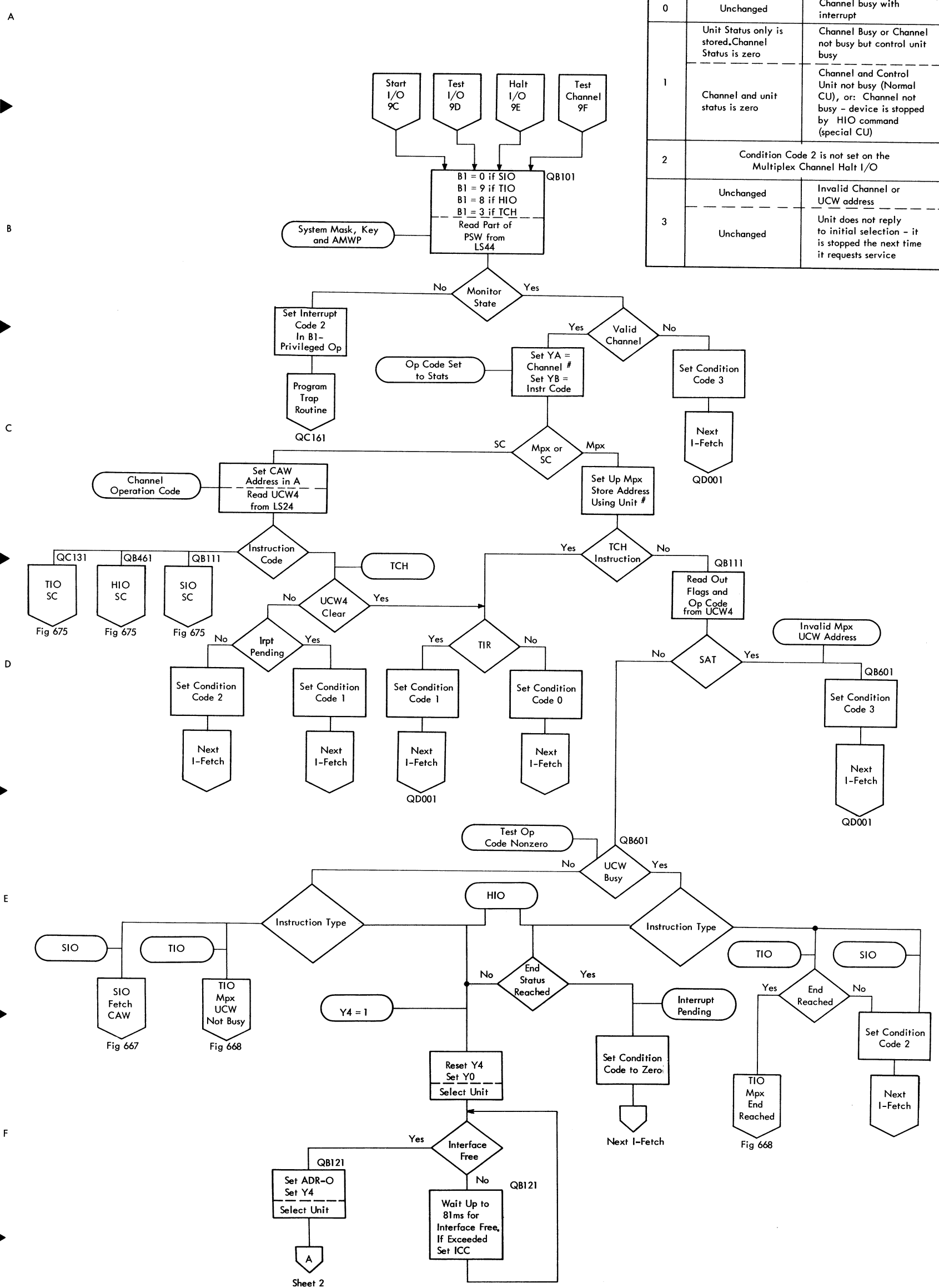
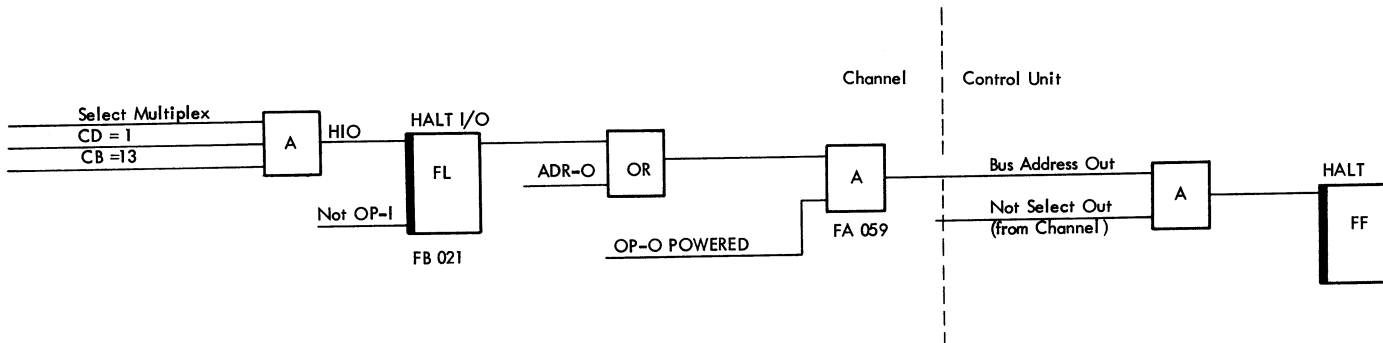
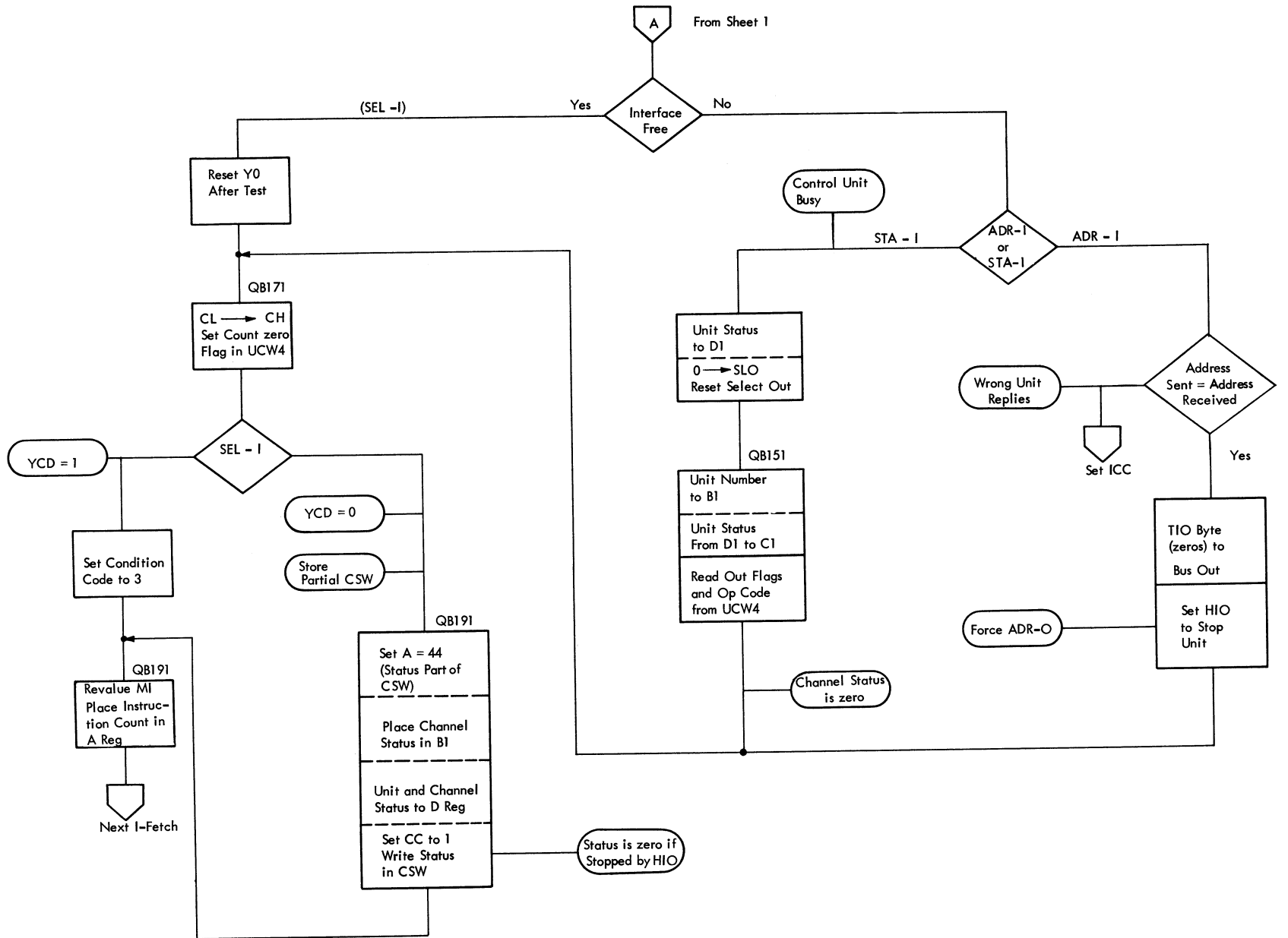


FIGURE 666. I/O CODES, COMMON DECODING; TEST CHANNEL AND MPX HALT I/O (SHEET 1 OF 2)



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

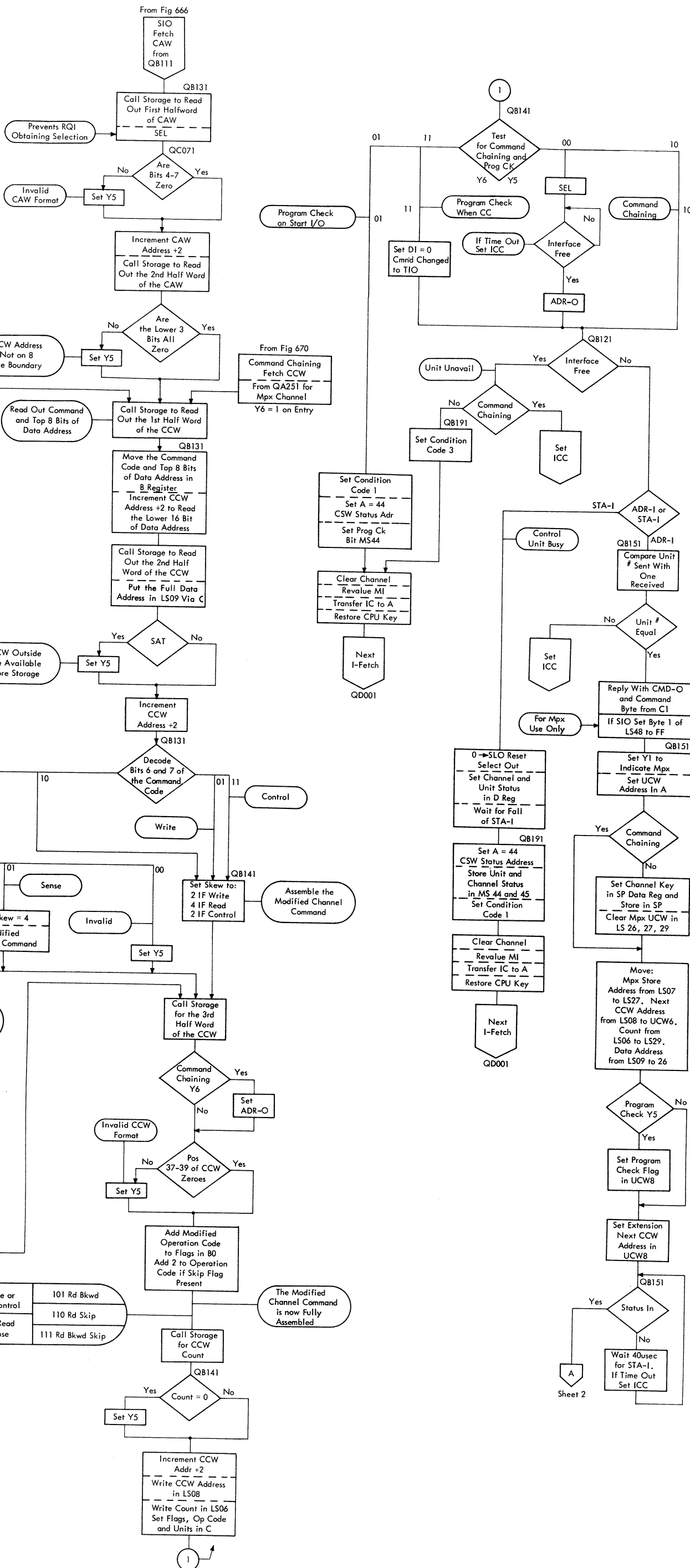


FIGURE 667. START I/O MICROPROGRAM MPX CHANNEL (SHEET 1 OF 2)

A

B

C

D

E

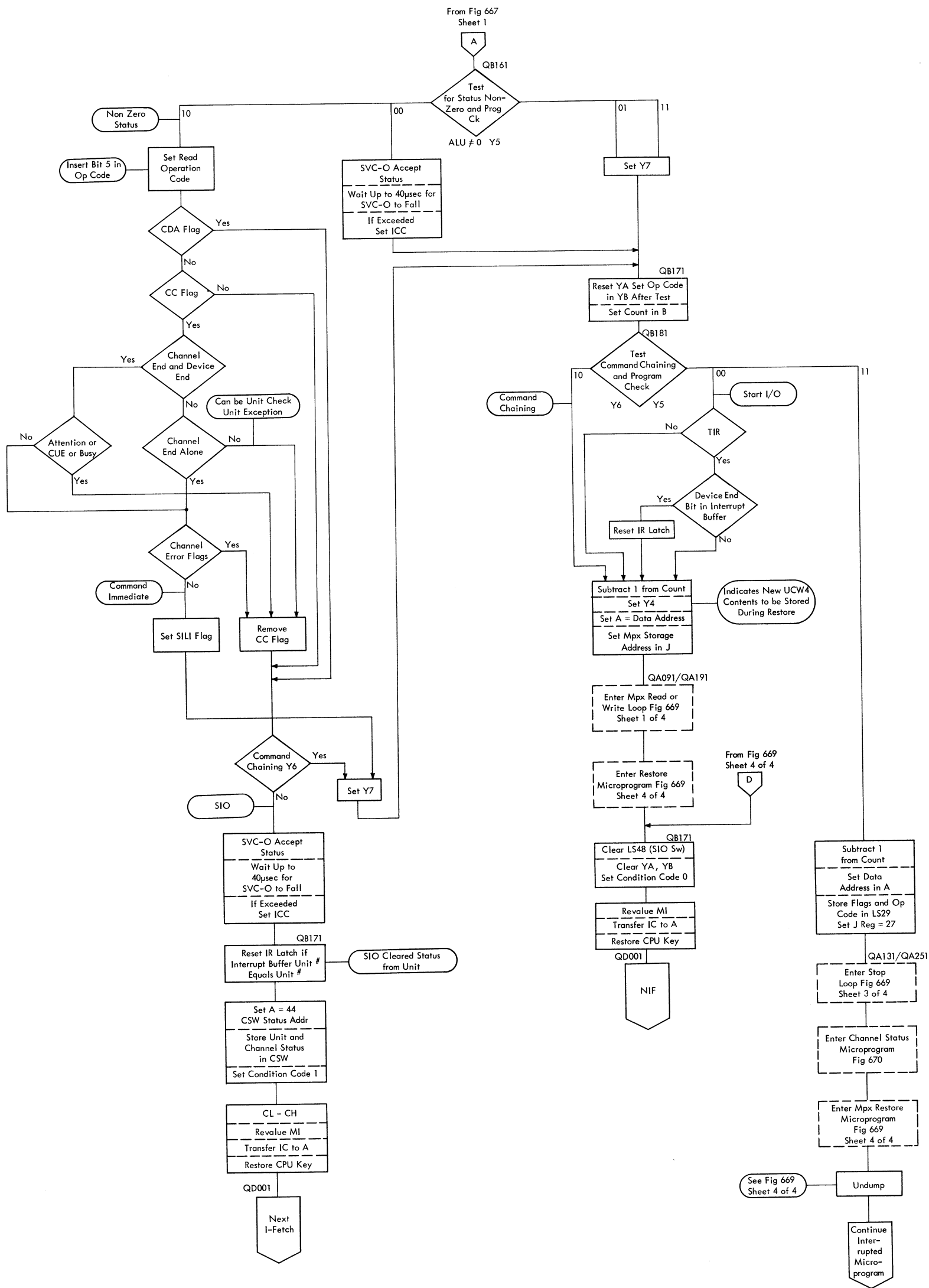
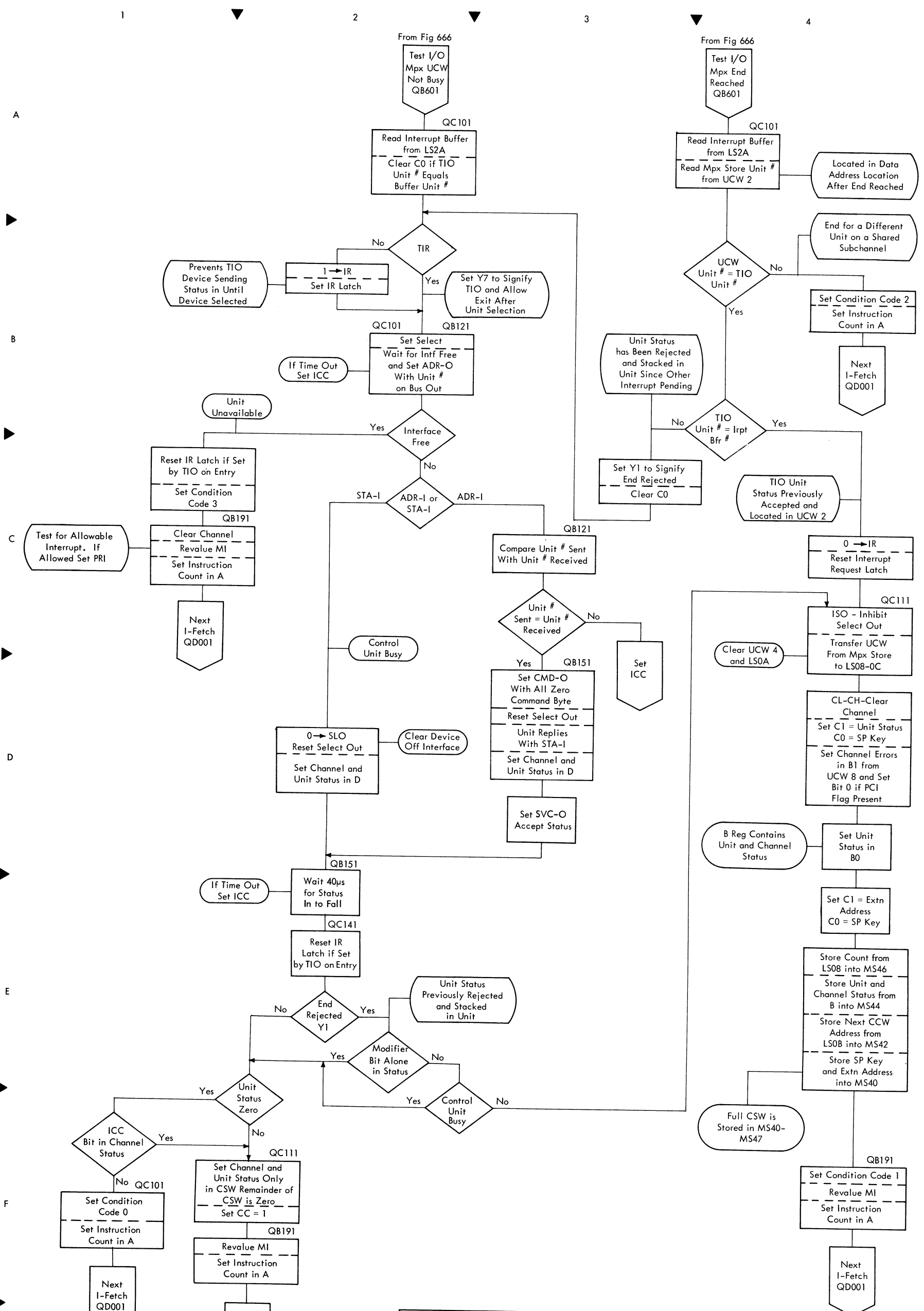


FIGURE 667. START I/O MICROPROGRAM MPX CHANNEL (SHEET 2 OF 2)



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

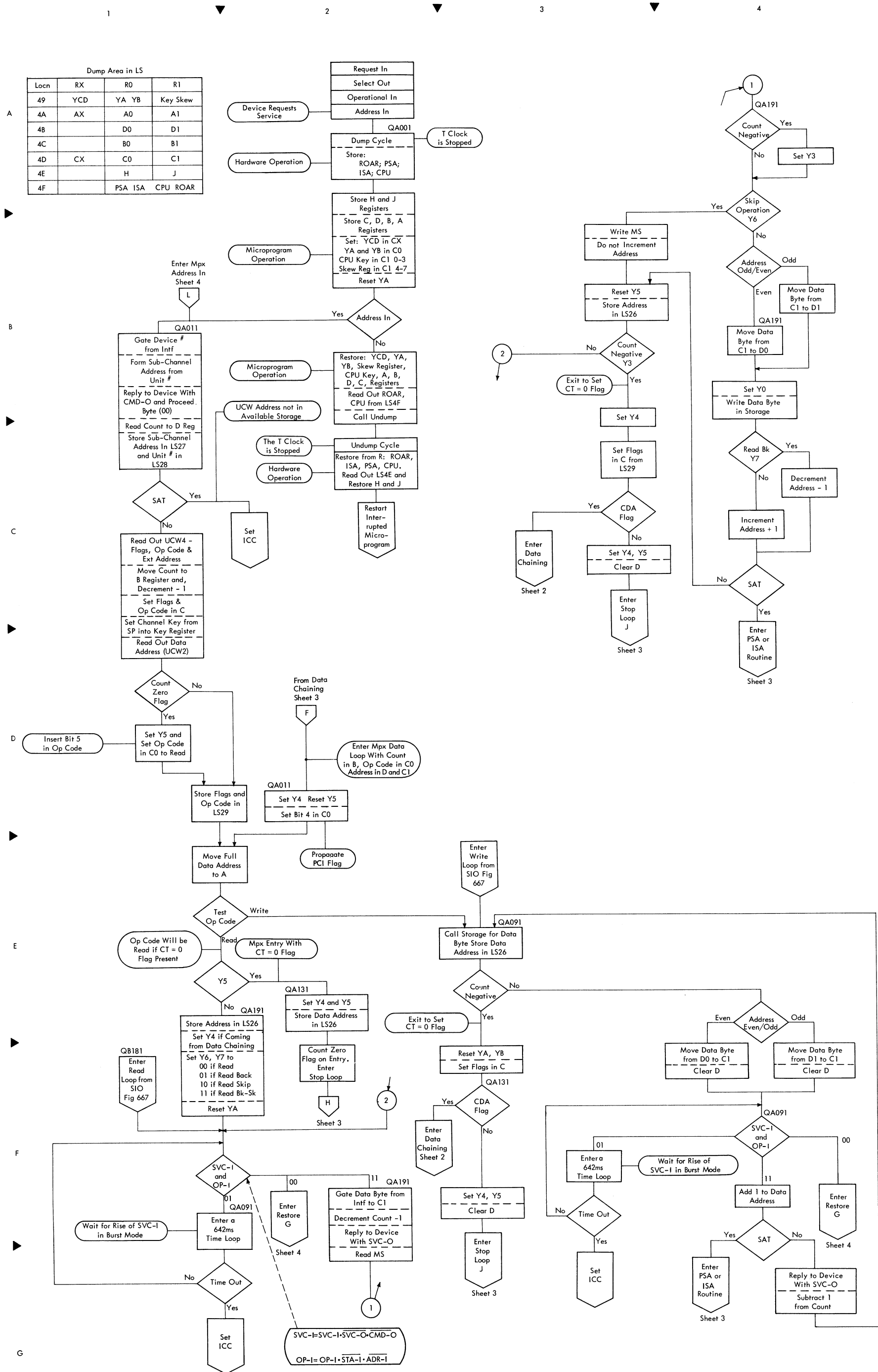


FIGURE 669. MULTIPLEX CHANNEL MICROPROGRAM (SHEET 1 OF 4)

A

B

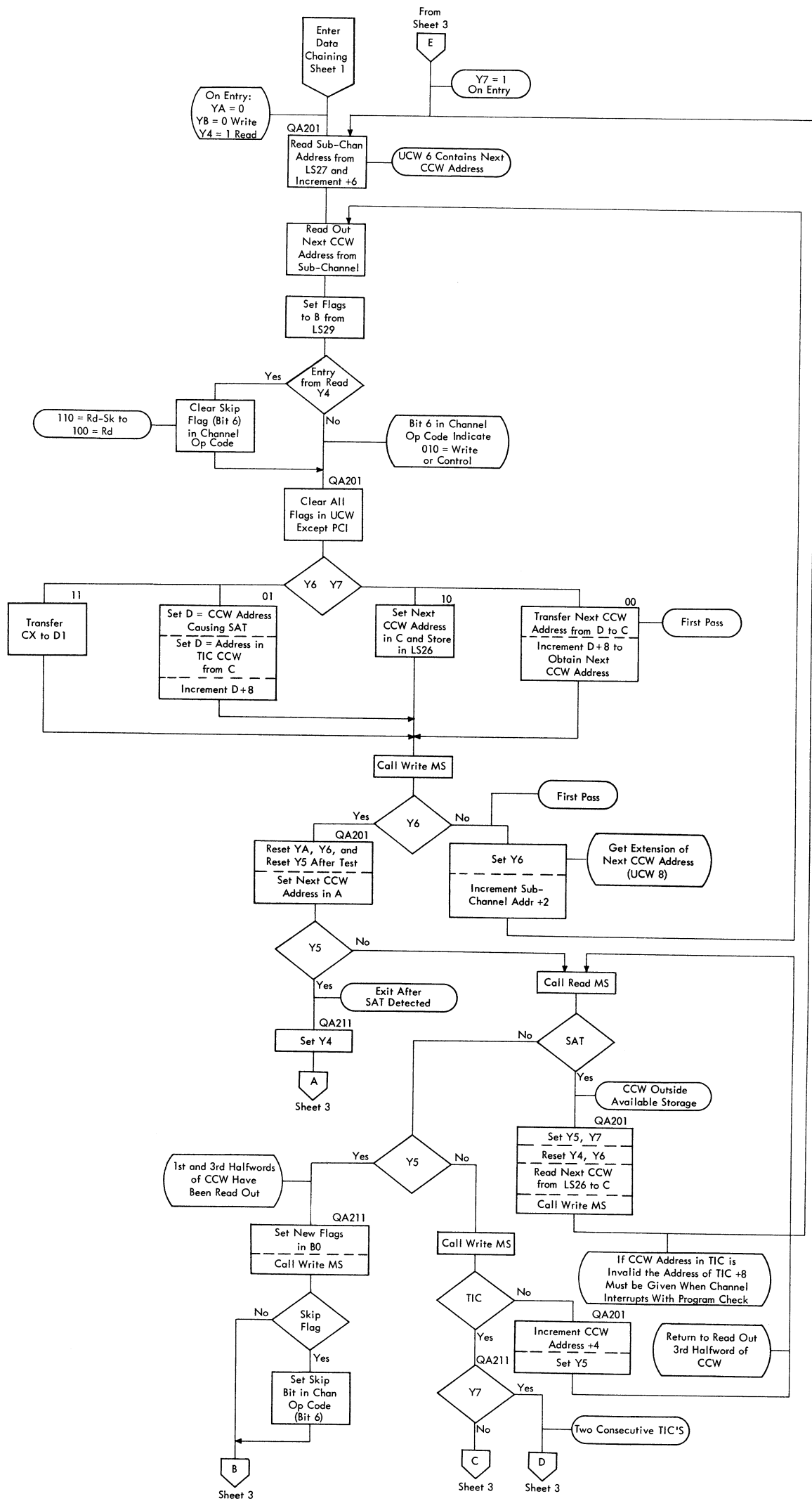
C

D

E

F

G



- 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 hfrwd 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 hfrws 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	W L R	Prog Ck	Prot Ck		I C C				Extention Next CCW Address
UCW6									Next CCW Address
UCW4	C D A	C C	S I L I	S K I P	P C I	Op Code	CT=0	End Stat Rch	Extention Data Address
UCW2							Data Address		Unit Status at End Time
UCW0							Count		

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

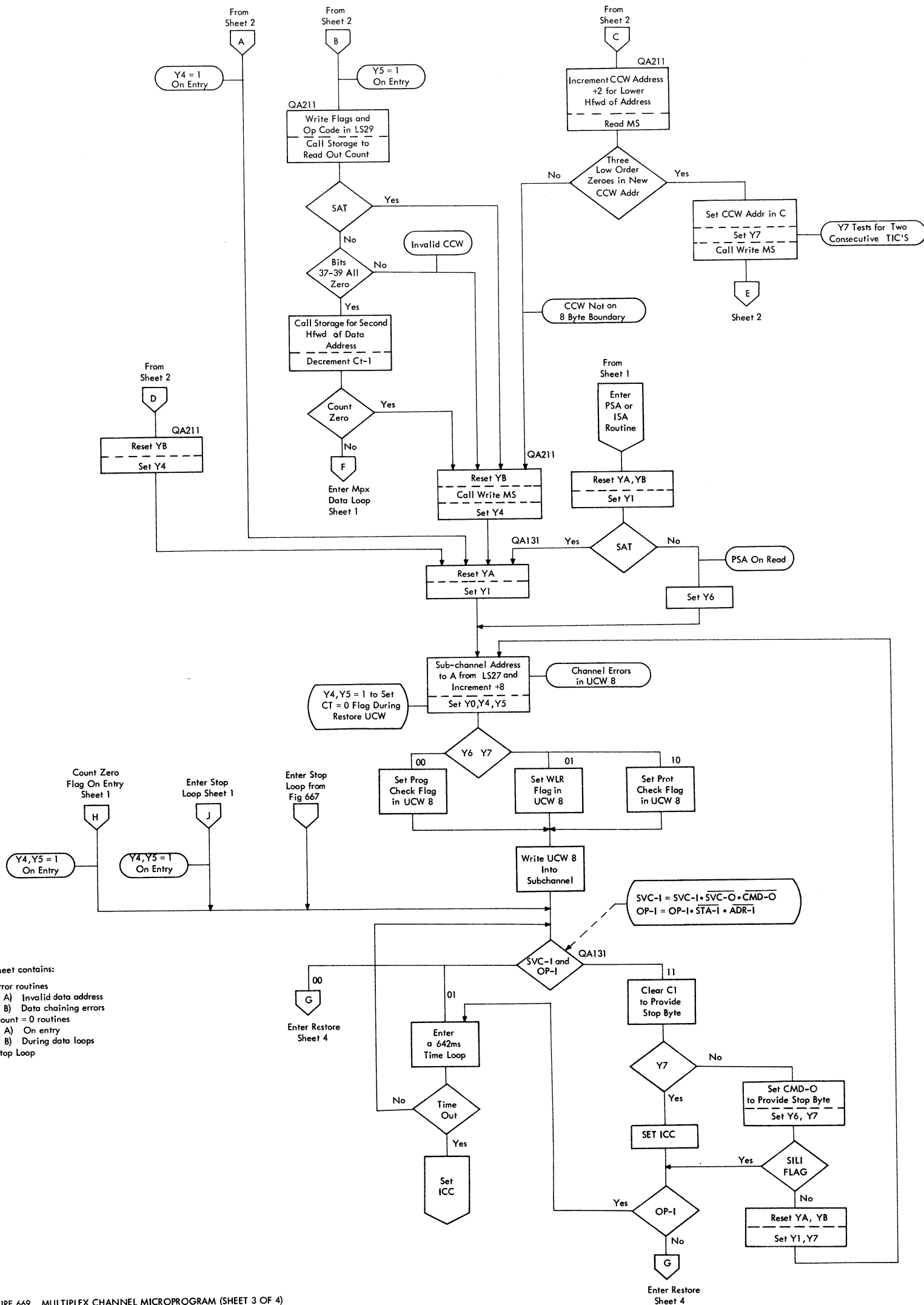
A

B

C

D

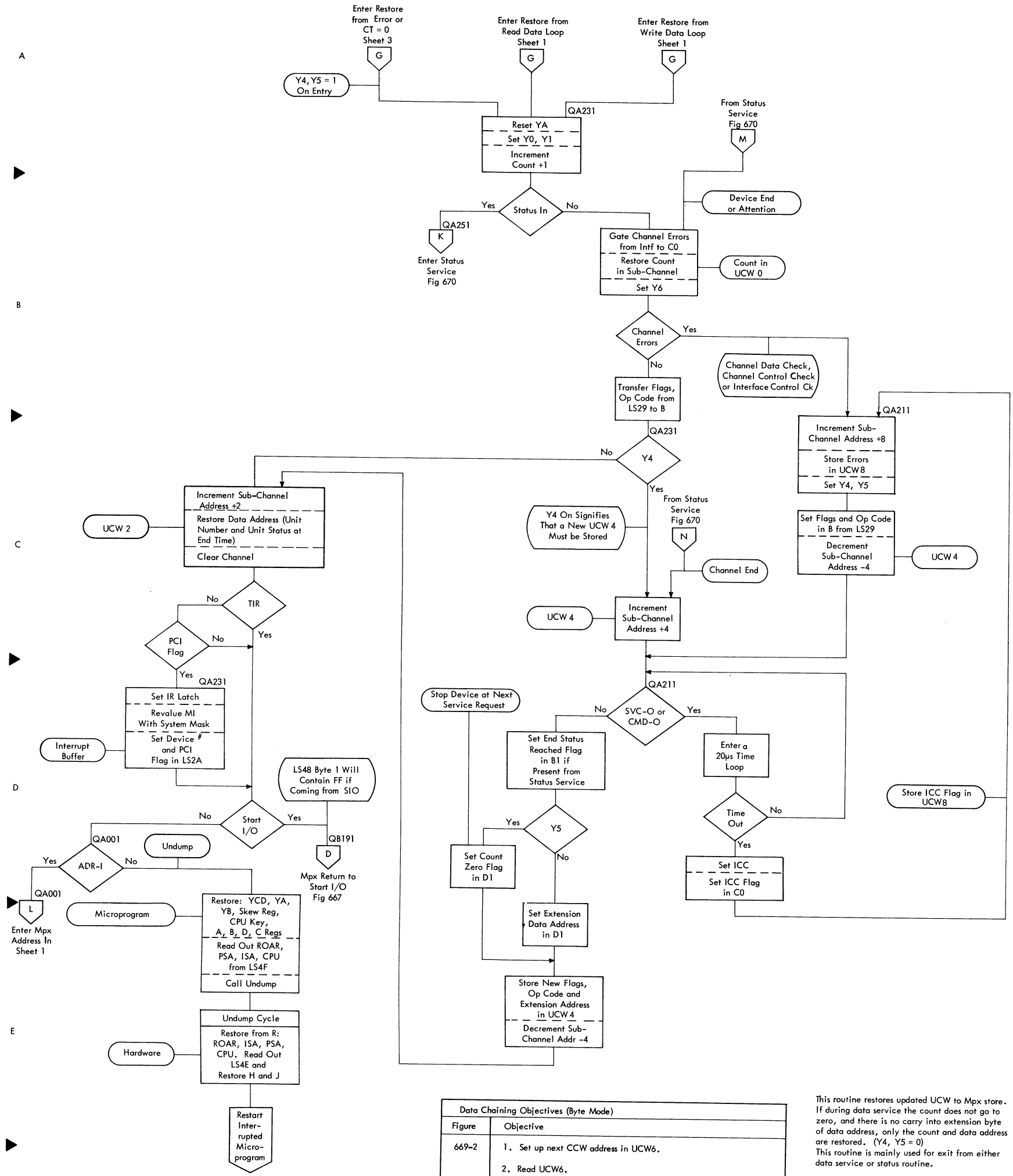
E



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)



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.

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).
669-3	c. If not a TIC, read out third halfword of CCW. 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.

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

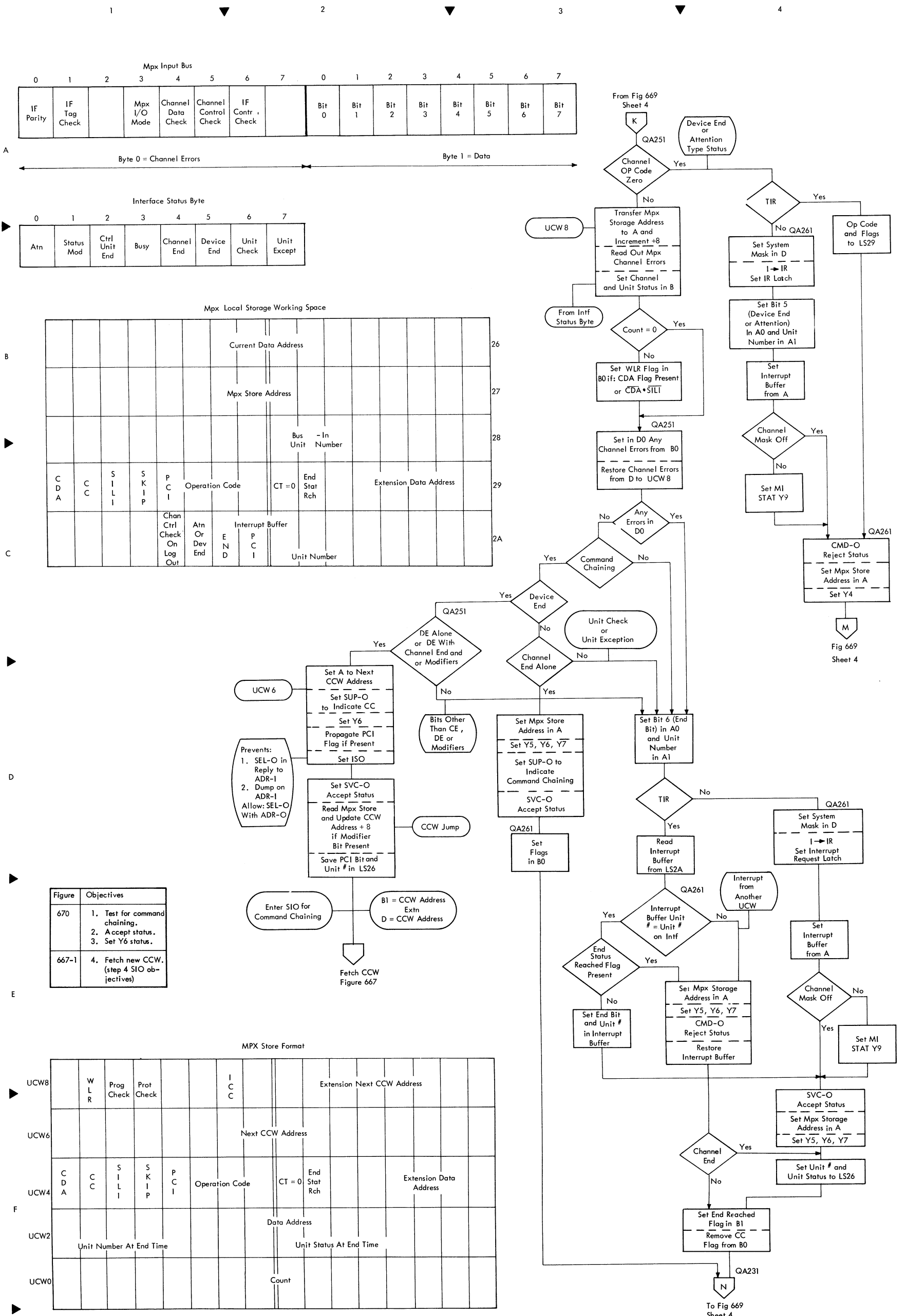


FIGURE 670. MULTIPLEX CHANNEL STATUS



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.

A  
B  
C  
D  
E  
F  
G

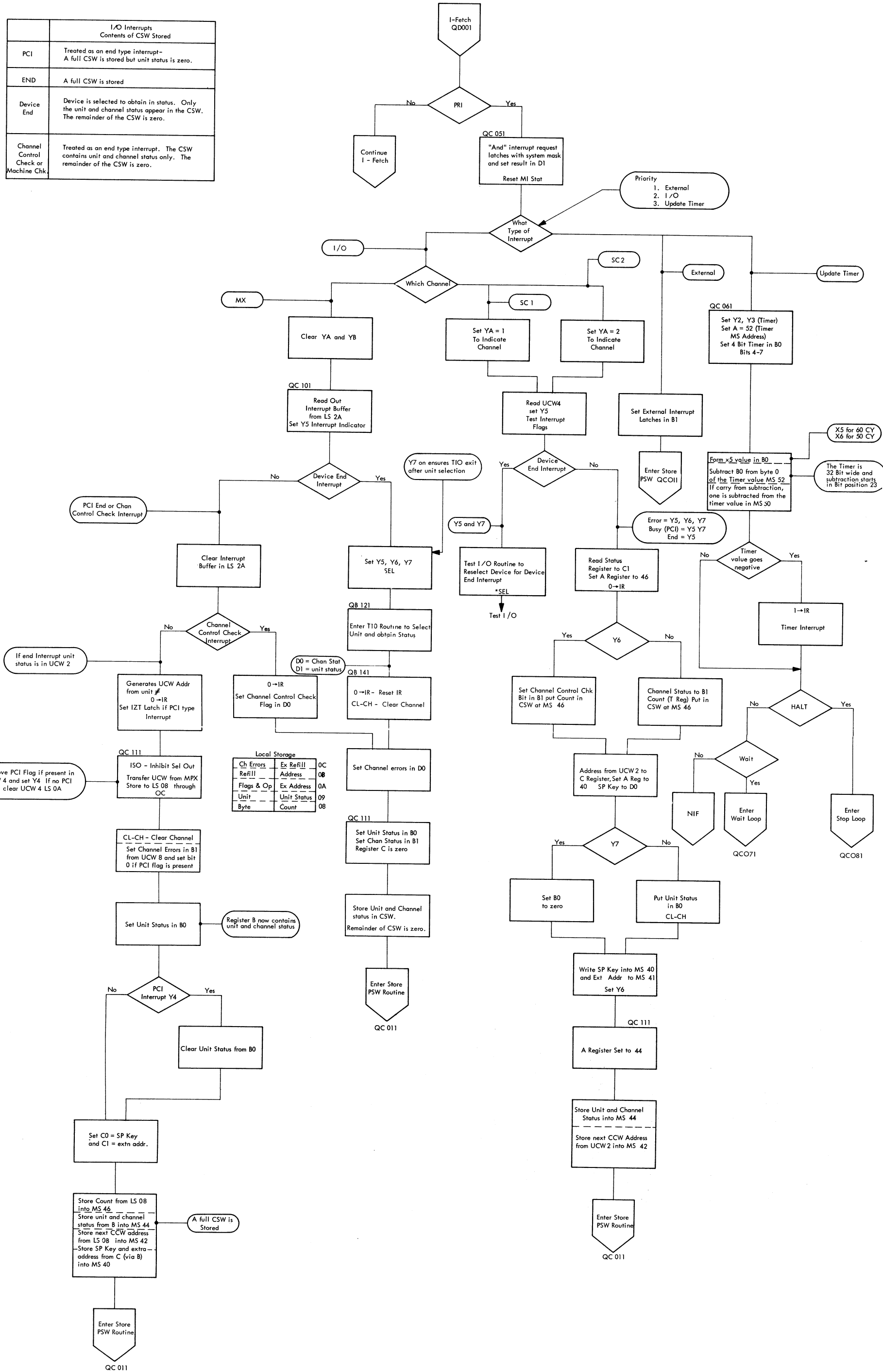


FIGURE 671. I/O INTERRUPTS AND UPDATE TIMER MICROPROGRAM

A  
B  
C  
D  
E  
F

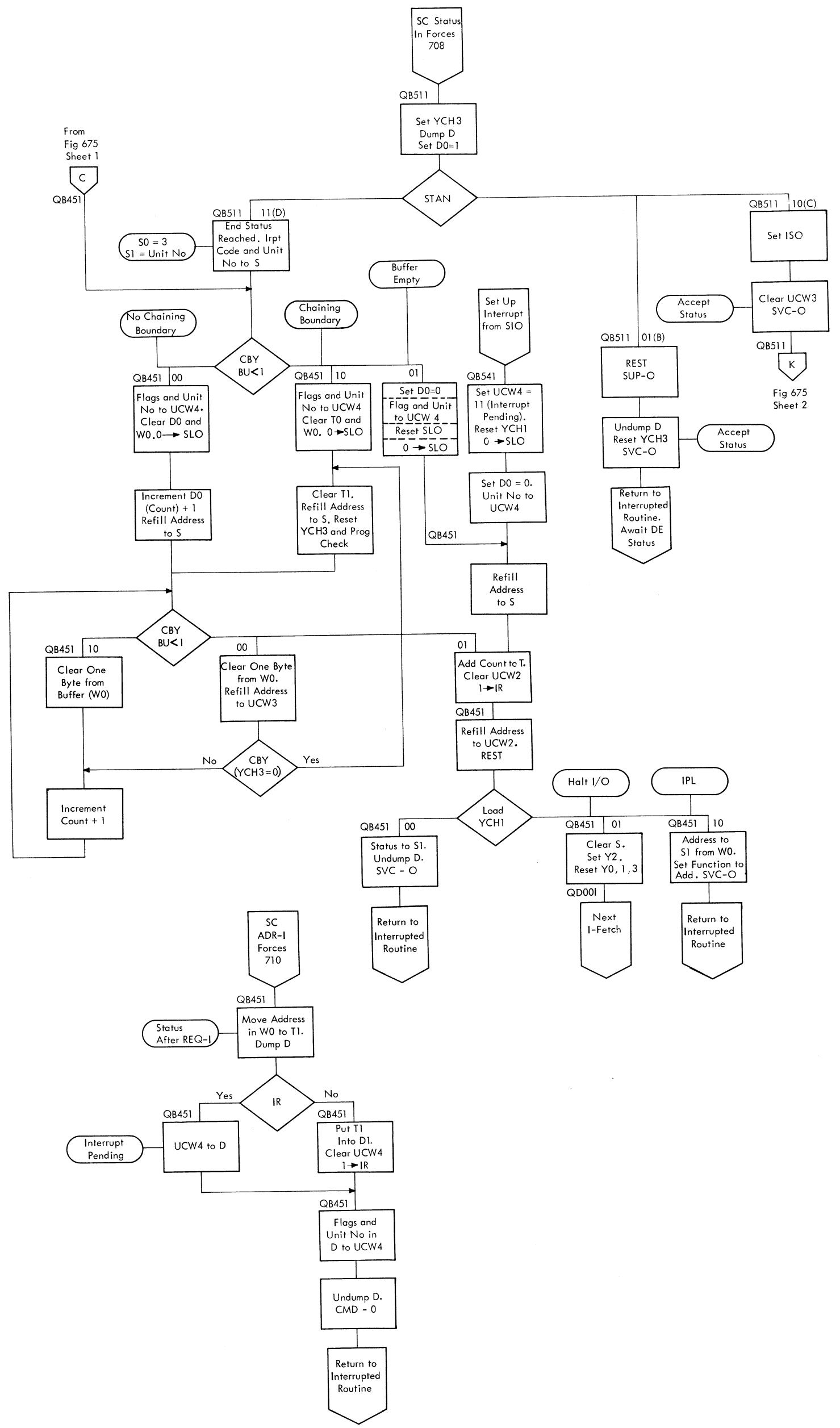


FIGURE 674. SELECTOR CHANNEL STATUS

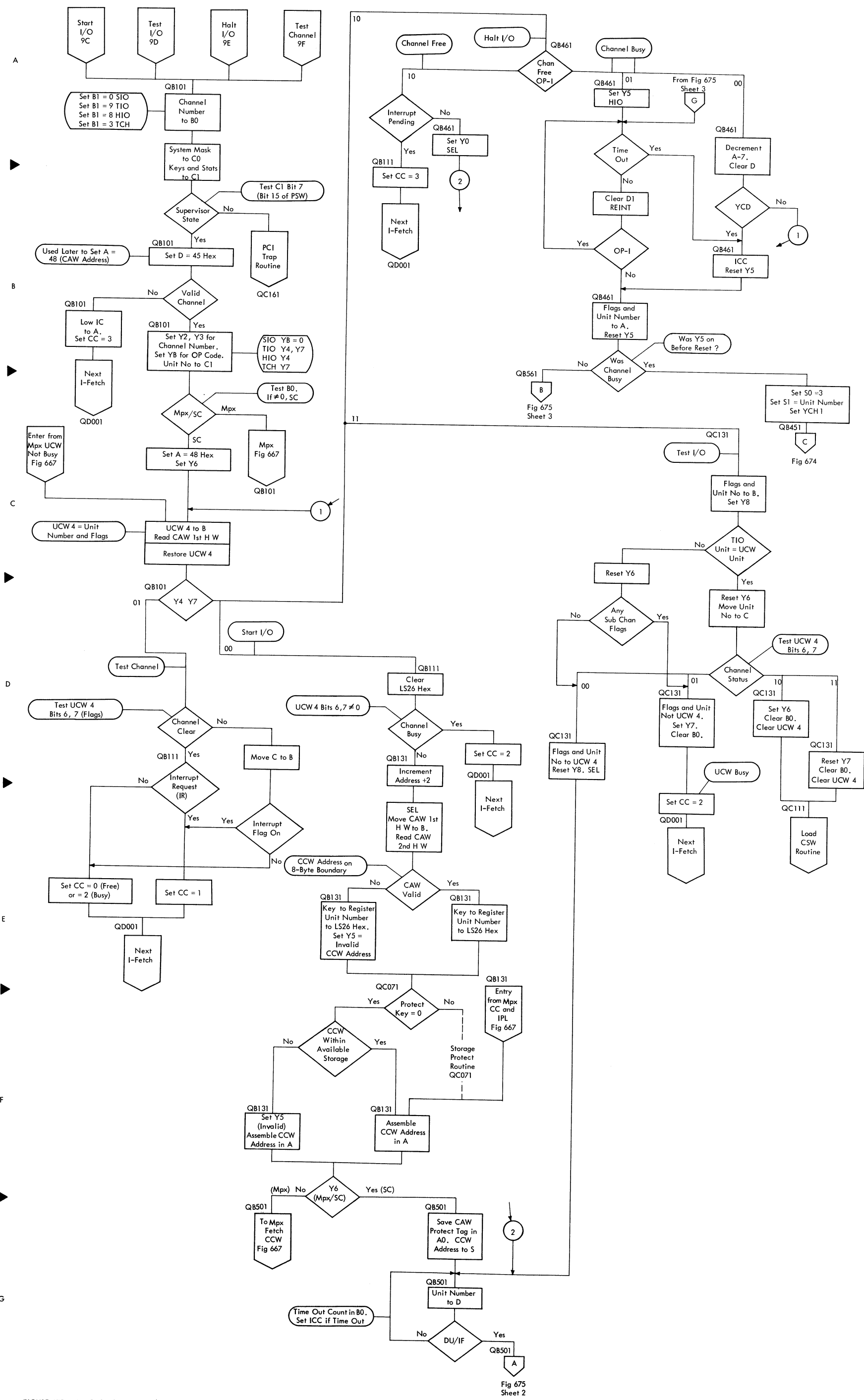


FIGURE 675. SELECTOR CHANNEL I/O INSTRUCTIONS MICROPROGRAM (SHEET 1 OF 4)

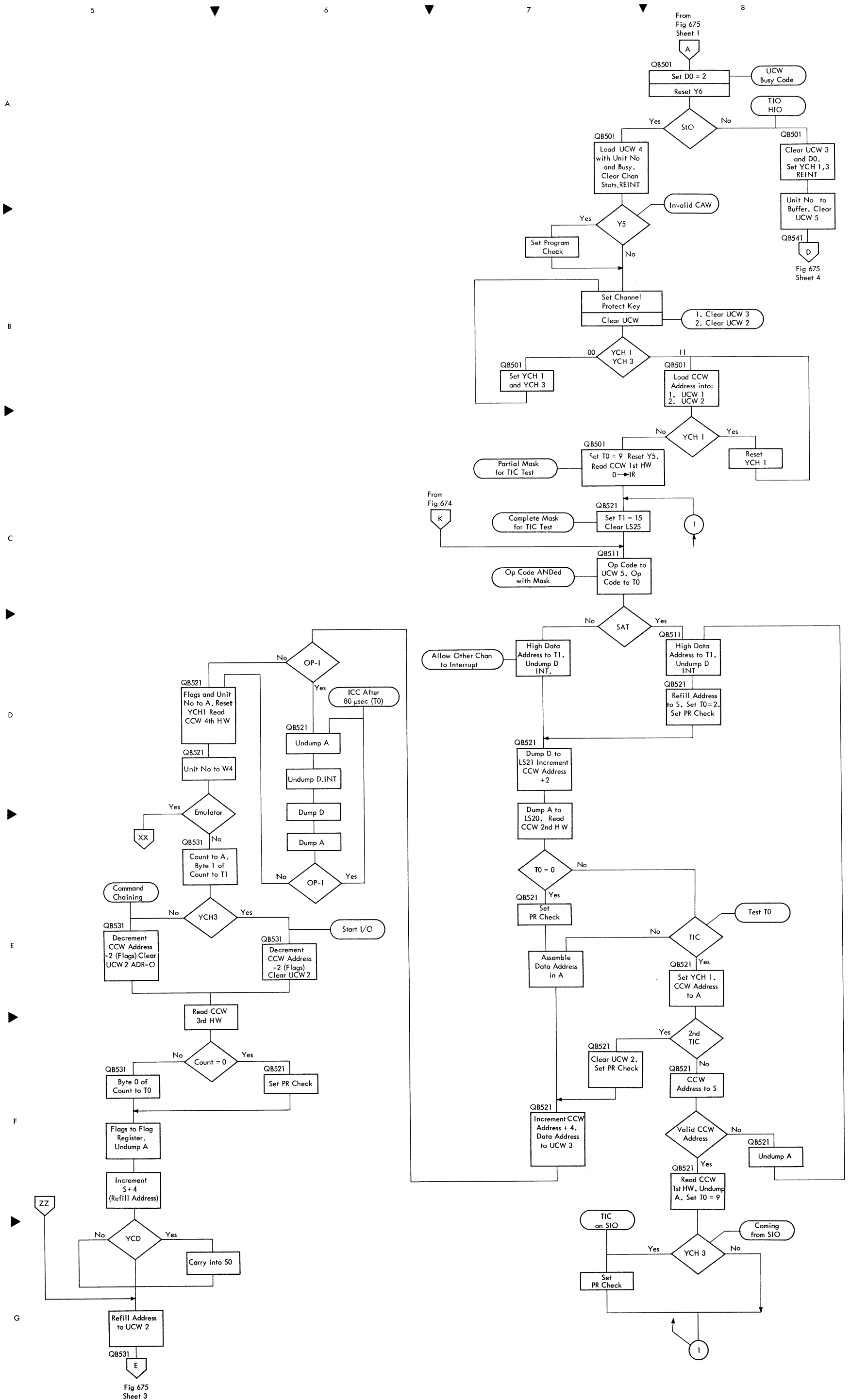


FIGURE 675. SELECTOR CHANNEL I/O INSTRUCTIONS MICROPROGRAM (SHEET 2 OF 4)

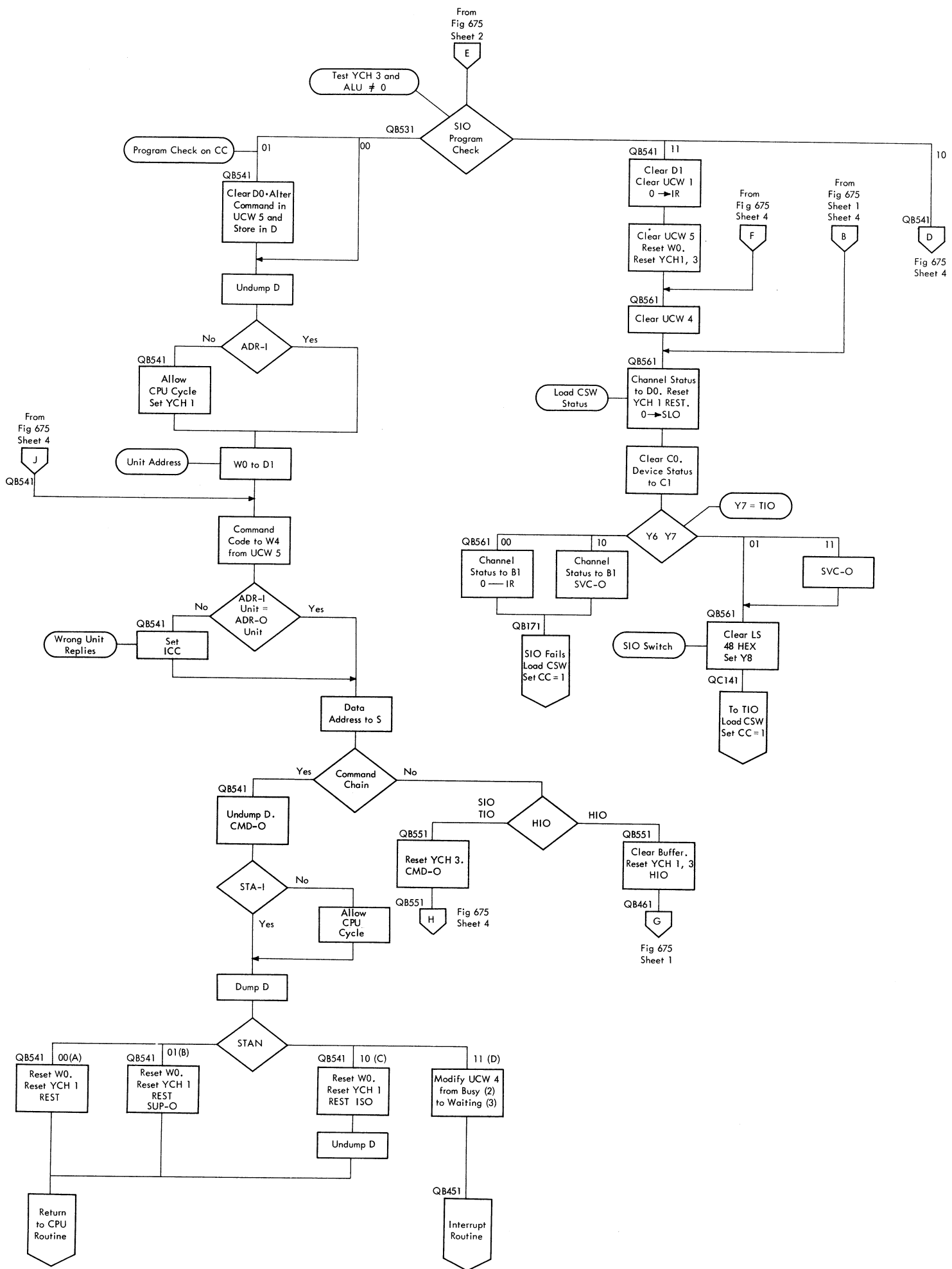


FIGURE 675. SELECTOR CHANNEL I/O INSTRUCTIONS MICROPROGRAM (SHEET 3 OF 4)

A

B

C

D

E

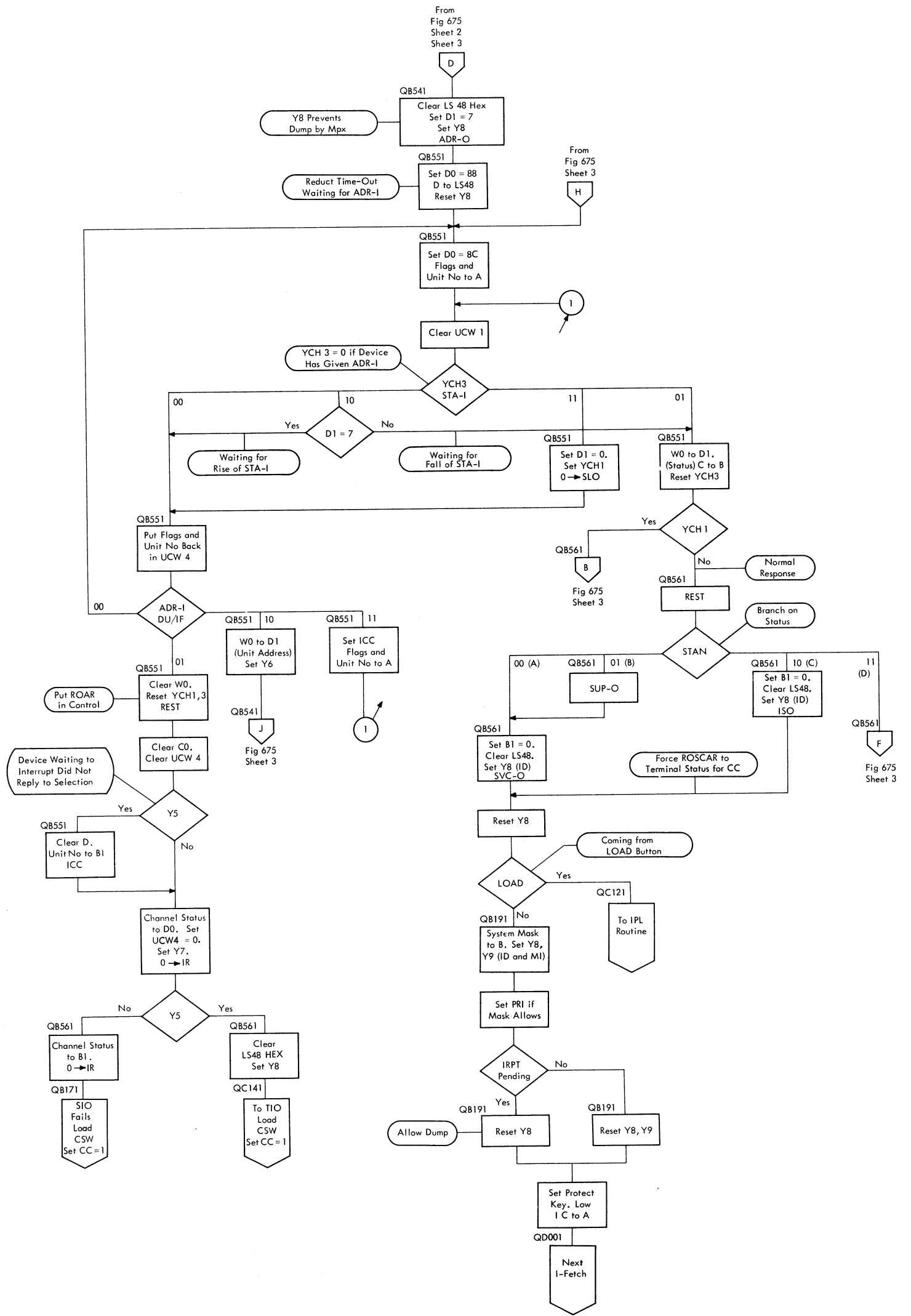


FIGURE 675. SELECTOR CHANNEL I/O INSTRUCTIONS MICROPROGRAM (SHEET 4 OF 4) (11/2/65)

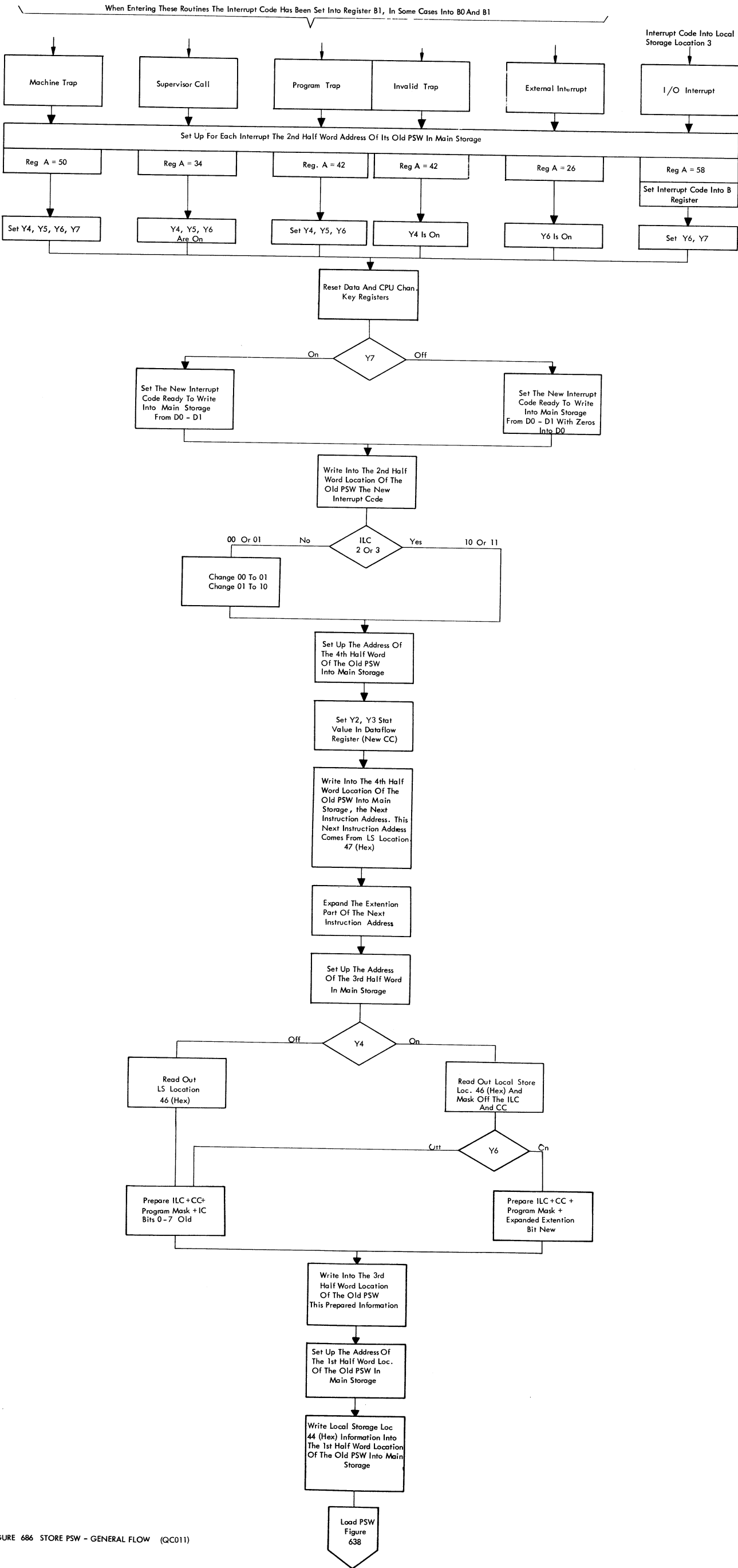
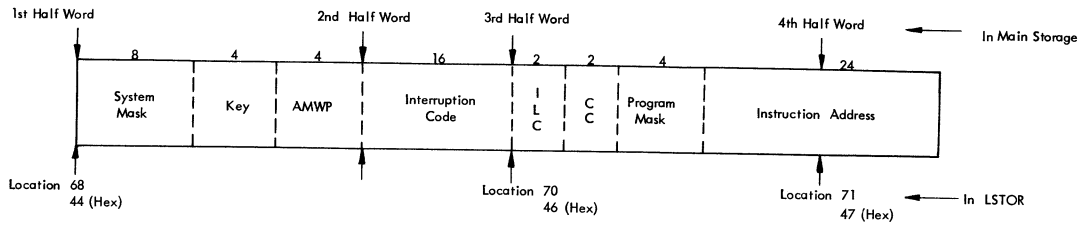


FIGURE 686 STORE PSW - GENERAL FLOW (QC011)

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 elimination is far less susceptible to misleading conclusions
7. Always look for some common relationship between fault symptoms

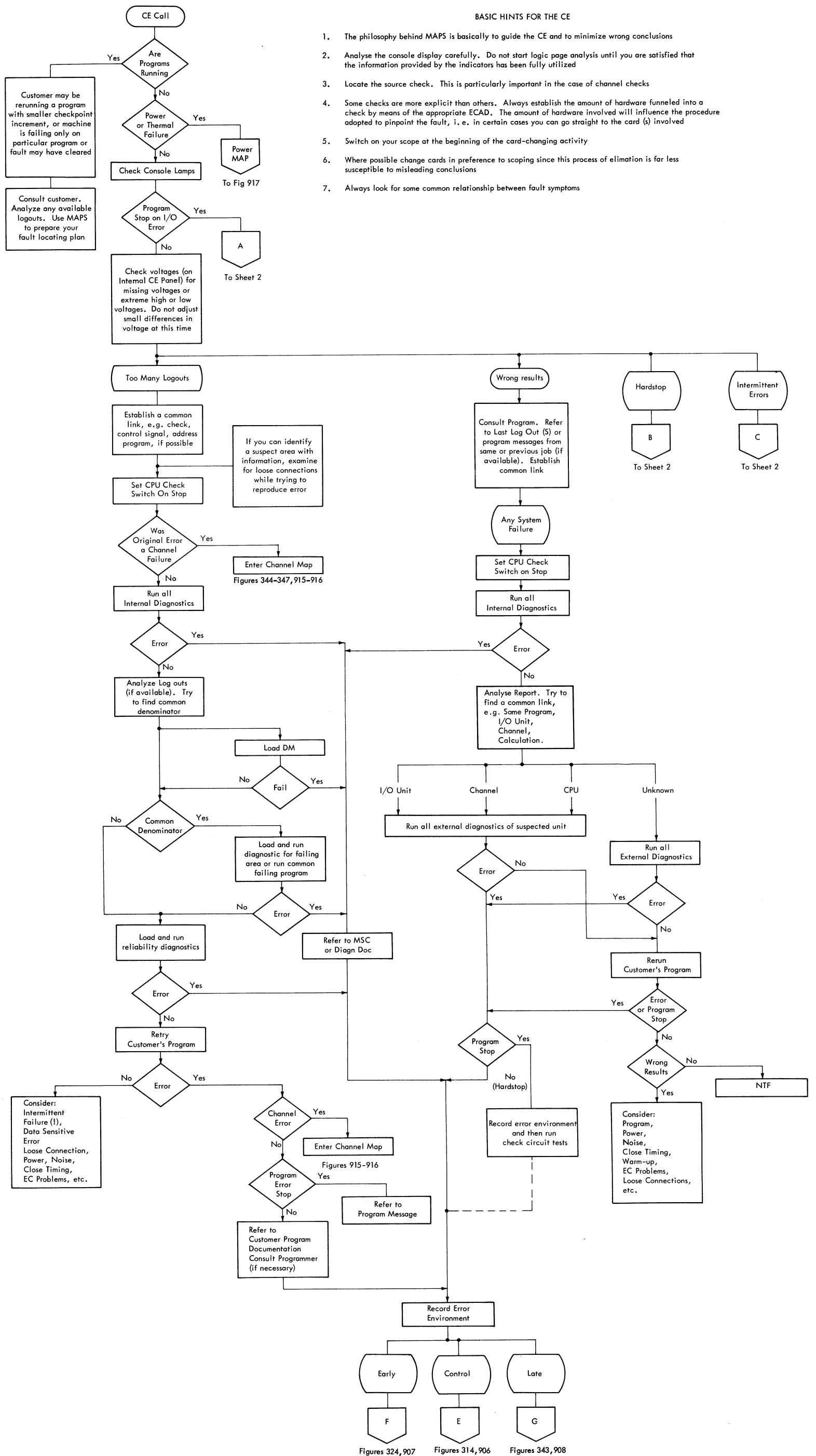


FIGURE 901. INTERPRET ERRORS (SHEET 1 OF 2)



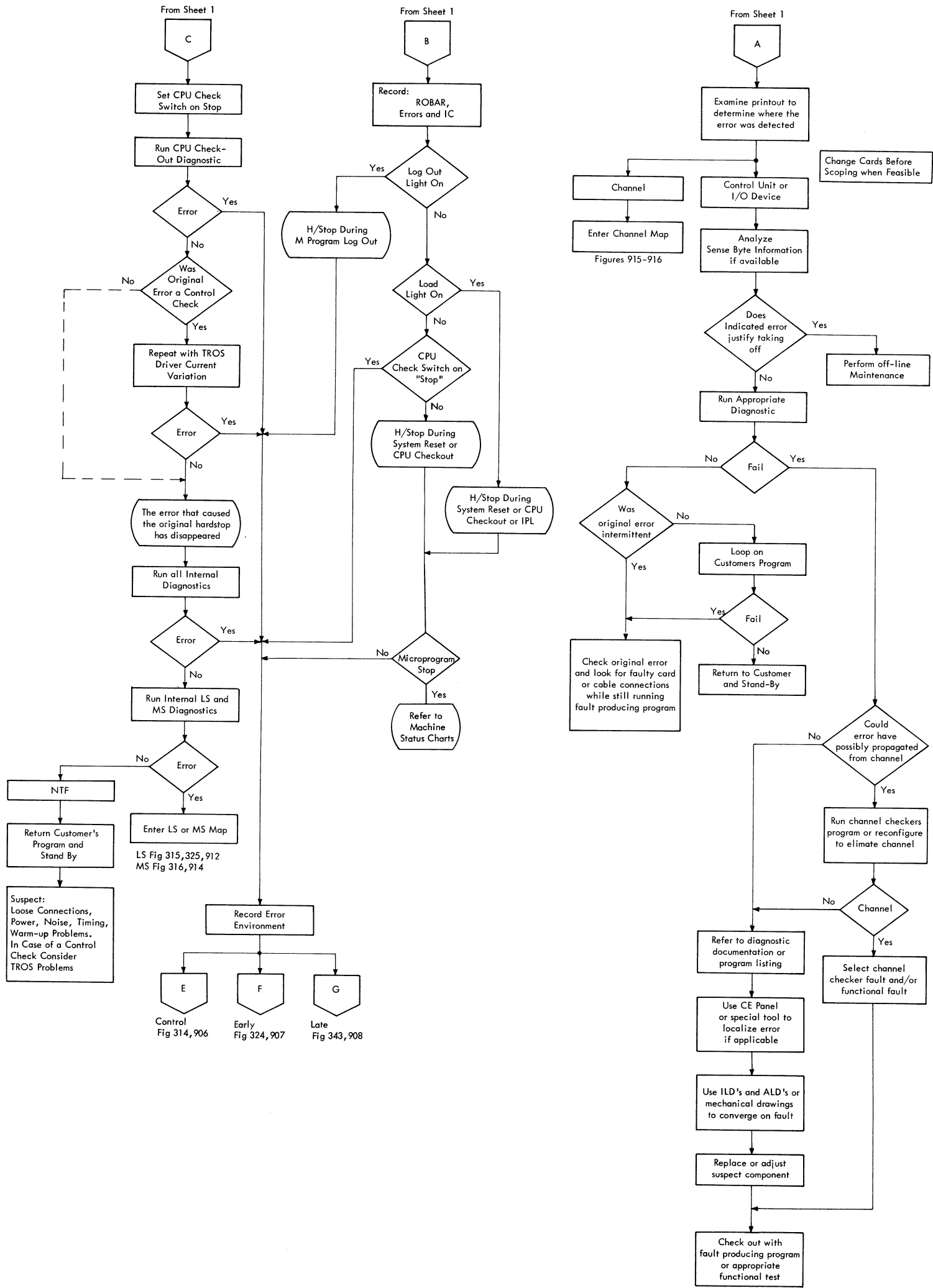


FIGURE 901. INTERPRET ERRORS (SHEET 2 OF 2)

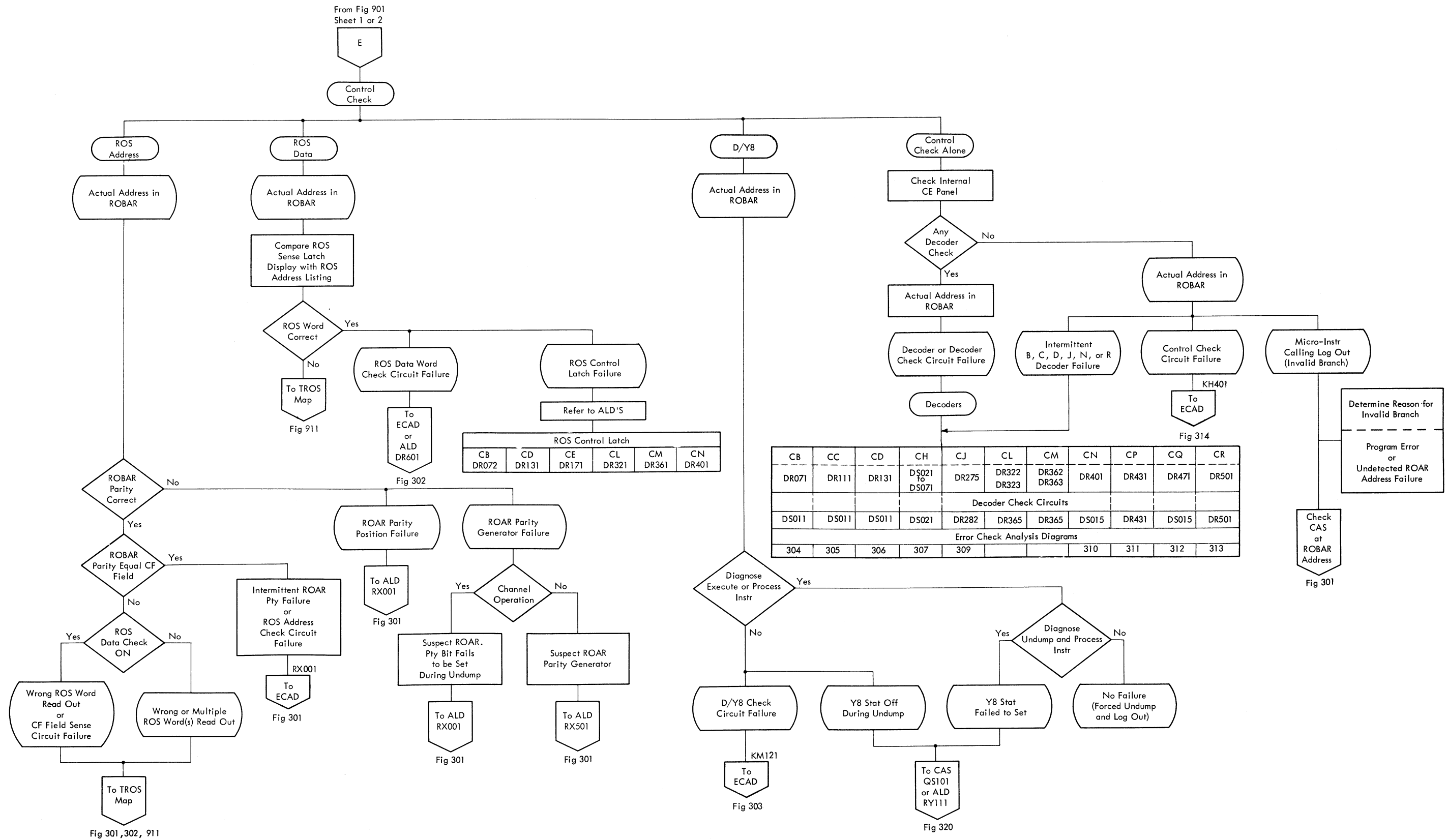


FIGURE 906. CONTROL CHECK

From Fig 901  
Sheet 1 or 2

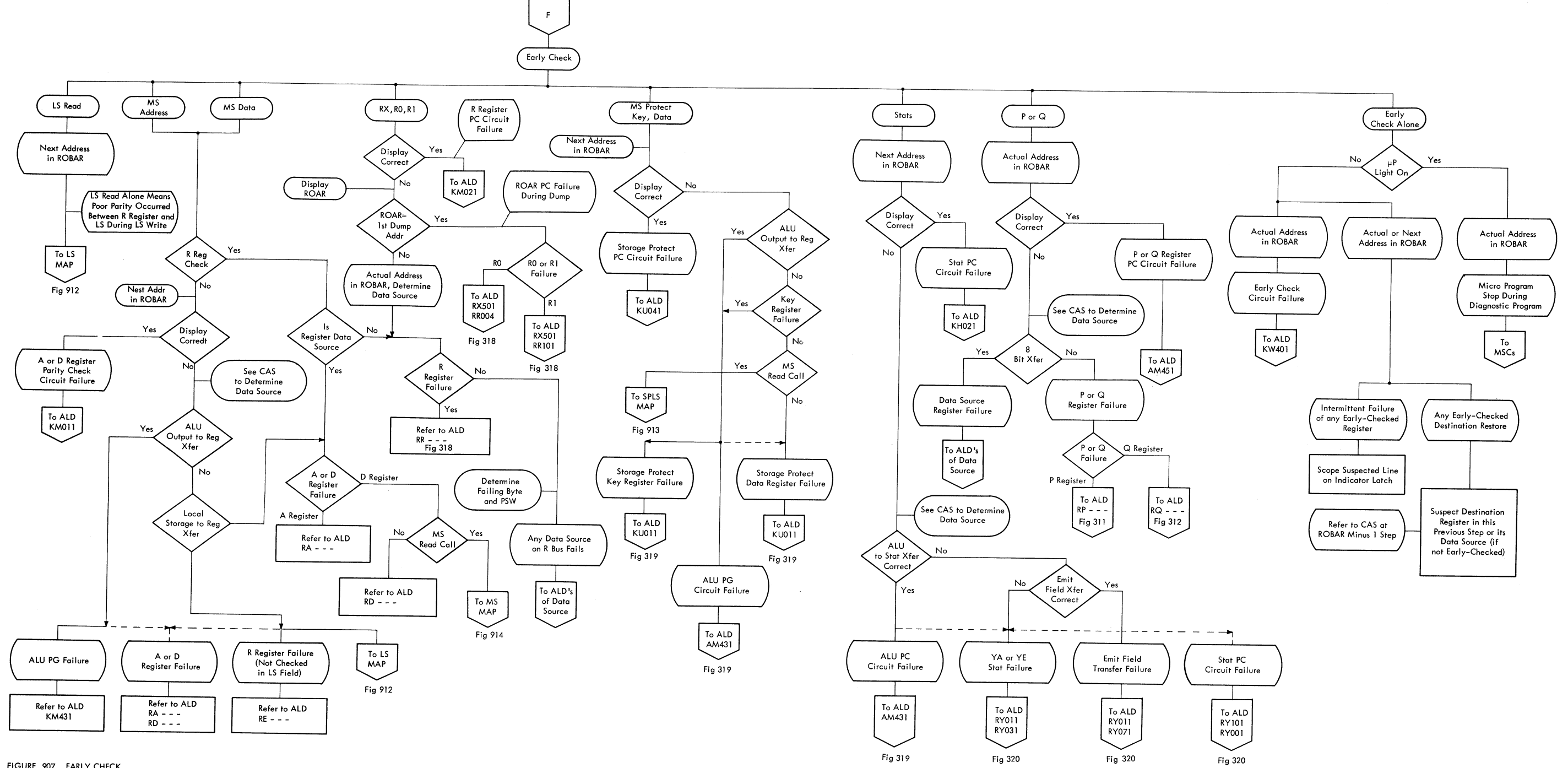


FIGURE 907. EARLY CHECK

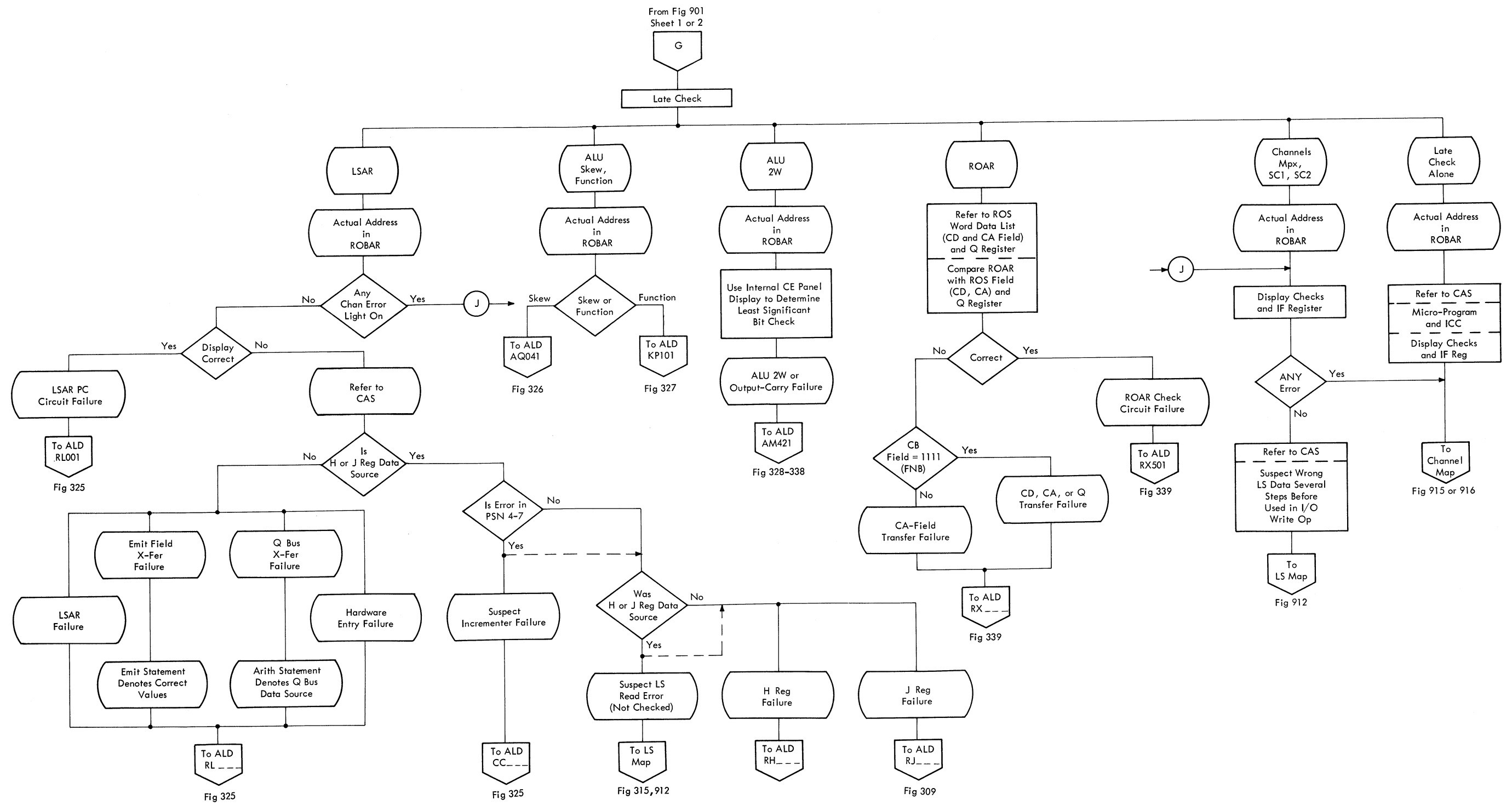


FIGURE 908. LATE CHECK

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

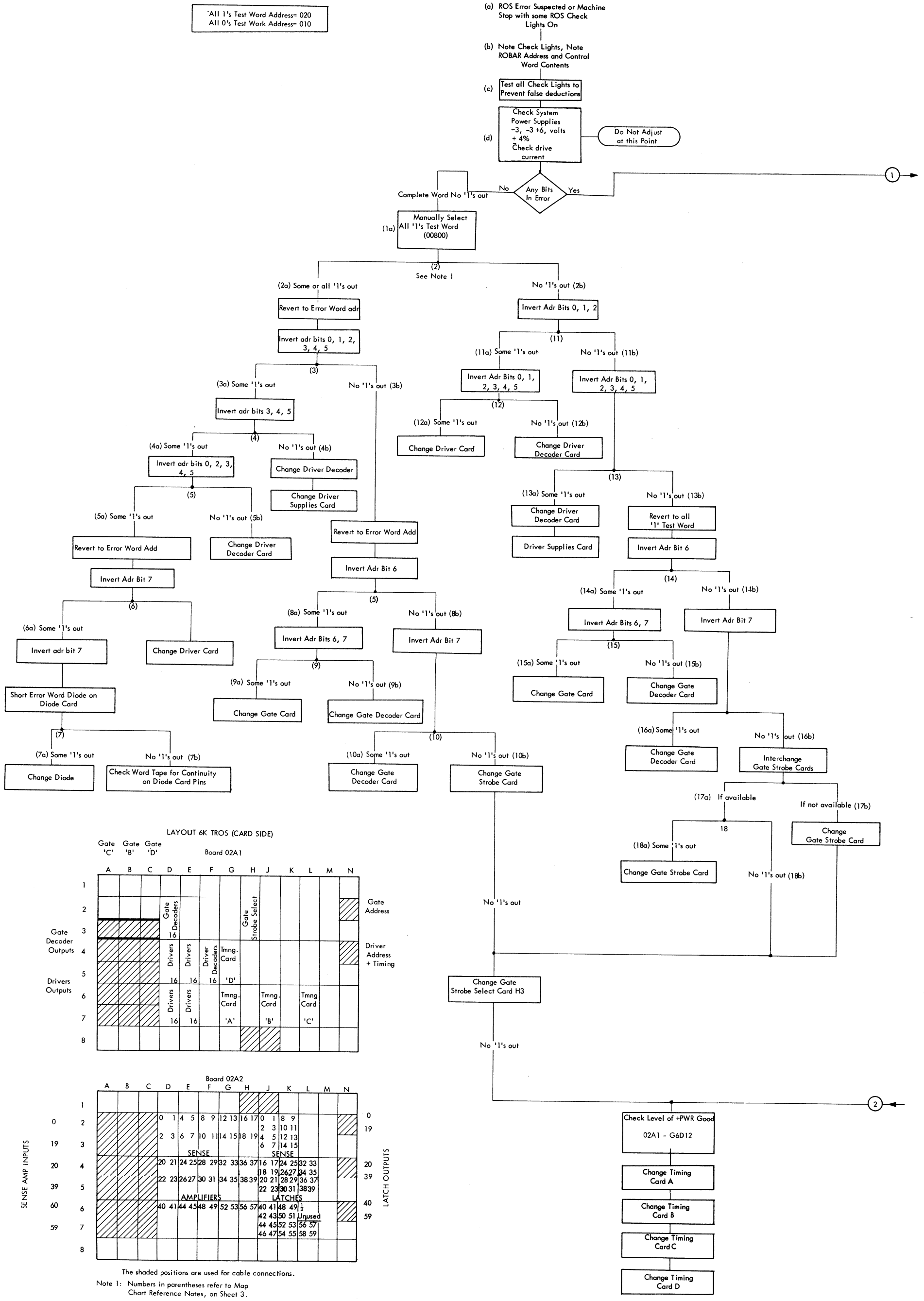


FIGURE 911. READ ONLY STORAGE (SHEET 1 OF 3)

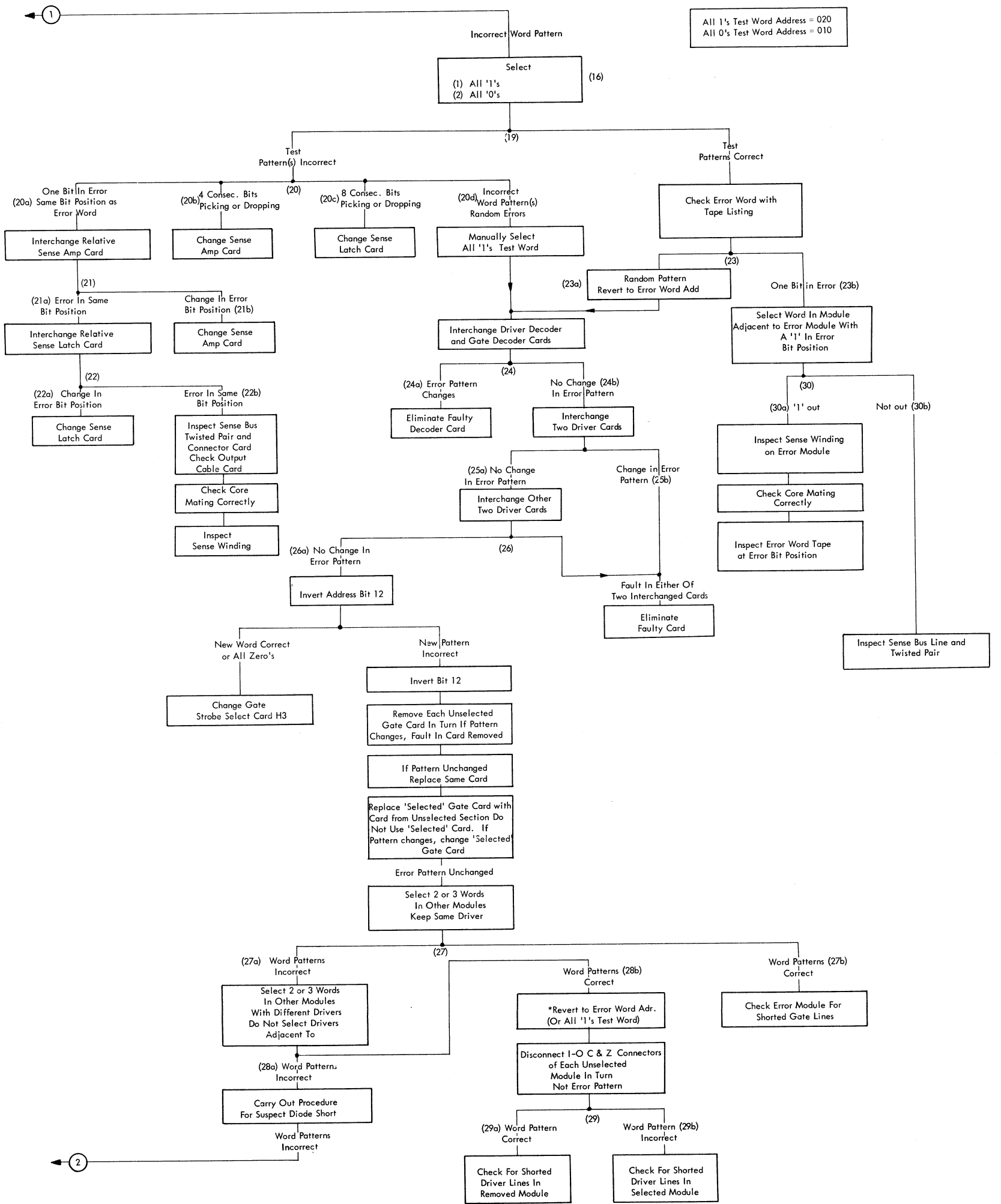


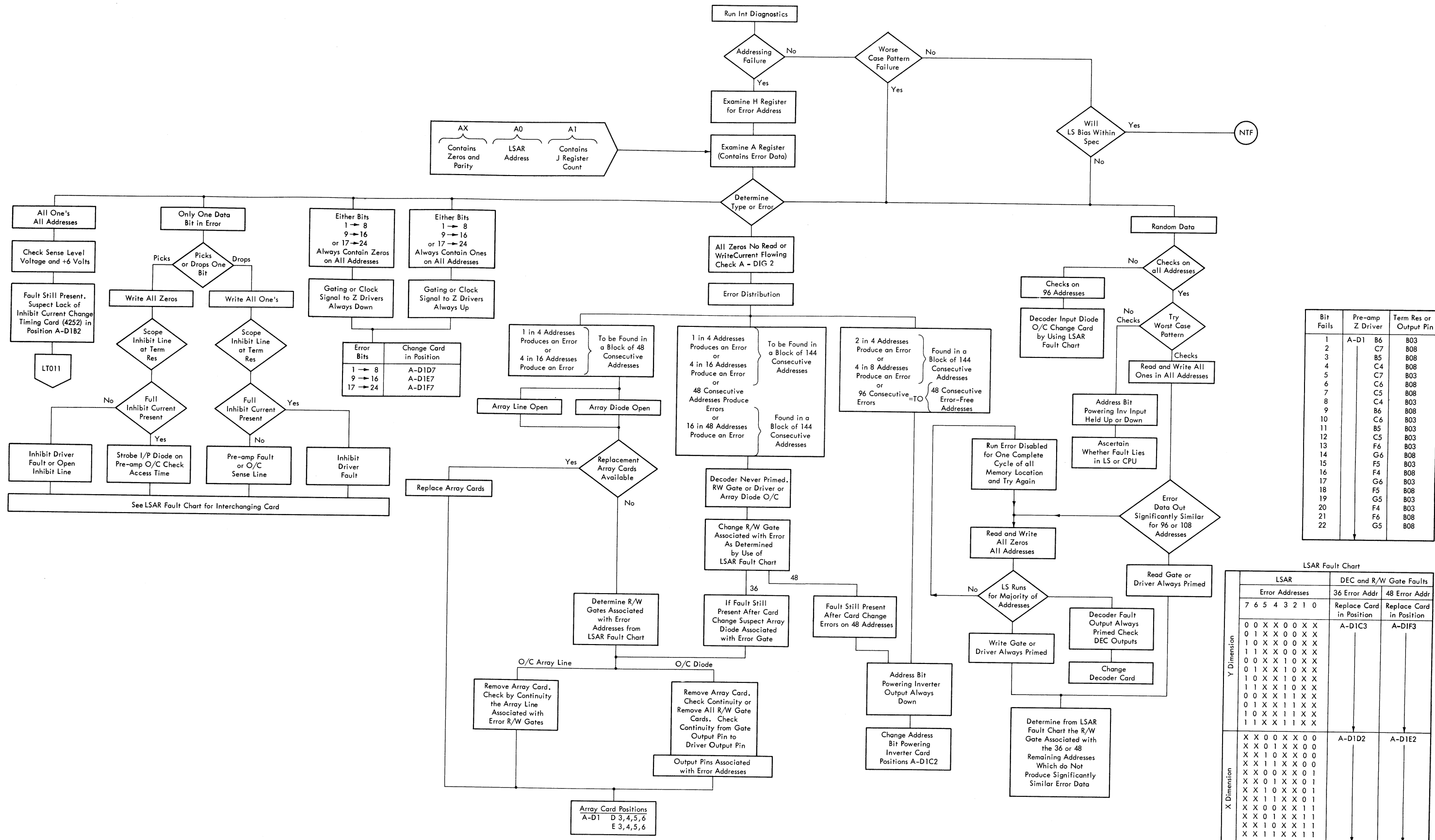
FIGURE 911. READ ONLY STORAGE (SHEET 2 OF 3)

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)	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	
5	
5	
7	
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)	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.
11 through 18	
16 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).
17 b)	
1 b)	If incorrect word patterns 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	One particular bit in error is isolated by the interchange of cards and tests 21 and 22.
22	
24	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.
25	
26	
27	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	
28 b)	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.
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)	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.
30	

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



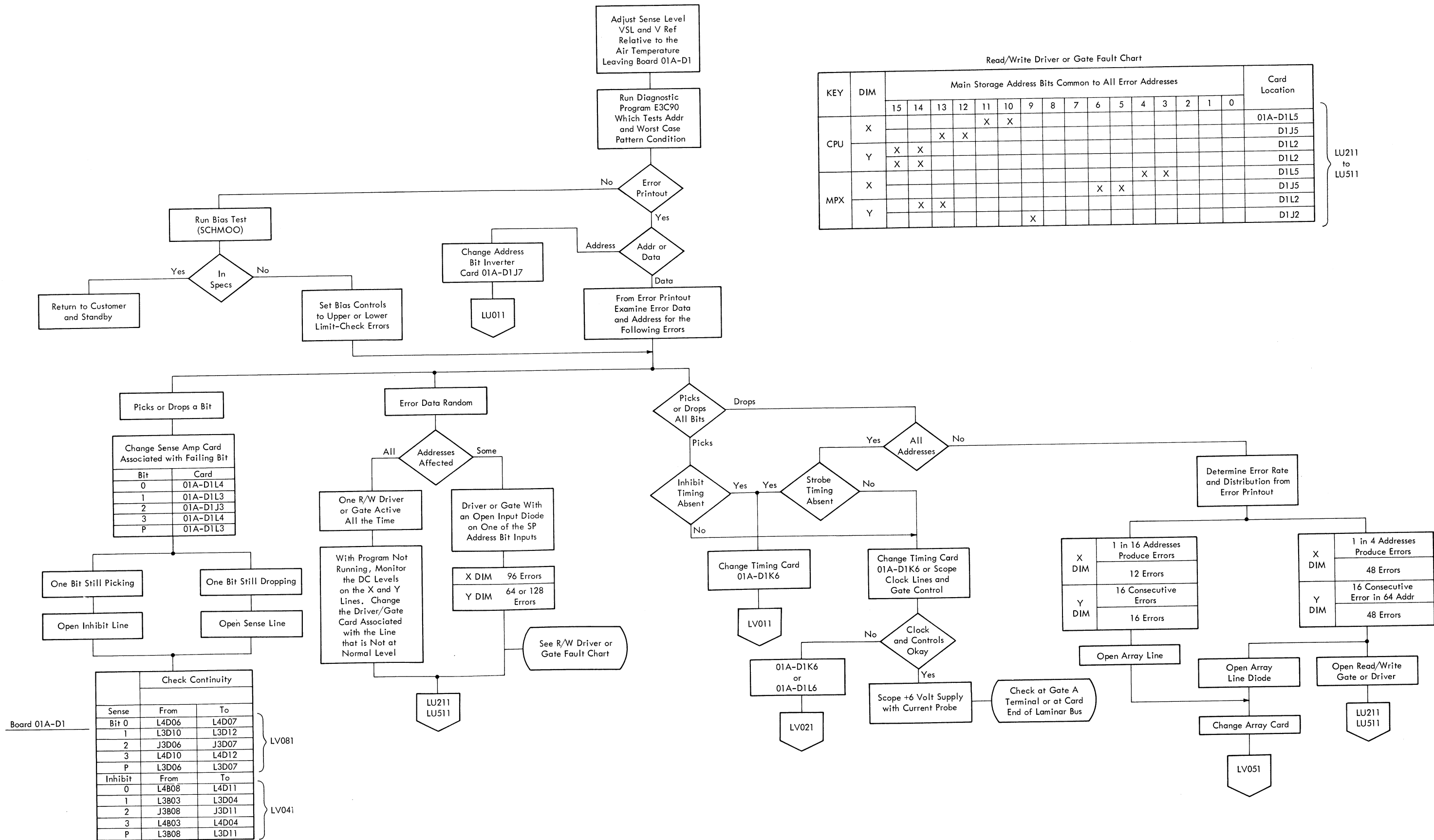
Bit Fails	Pre-amp Z Driver	Term Res or Output Pin
1	A-D1	B6 B03
2	B6	C7 B08
3	B5	C4 B08
4	C4	C7 B03
5	C7	C6 B08
6	C6	C5 B08
7	C5	C4 B03
8	C4	B6 B08
9	B6	C6 B03
10	C6	B5 B03
11	B5	C5 B03
12	C5	F6 B03
13	F6	G6 B08
14	G6	F5 B03
15	F5	F4 B08
16	F4	G6 B03
17	G6	F5 B08
18	F5	G5 B08
19	G5	F4 B03
20	F4	F6 B08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

LSAR Fault Chart

Y Dimension	LSAR							DEC and R/W Gate Faults		
	7	6	5	4	3	2	1	0	36 Error Addr	48 Error Addr
	Error Addresses							Replace Card in Position	Replace Card in Position	
	0	0	X	X	0	0	X	X	A-DIC3	A-DIF3
	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
	Y	X	X		X													D1J5
MPX	X										X	X		X	X		D1L2	
	Y		X	X					X								D1L5	
																	D1J5	
																	D1L2	
																	D1J2	

} LU211 to LU511

Change Sense Amp Card Associated with Failing Bit

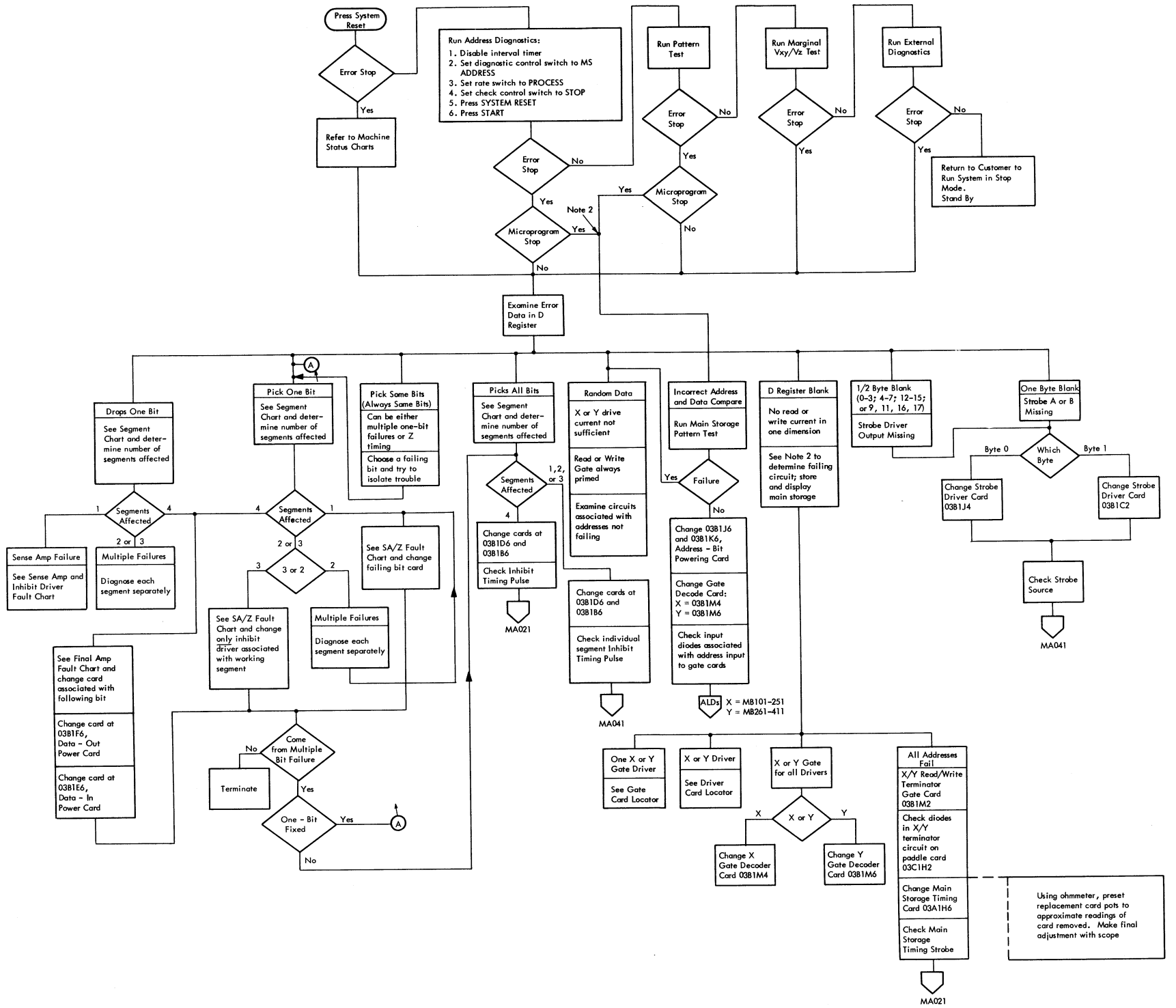
Bit	Card
0	01A-D1L4
1	01A-D1L3
2	01A-D1J3
3	01A-D1L4
P	01A-D1L3

Board 01A-D1

Check Continuity		
Sense Bit	From	To
0	L4D06	L4D07
1	L3D10	L3D12
2	J3D06	J3D07
3	L4D10	L4D12
P	L3D06	L3D07
Inhibit		
	From	To
0	L4B08	L4D11
1	L3B03	L3D04
2	J3B08	J3D11
3	L4B03	L4D04
P	L3B08	L3D11

} LV081  
} LV041

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	A
0 0 0	B
0 0 1	C
0 1 0	D
0 1 1	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

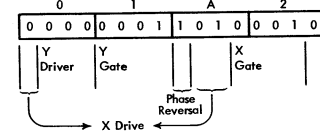
X and Y Drivers 0-3 = B1N4  
X and Y Drivers 4-7 = B1N6

Gate Card Locator

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

Notes:

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



- All card references are to frame 03.

FIGURE 914. MAIN STORAGE (64K)

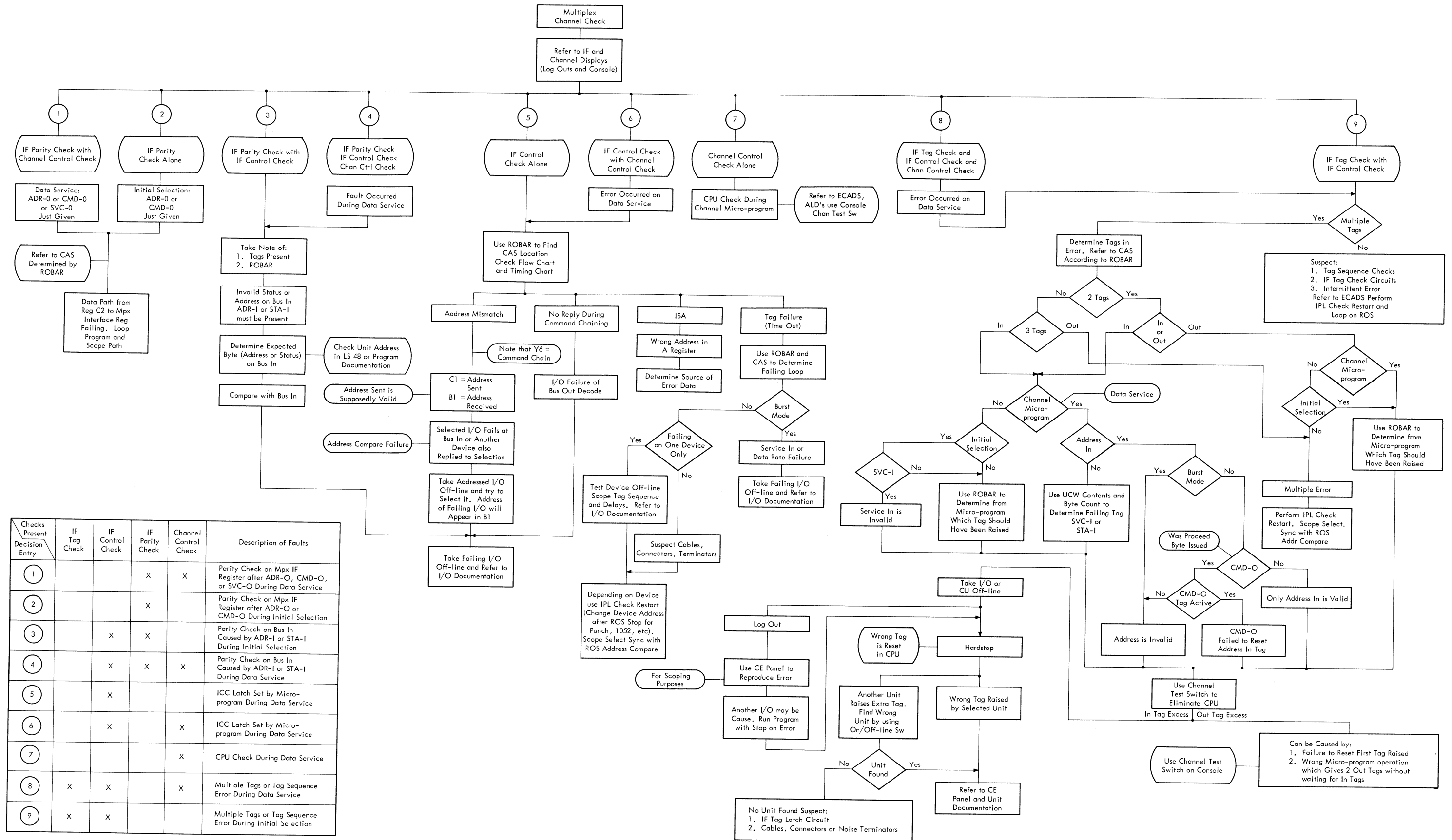


FIGURE 915. MULTIPLEX CHANNEL MAP

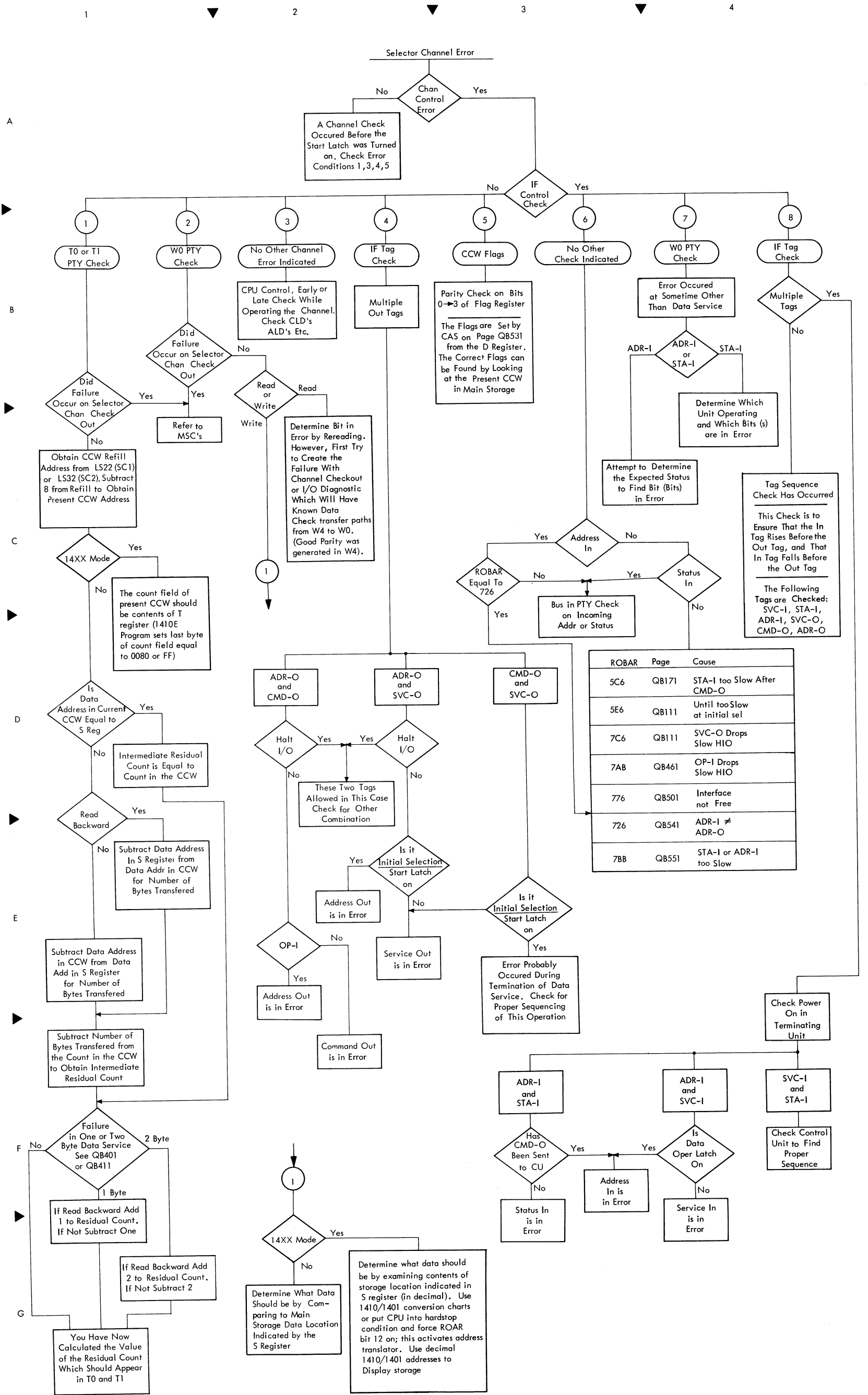


FIGURE 916. SELECTOR CHANNEL

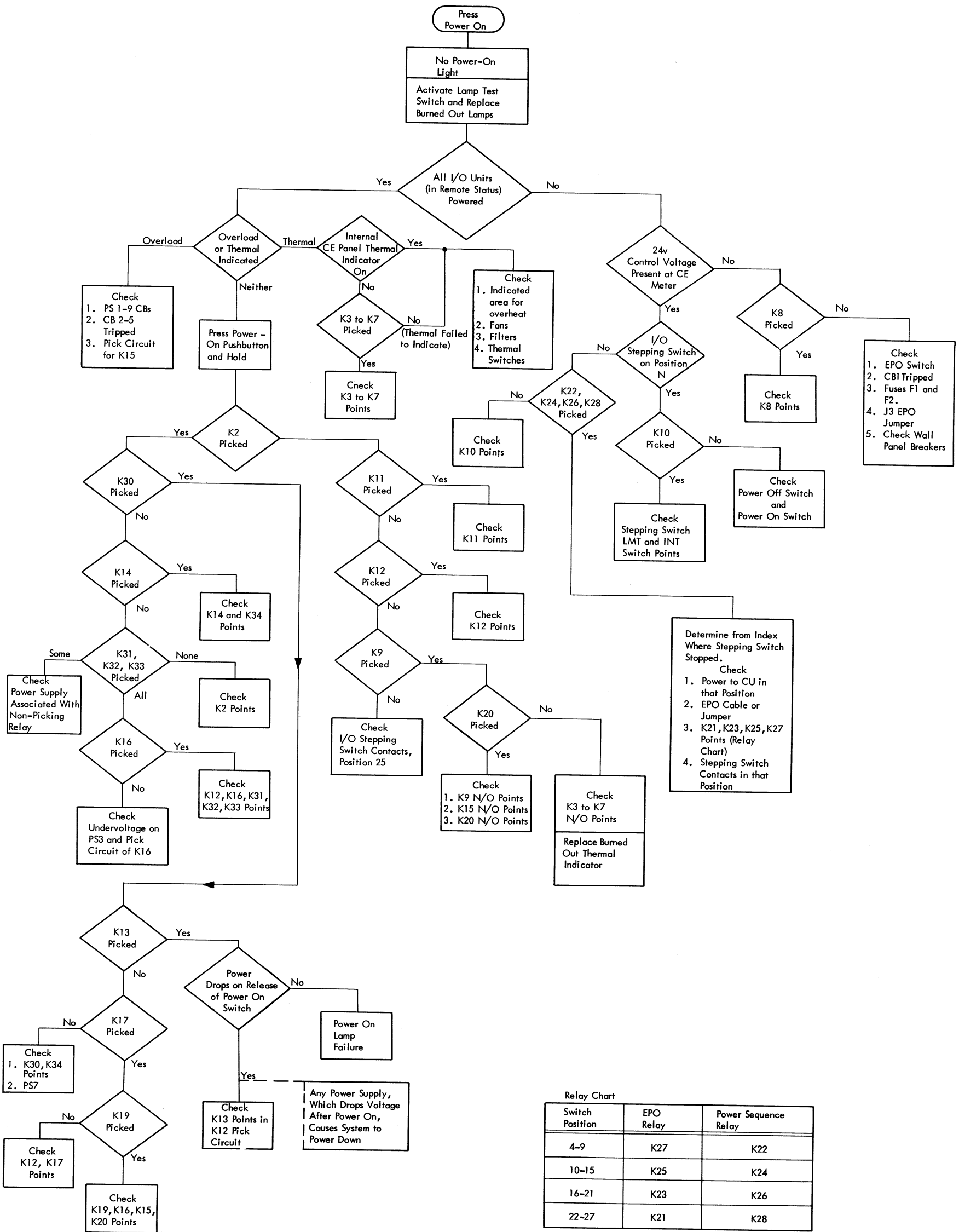


FIGURE 917. MID - PAC POWER SUPPLY

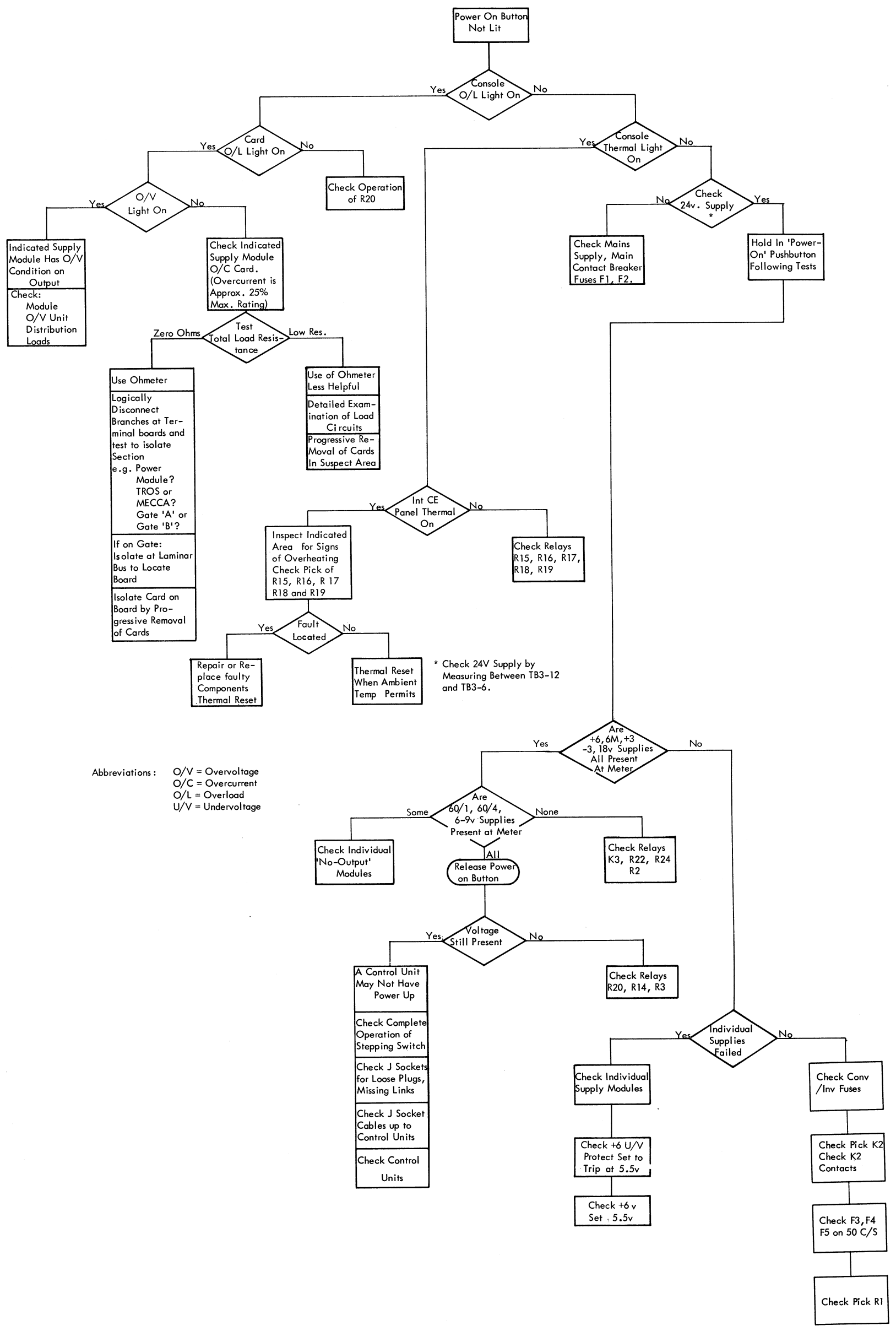
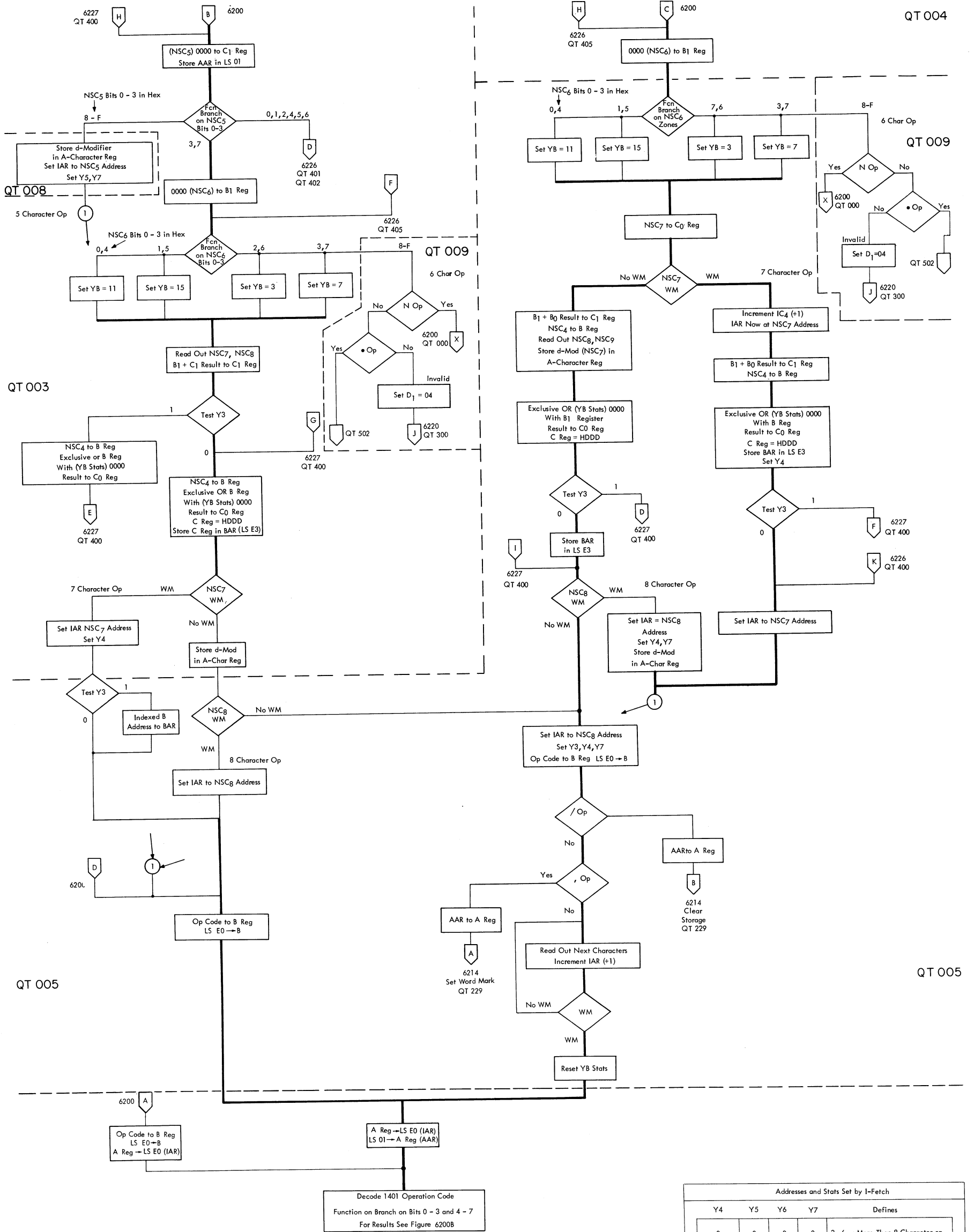


FIGURE 918. 2.5 KC HF POWER SUPPLY



QT 001

QT 005

QT 003

QT 008

QT 004

QT 009

QT 005

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

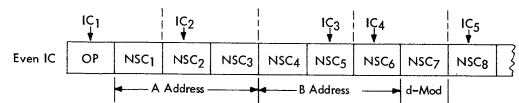
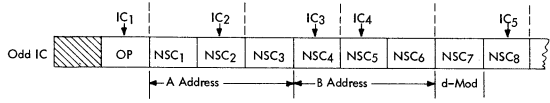
Op Code *		Bits 4 - 7															
Bits 0 - 3		XXXX0000	XXXX0001	XXXX0010	XXXX0011	XXXX0100	XXXX0101	XXXX0110	XXXX0111	XXXX1000	XXXX1001	XXXX1010	XXXX1011	XXXX1100	XXXX1101	XXXX1110	XXXX1111
1000XXXX		Invalid Op QT 307 or QT 308											• Halt QT 502	• Clear WM QT 221 6214	Invalid Op QT 307 or QT 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 NSC1	
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



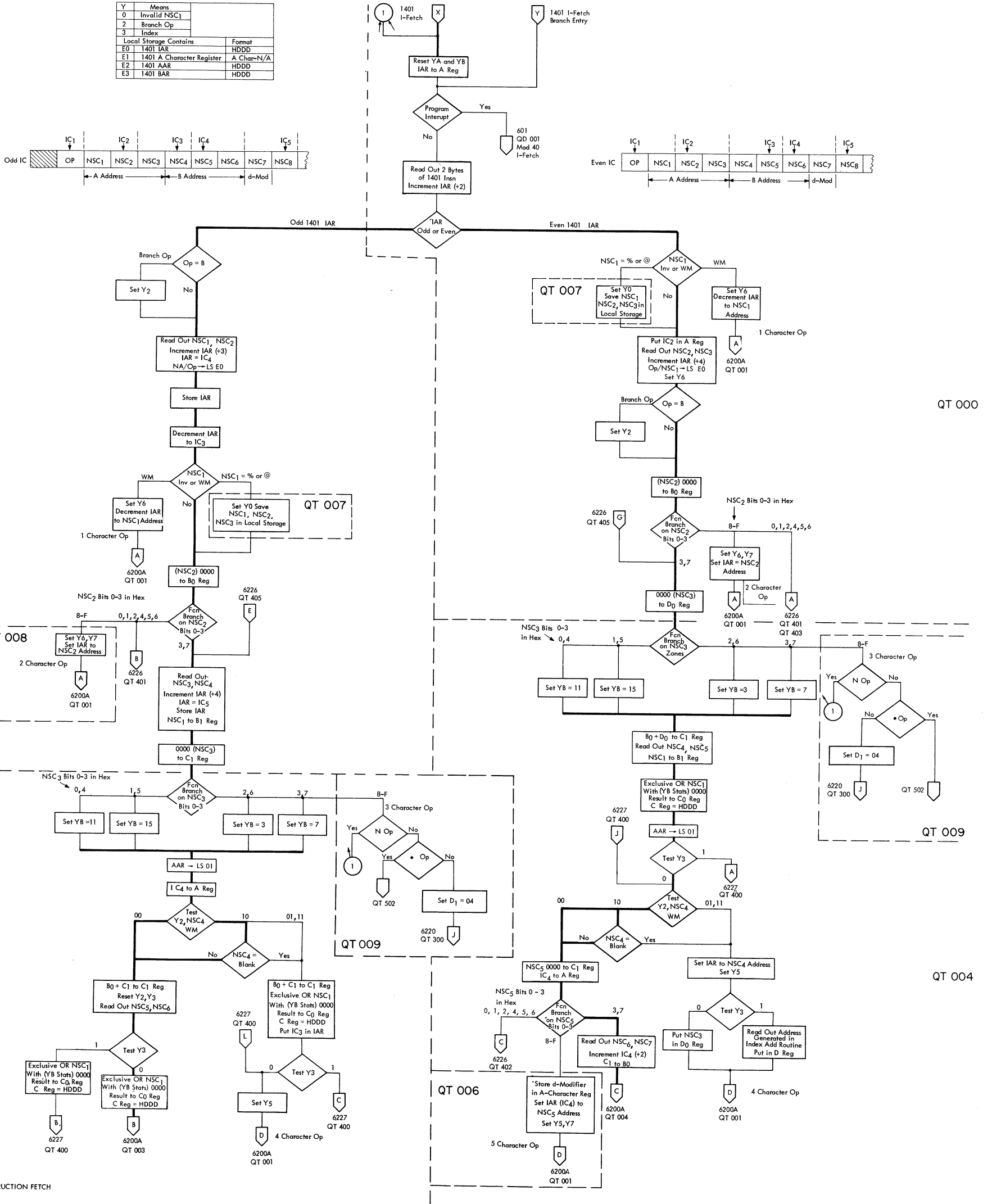
QT 002

QT 000

QT 003

QT 004

FIGURE 6200. 1401 INSTRUCTION FETCH



QT 009

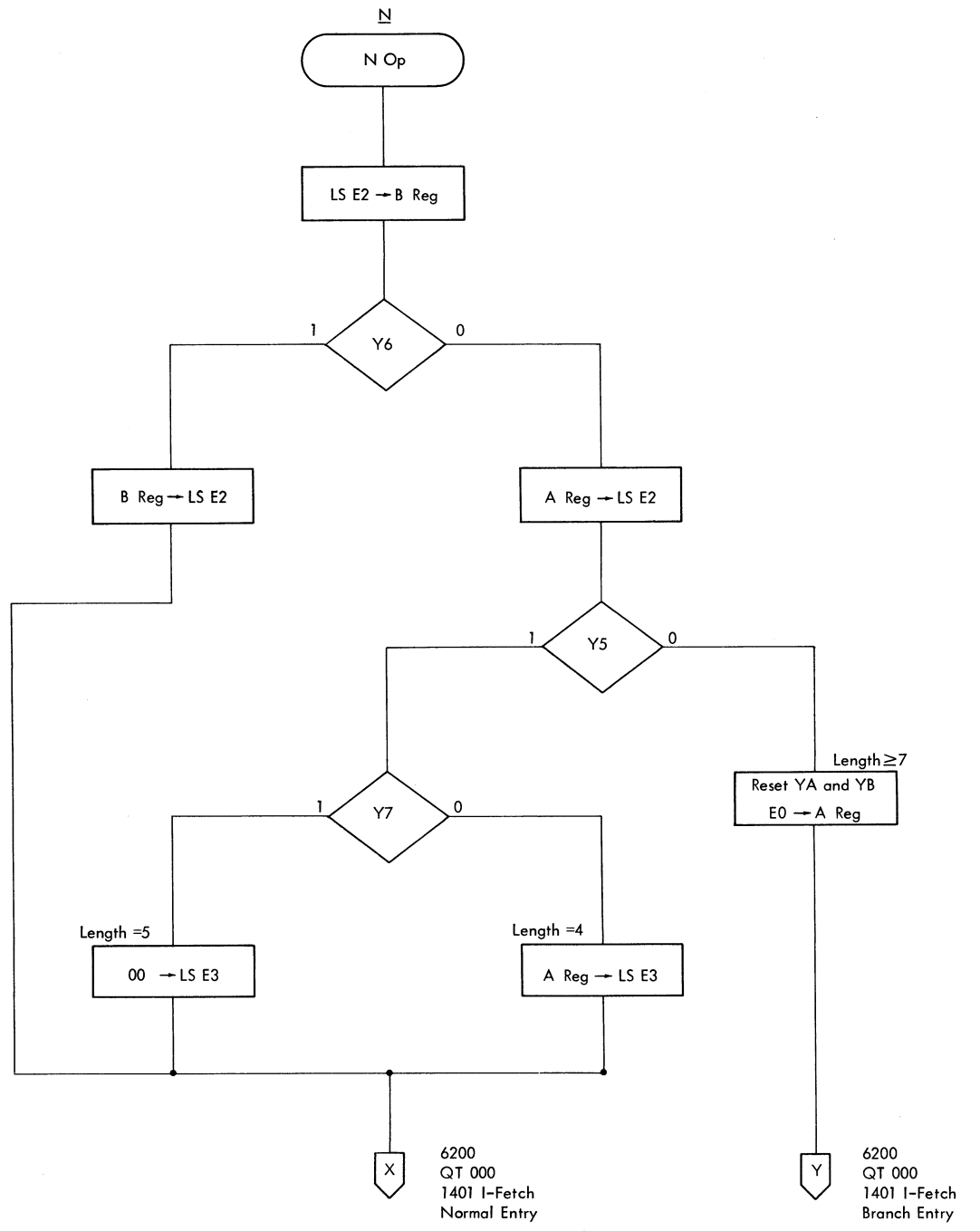
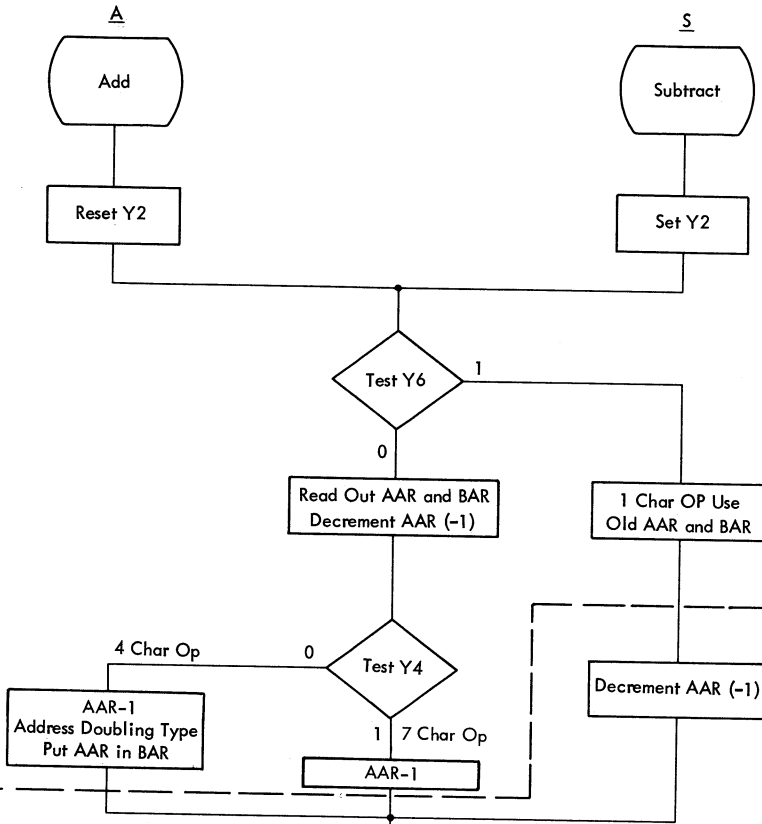


FIGURE 6201. 1401 N OPERATION

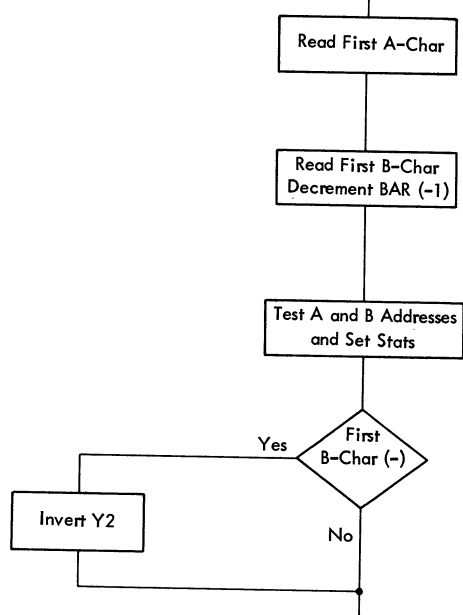
Stats	
Y	Means
4	7 Char Op
5	4 Char Op
6	1 Char Op

QT 117

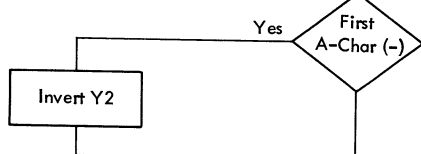


Address Test	
Y	Means
4	Address (A-B) = 1
5	B Address Odd
6	Both Addresses Odd
7	Address (A-B) Odd

QT 114

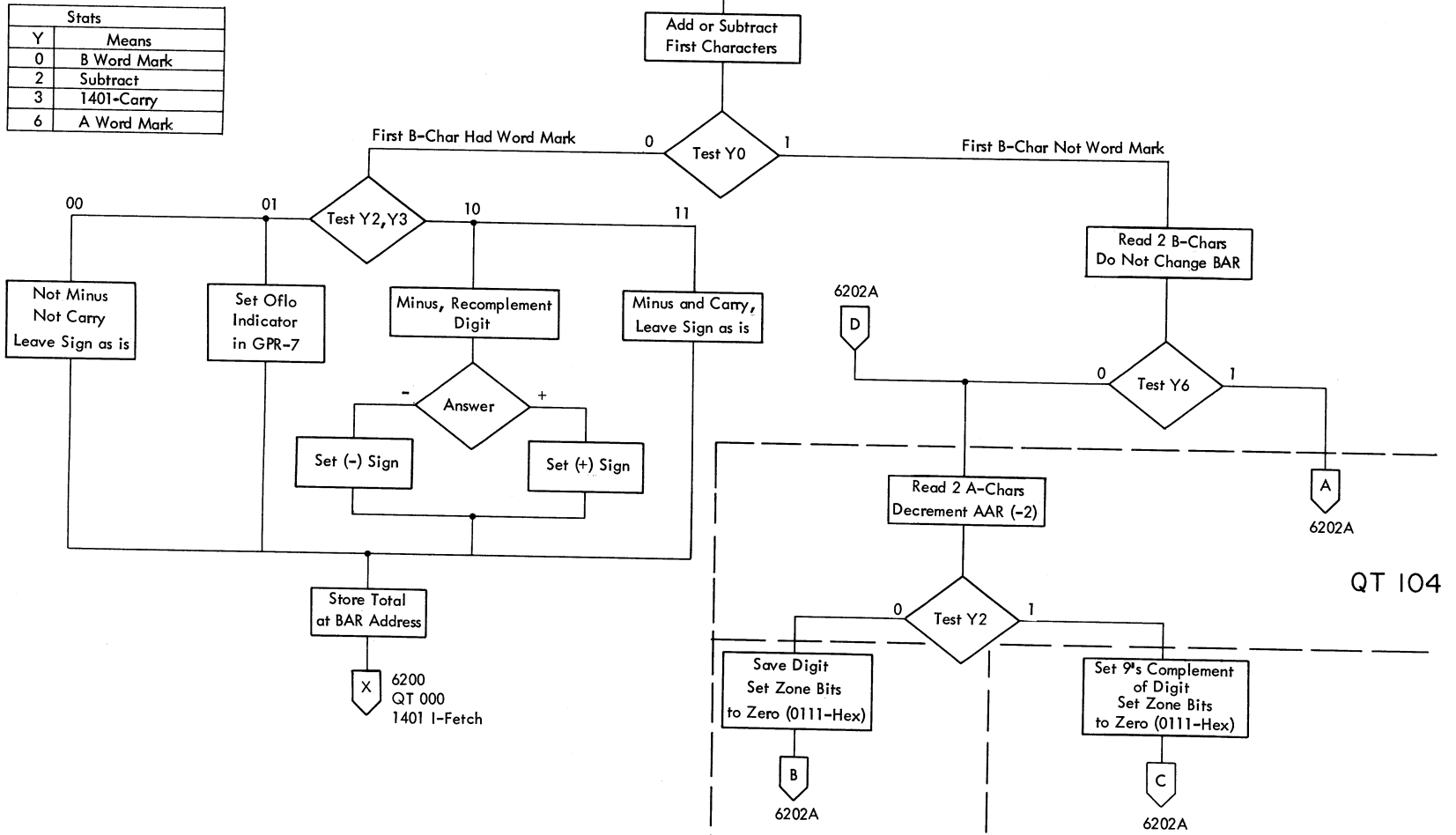


QT 115



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

QT 116



QT 104

QT 105

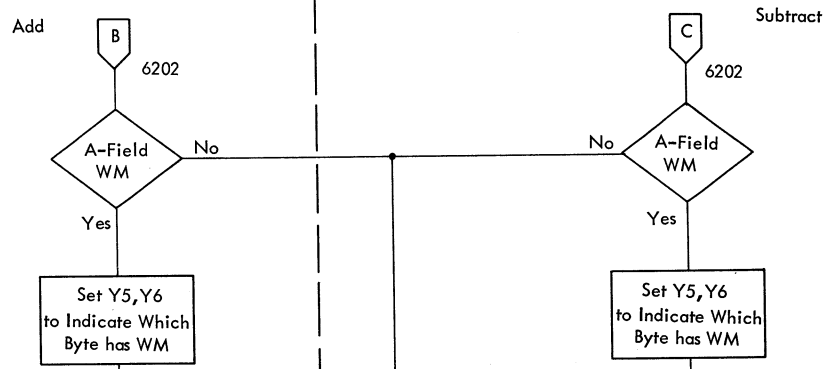
QT 106

FIGURE 6202. 1401 ADD AND SUBTRACT

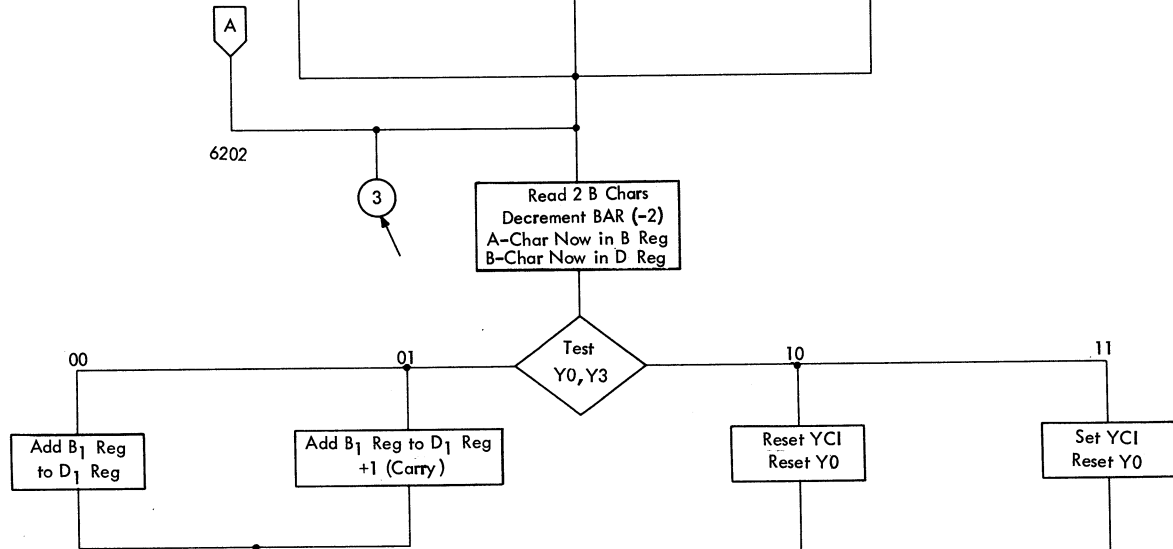
QT 105/107

Y	Means
0	B Address Odd
2	Subtract
3	1401 Carry
5	A <sub>1</sub> Word Mark
6	A Word Mark

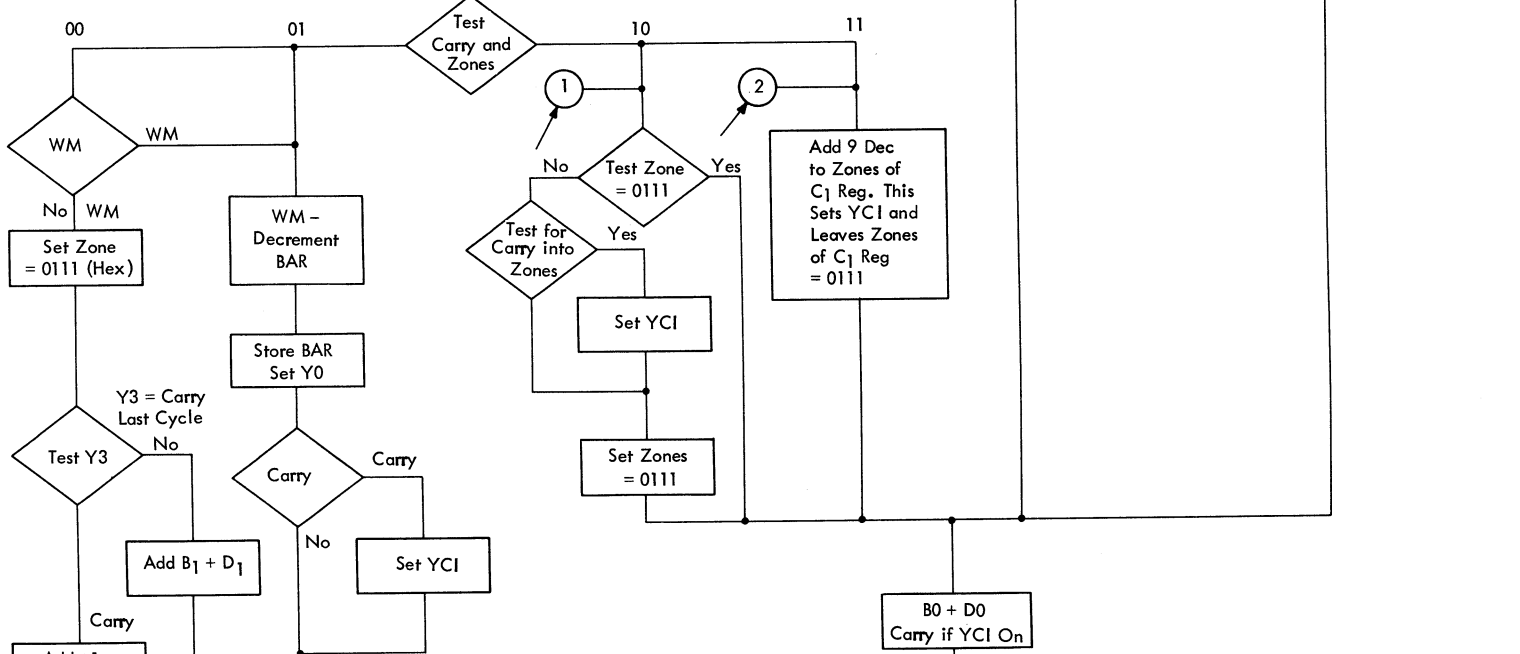
QT 106



QT 107



QT 108



QT 109

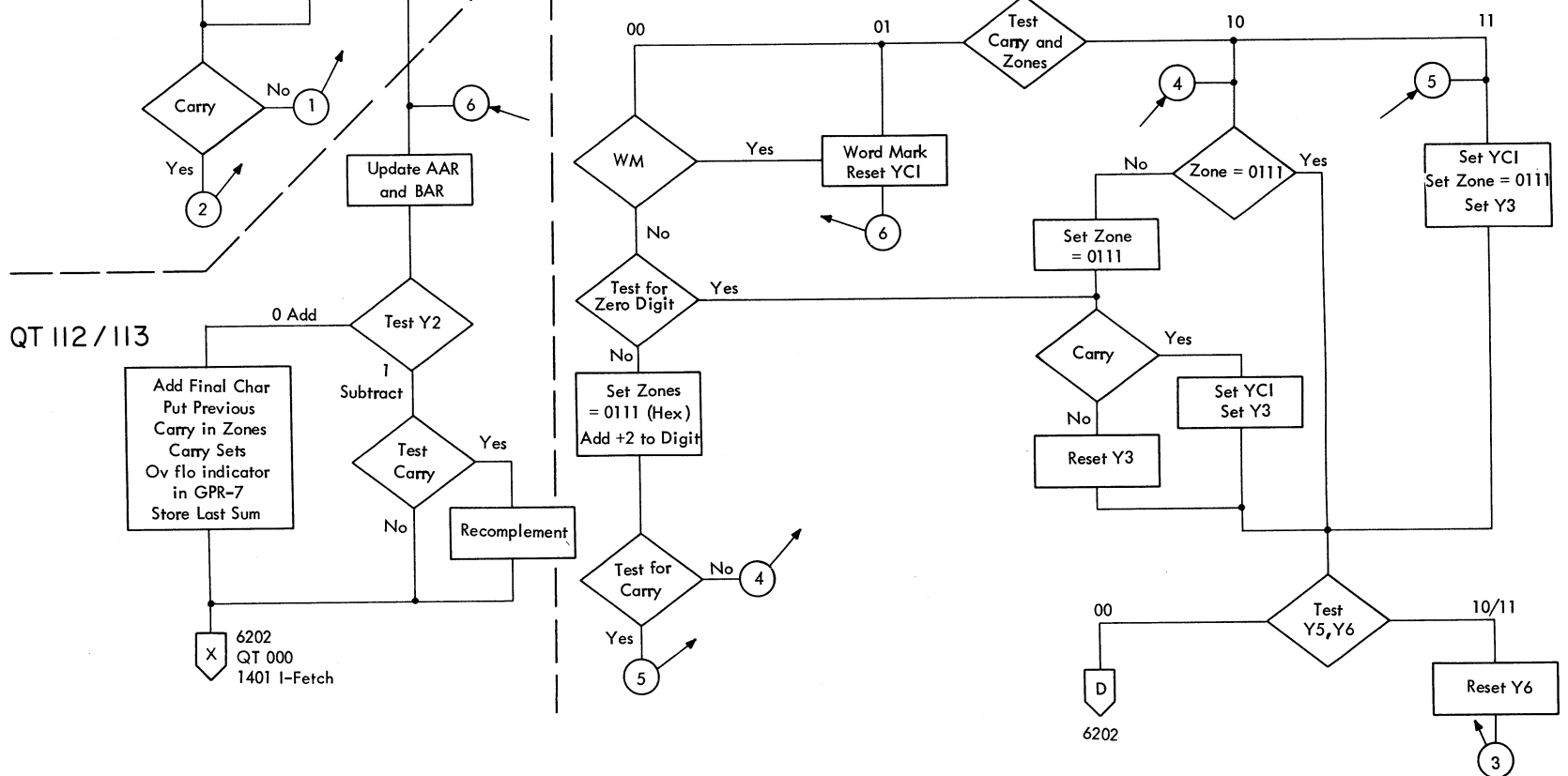
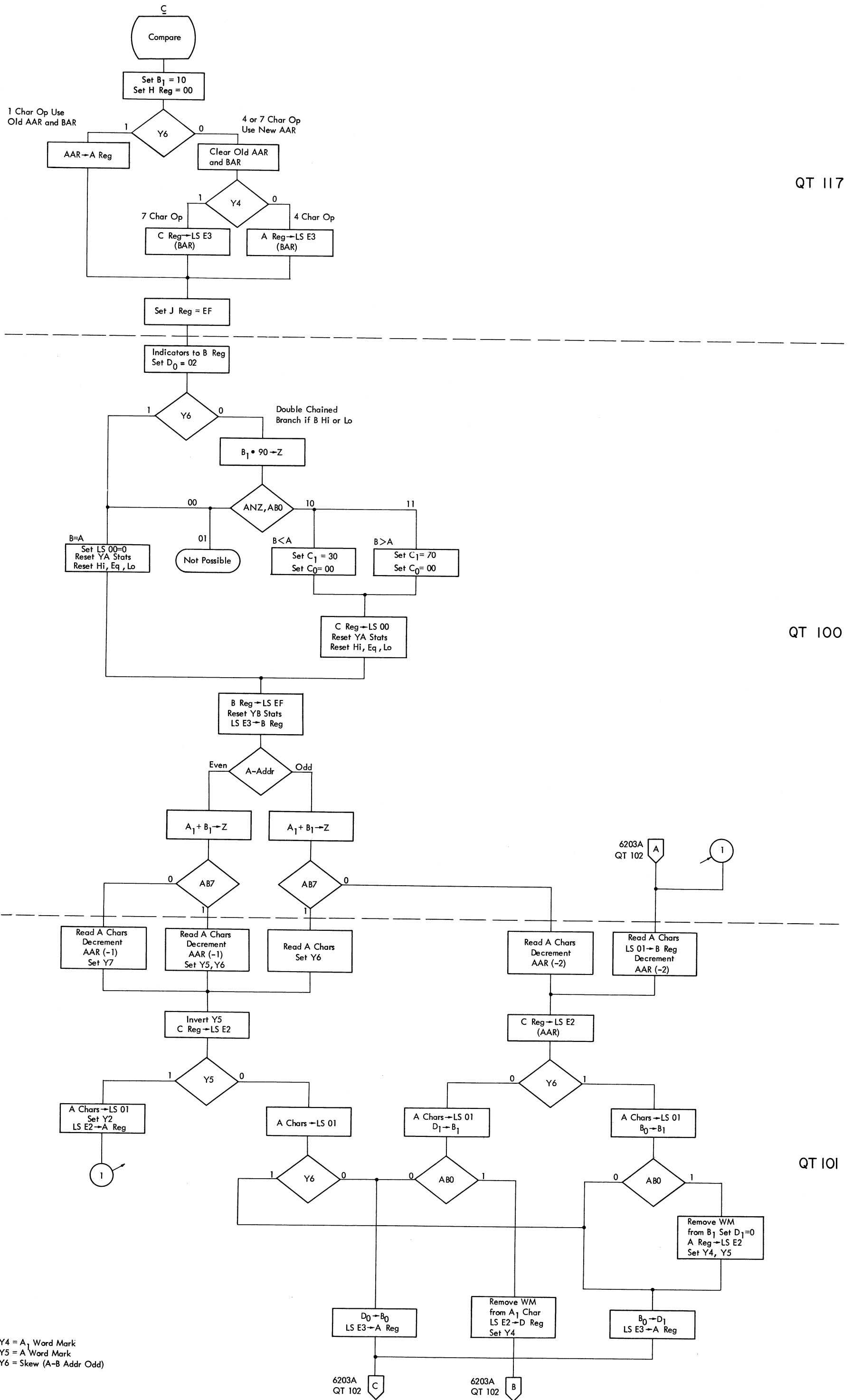


FIGURE 6202A. 1401 ADD AND SUBTRACT



QT 117

QT 100

QT 101

FIGURE 6203. 1401 COMPARE

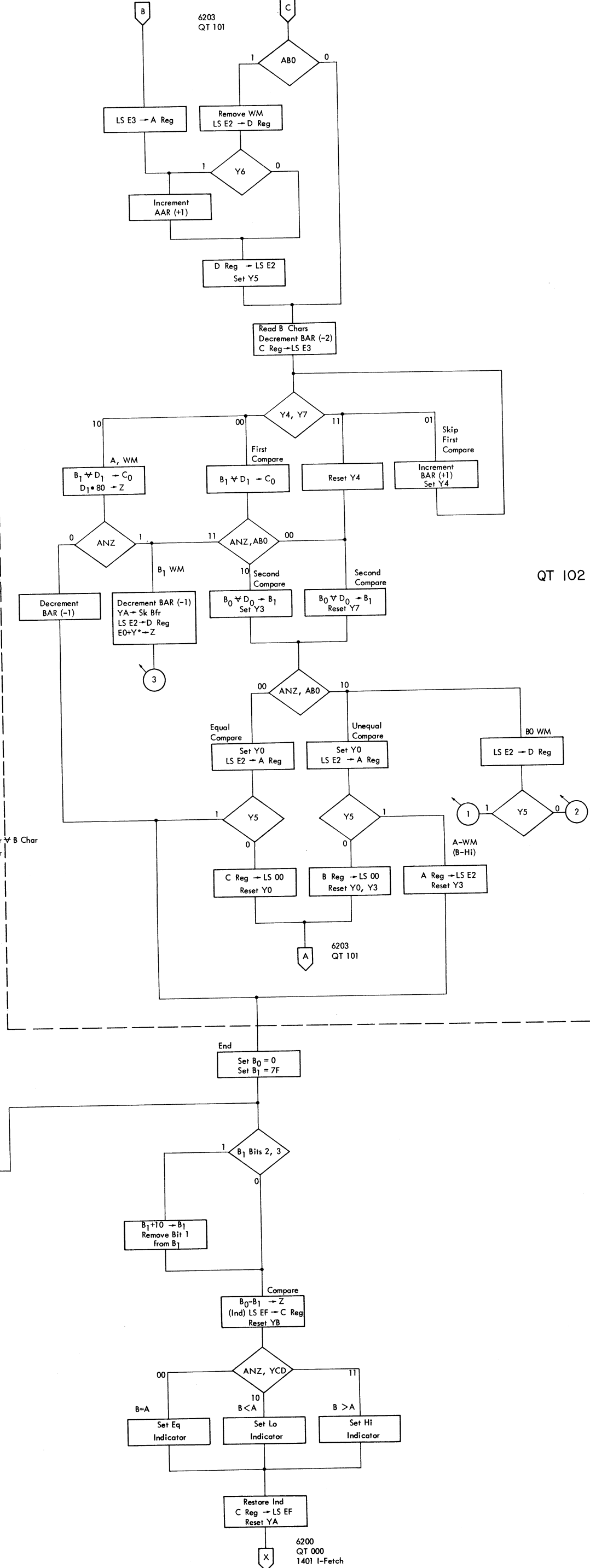
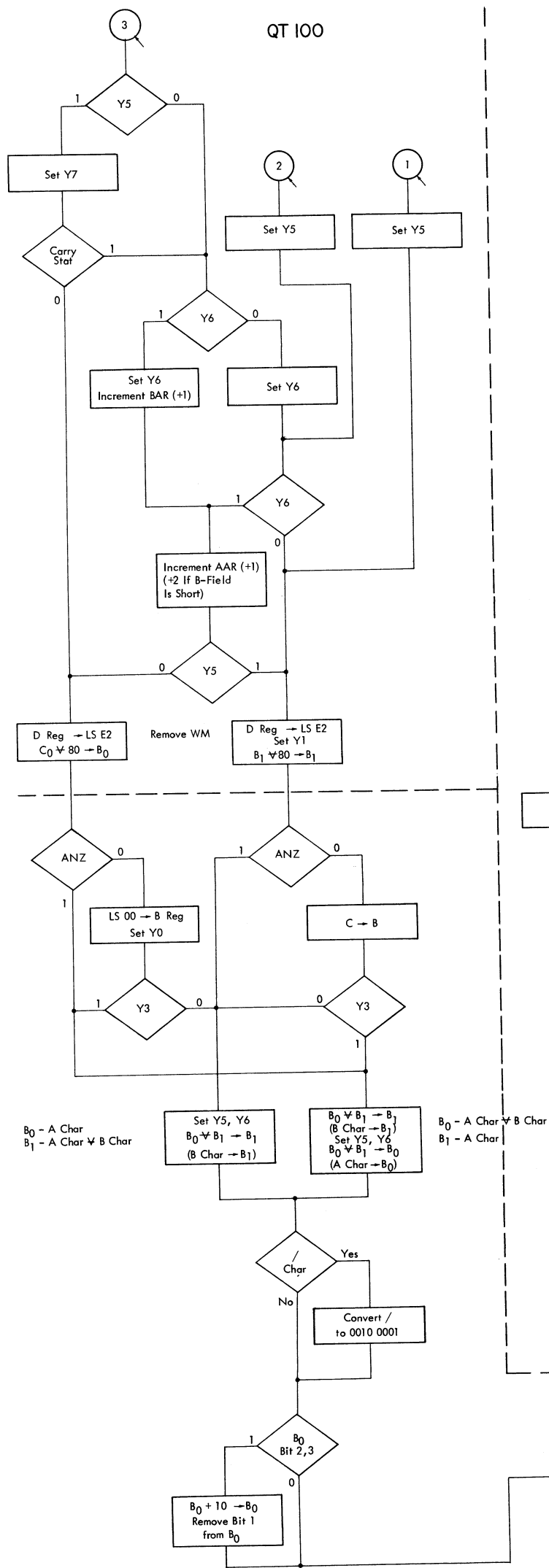
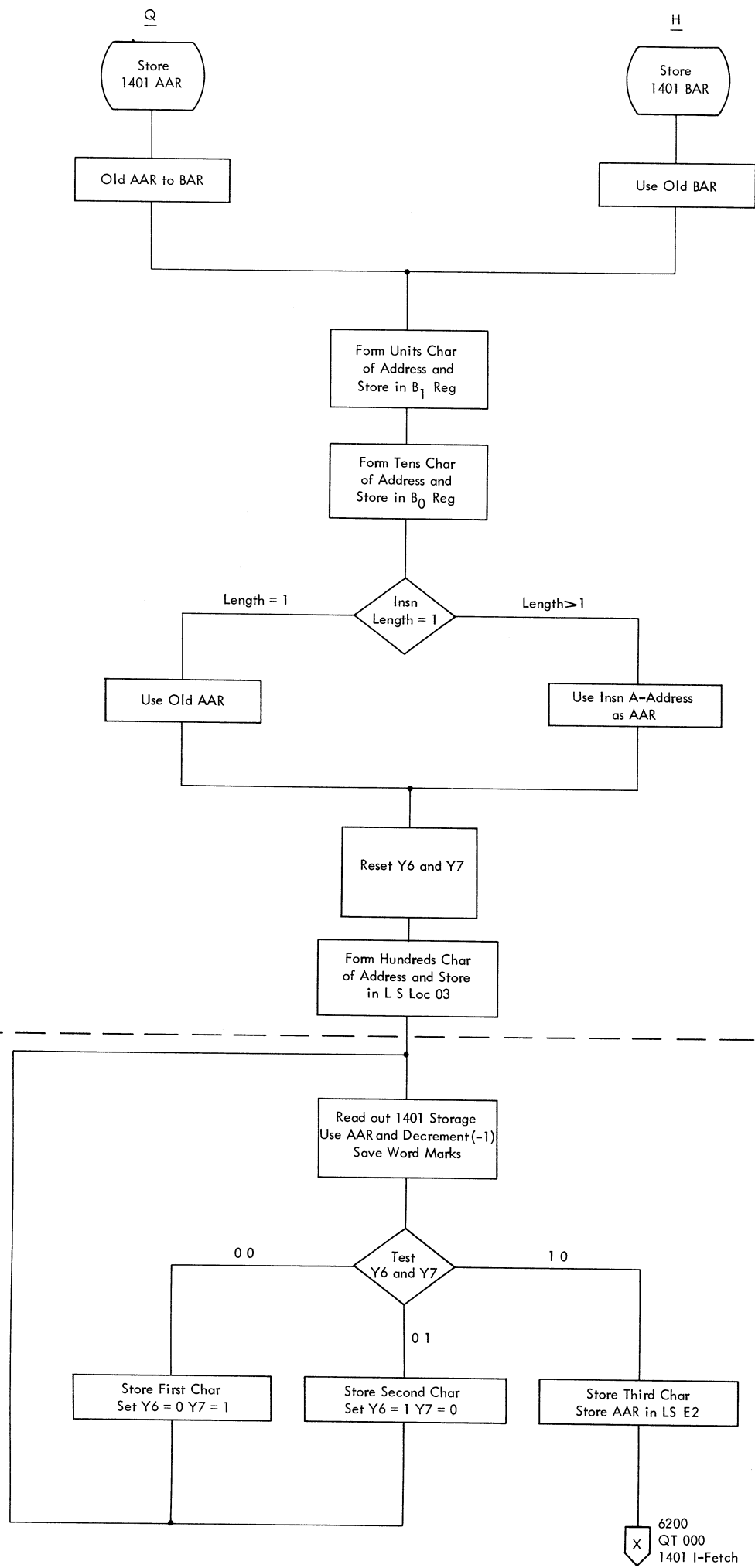


FIGURE 6203A. 1401 COMPARE



QT 206

FIGURE 6204. STORE 1401 AAR OR BAR

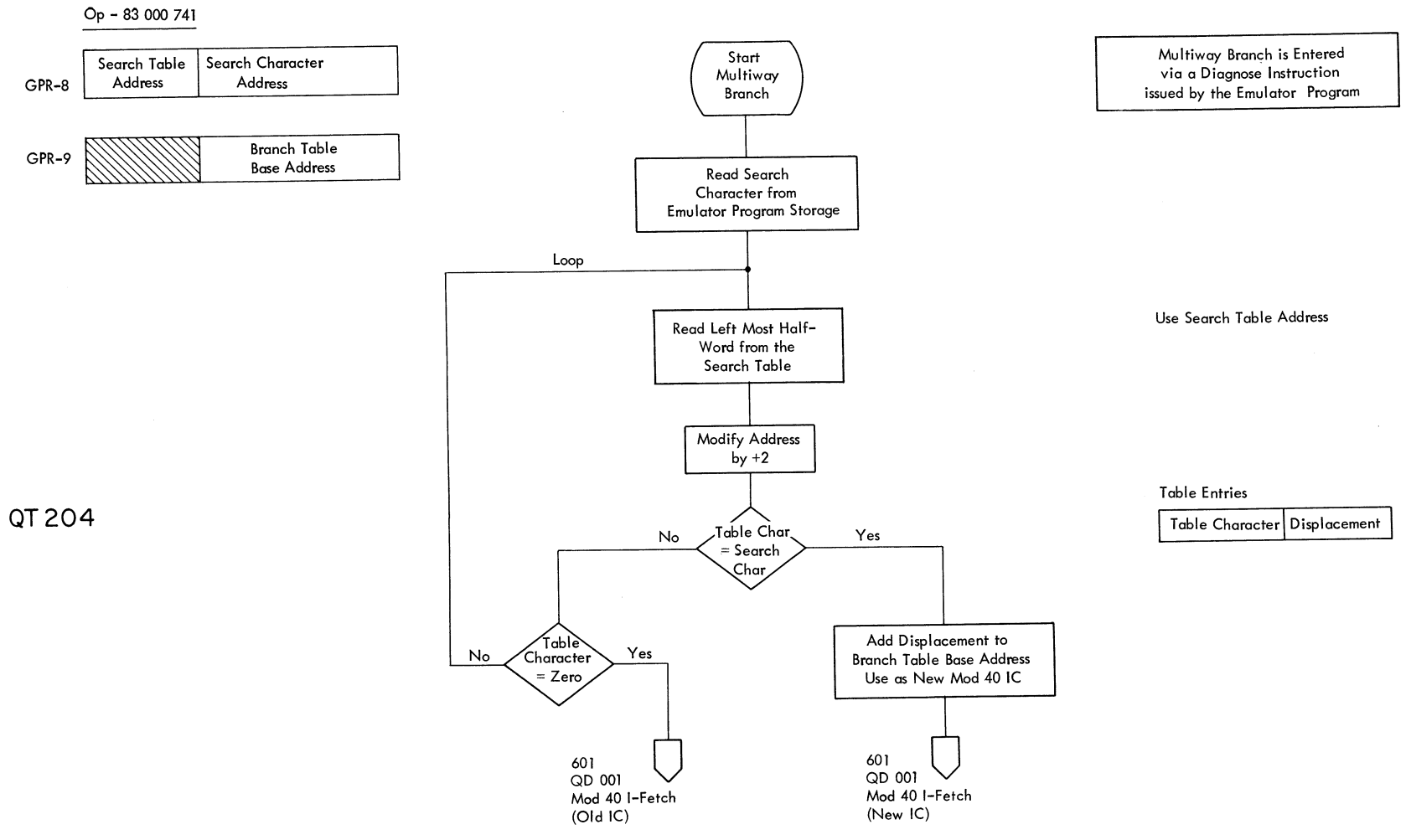


FIGURE 6205. MULTIWAY BRANCH

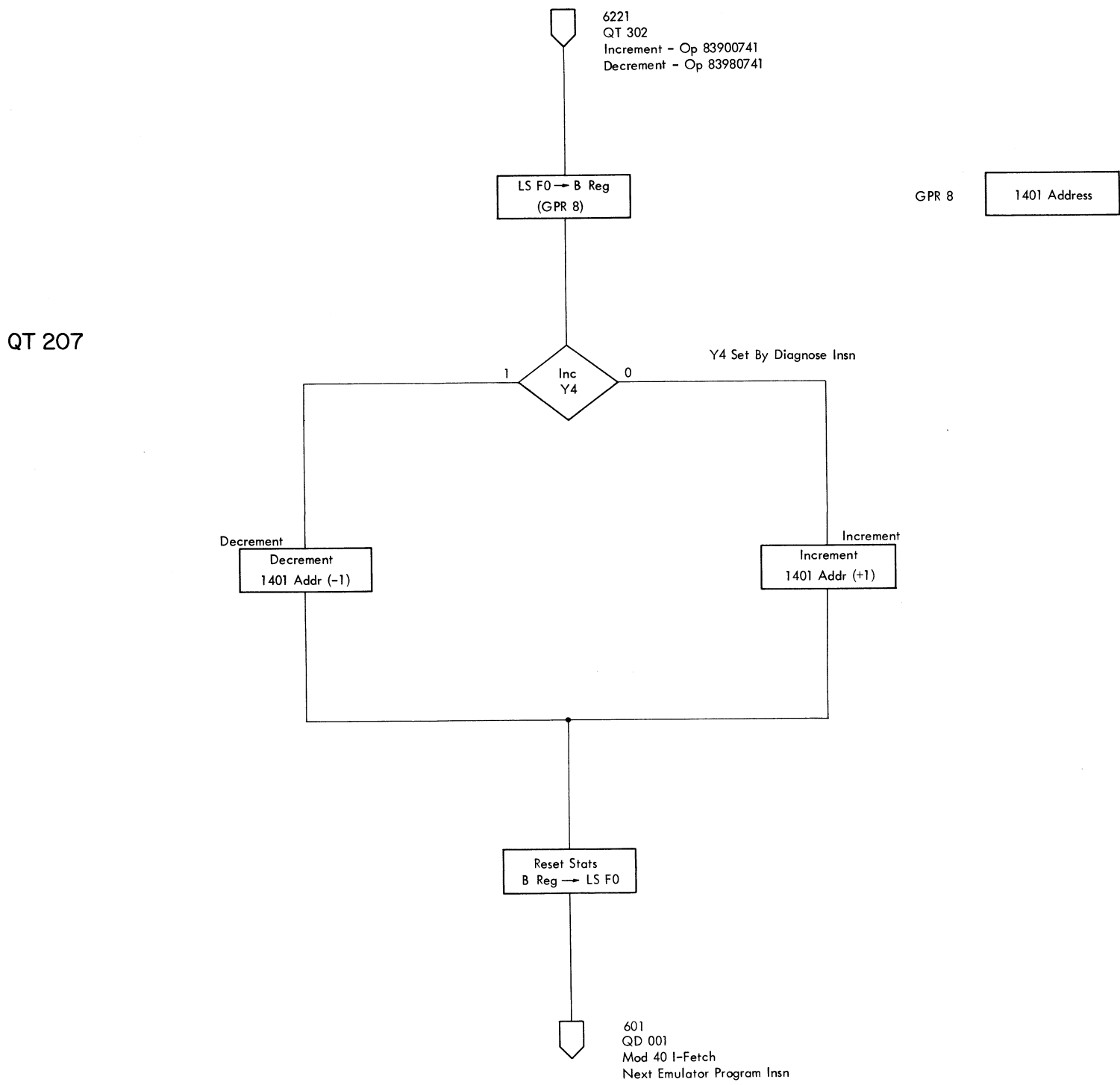


FIGURE 6206. 1401 INCREMENT - DECREMENT



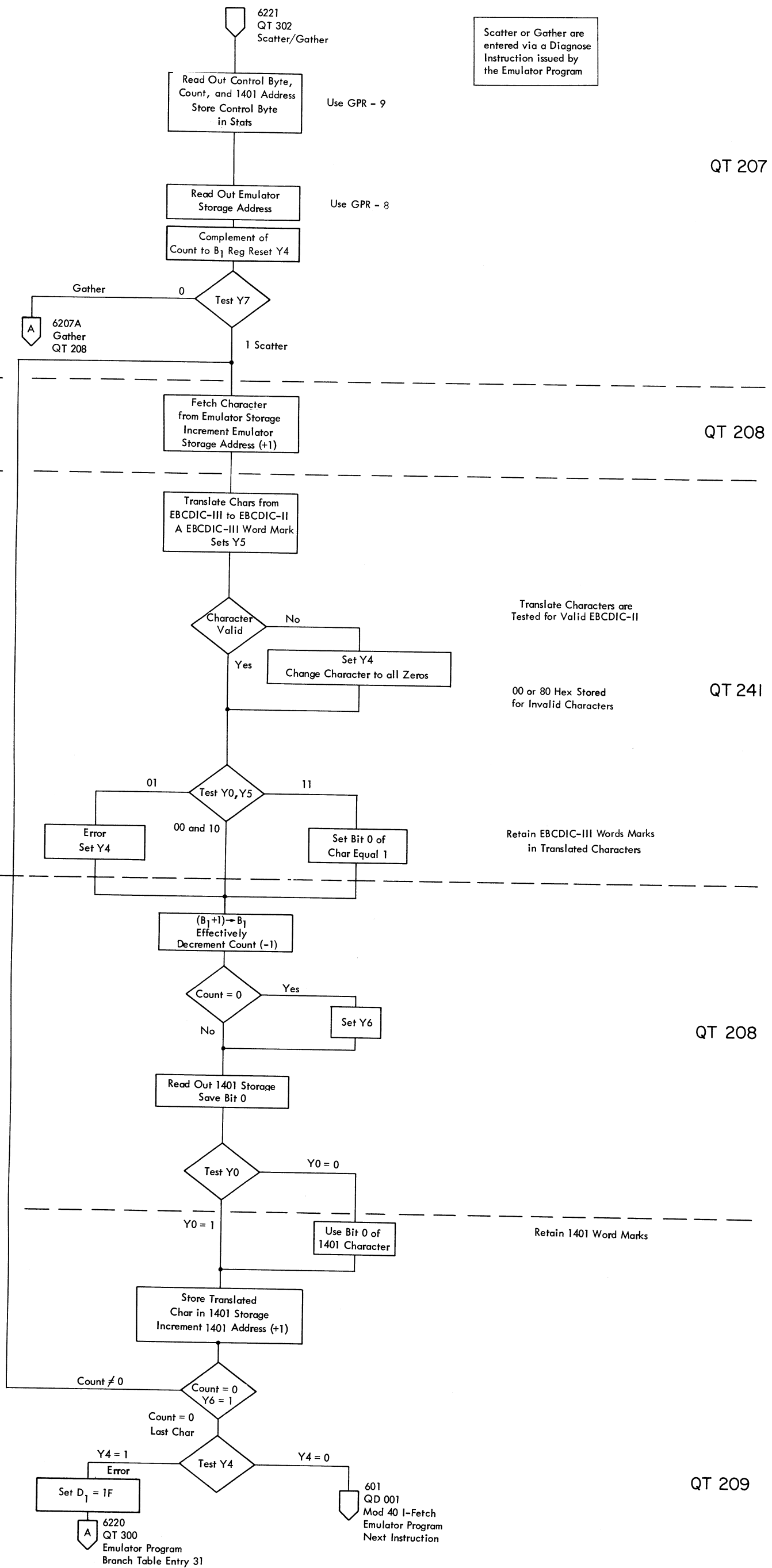
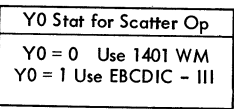
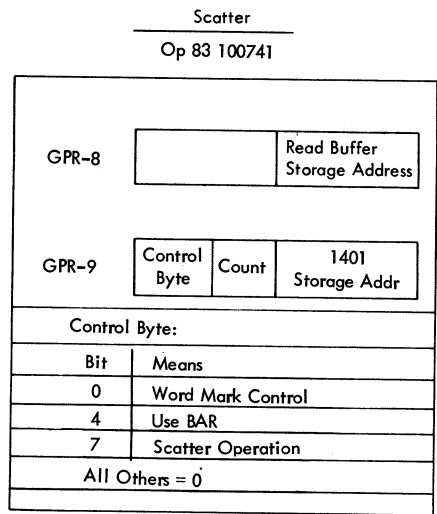
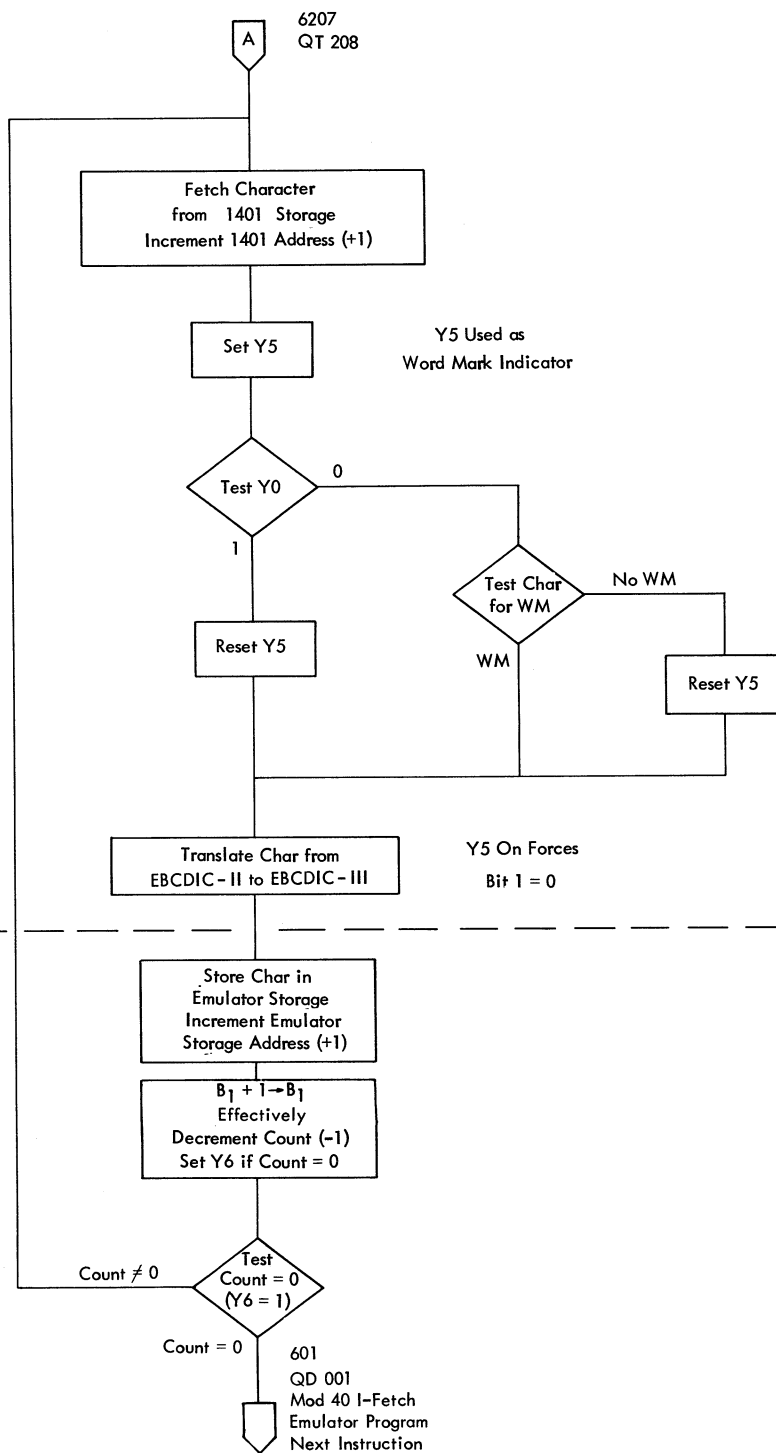
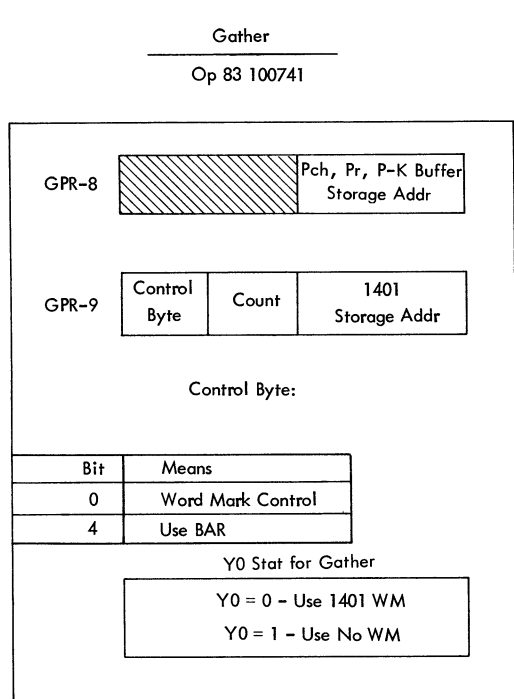


FIGURE 6207. SCATTER-GATHER



QT 208

QT 209

QT 208

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

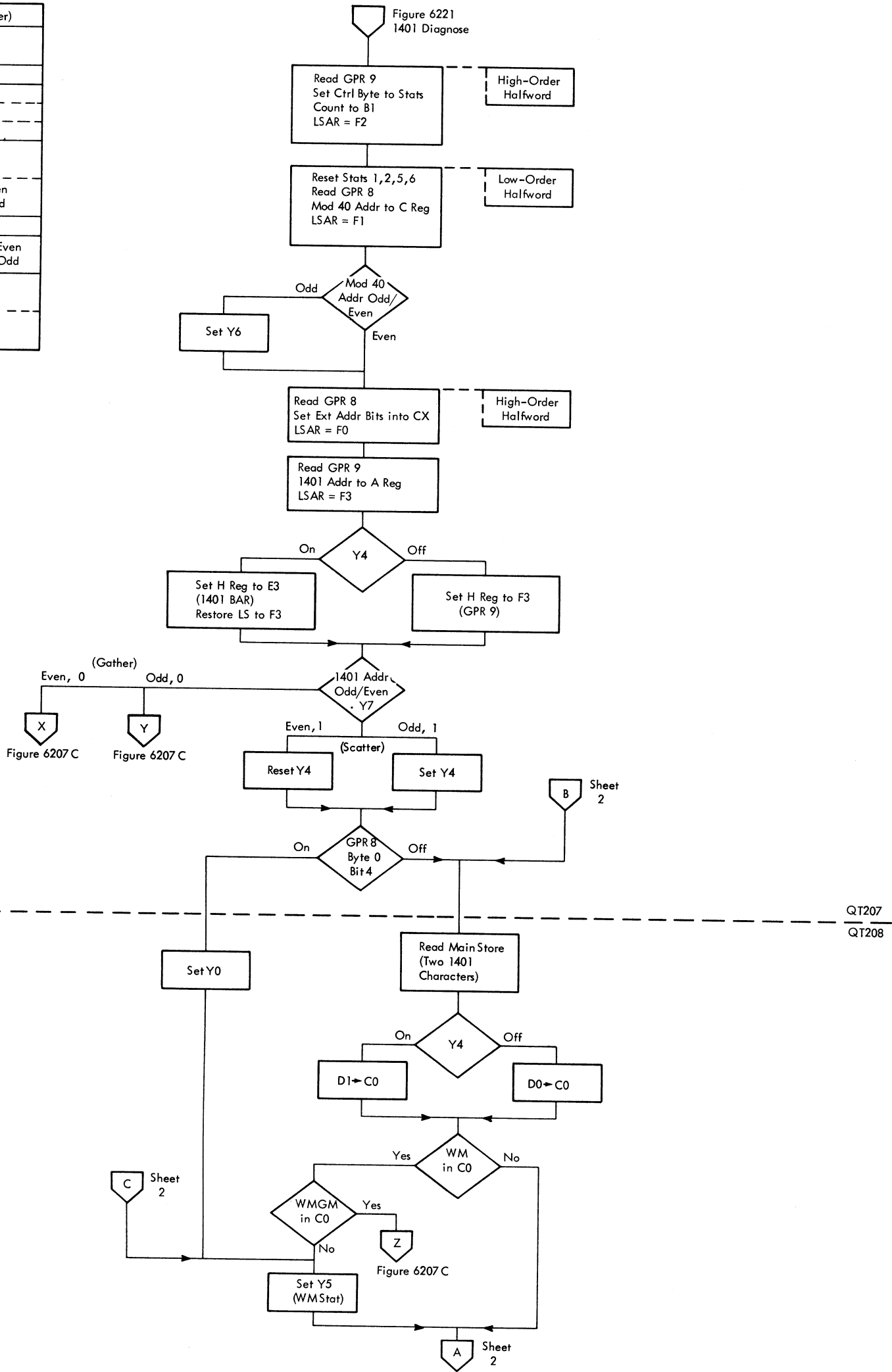


FIGURE 6207B. SCATTER (DOS COMPATIBILITY FEATURE) (PART 1 OF 2)

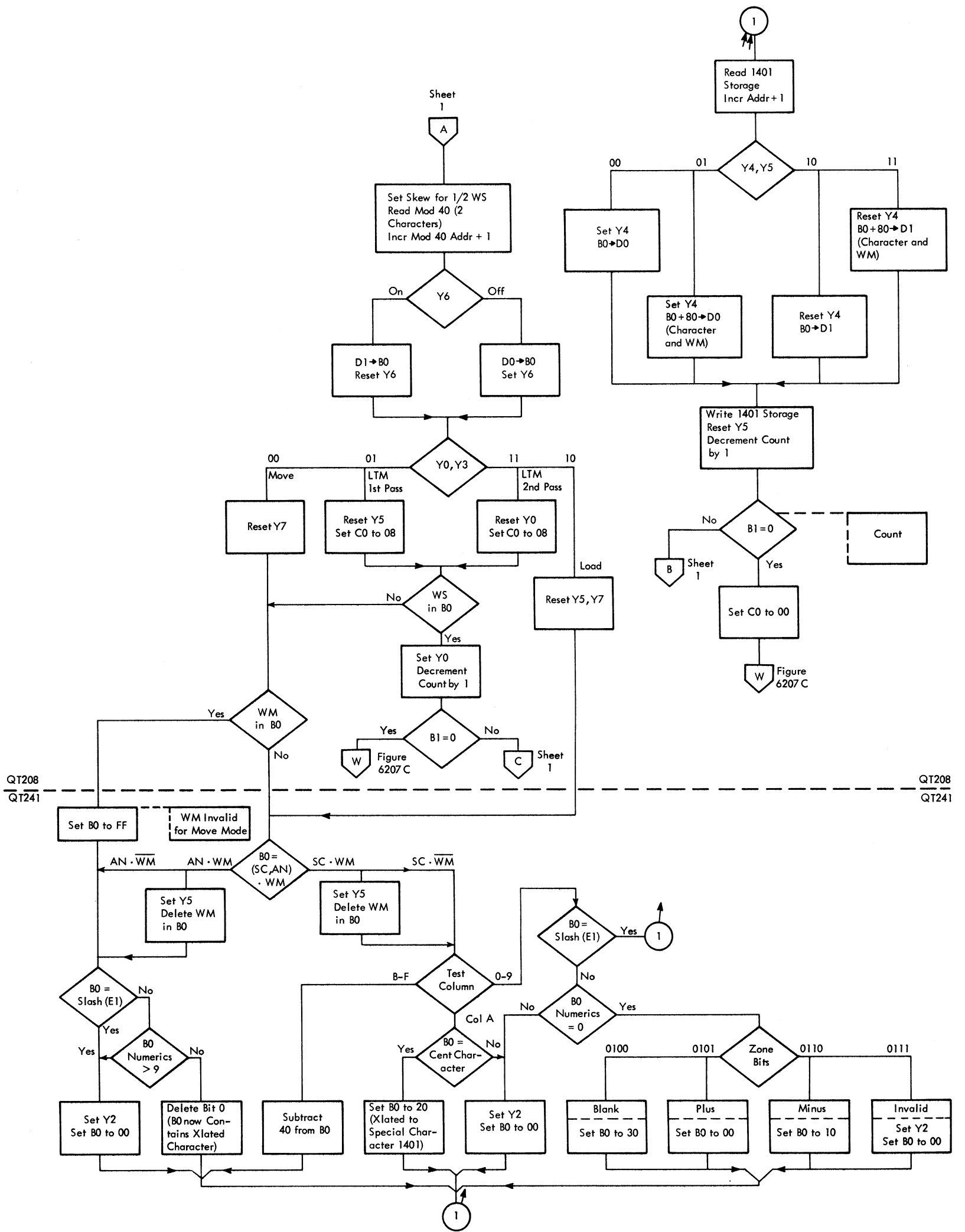


FIGURE 6207B. SCATTER (DOS COMPATIBILITY FEATURE) (PART 2 OF 2)

Stat	Meaning (Gather)
Y0	0 = Load 1 = Move
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

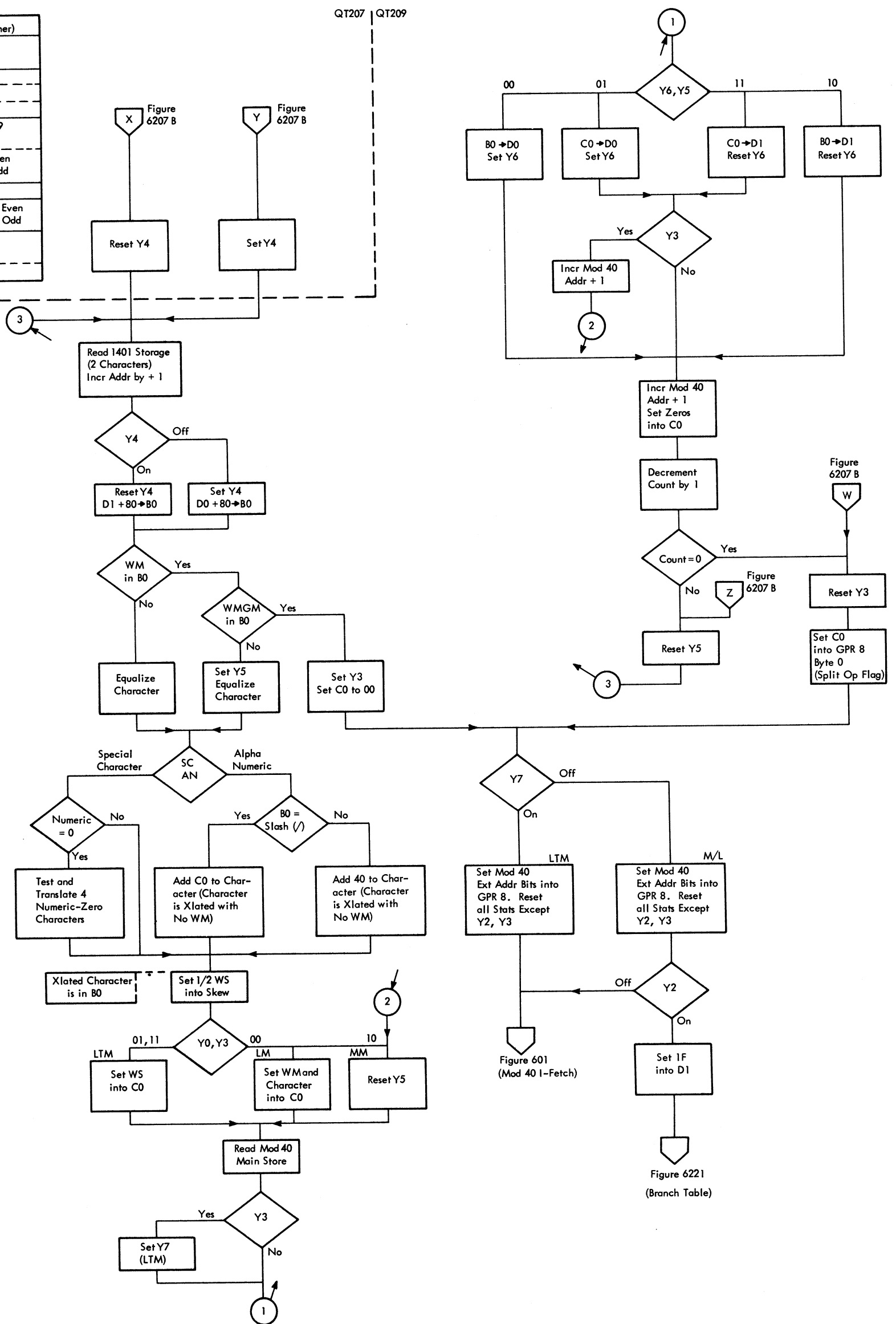
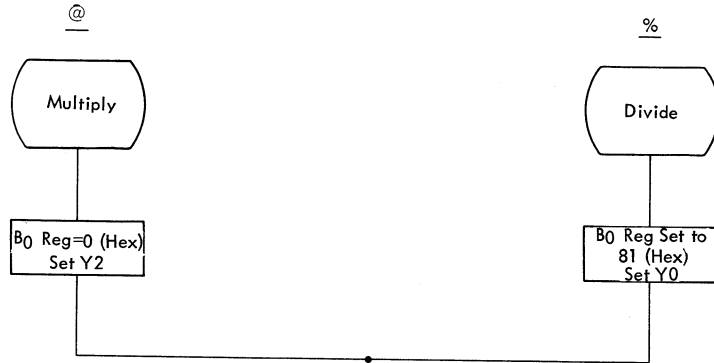
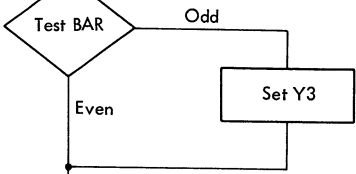
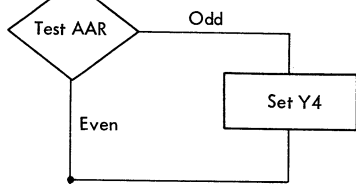


FIGURE 6207C. GATHER (DOS COMPATIBILITY FEATURE)

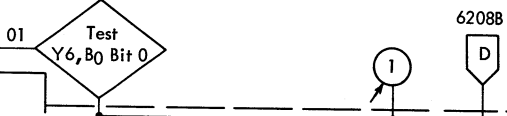
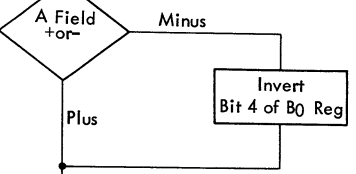
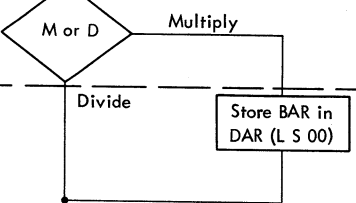
B0 Byte	
Bit	Means
0	Divide
2	Remainder Sign
3	A-WM
4	Minus Sign
5	RC MQ
6	B-WM (Mult)
7	IZT Zero (Div)
Stats	
0	Carry
2	First Scan
3	BAR Odd
4	AAR Odd
5	Comp Add
6	X or MQ
7	X or Y



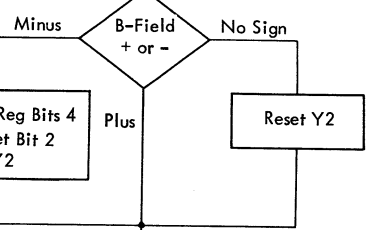
Put AAR in CAR  
Put BAR in DAR  
Read A Field Sign Digit



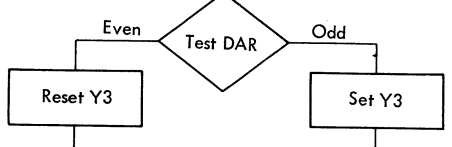
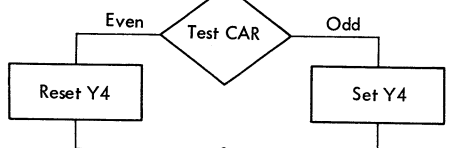
Invert Y7



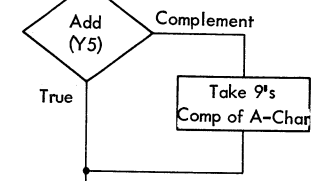
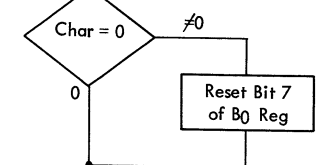
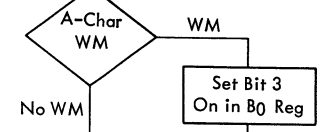
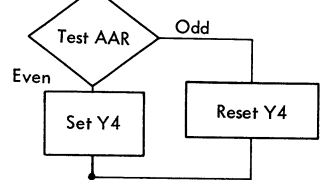
Invert Y5  
Put DAR in A Reg  
Read Out B-Field Sign Digit  
Read Out A-Character



Set Y0 and Y5  
CAR to AAR  
DAR to BAR  
Reset Y6 and Y7



Read out A-Character  
Decrement AAR (-1)



6208A

QT 210

QT 212

QT 213

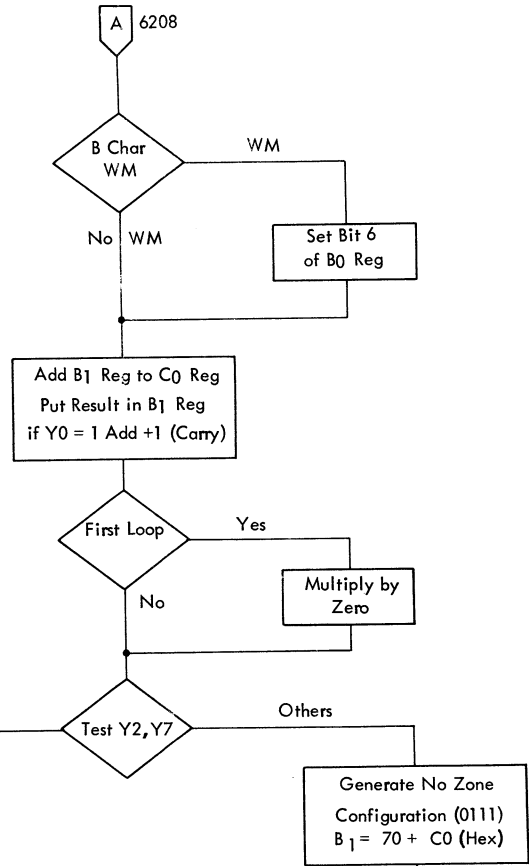
QT 215

QT 214

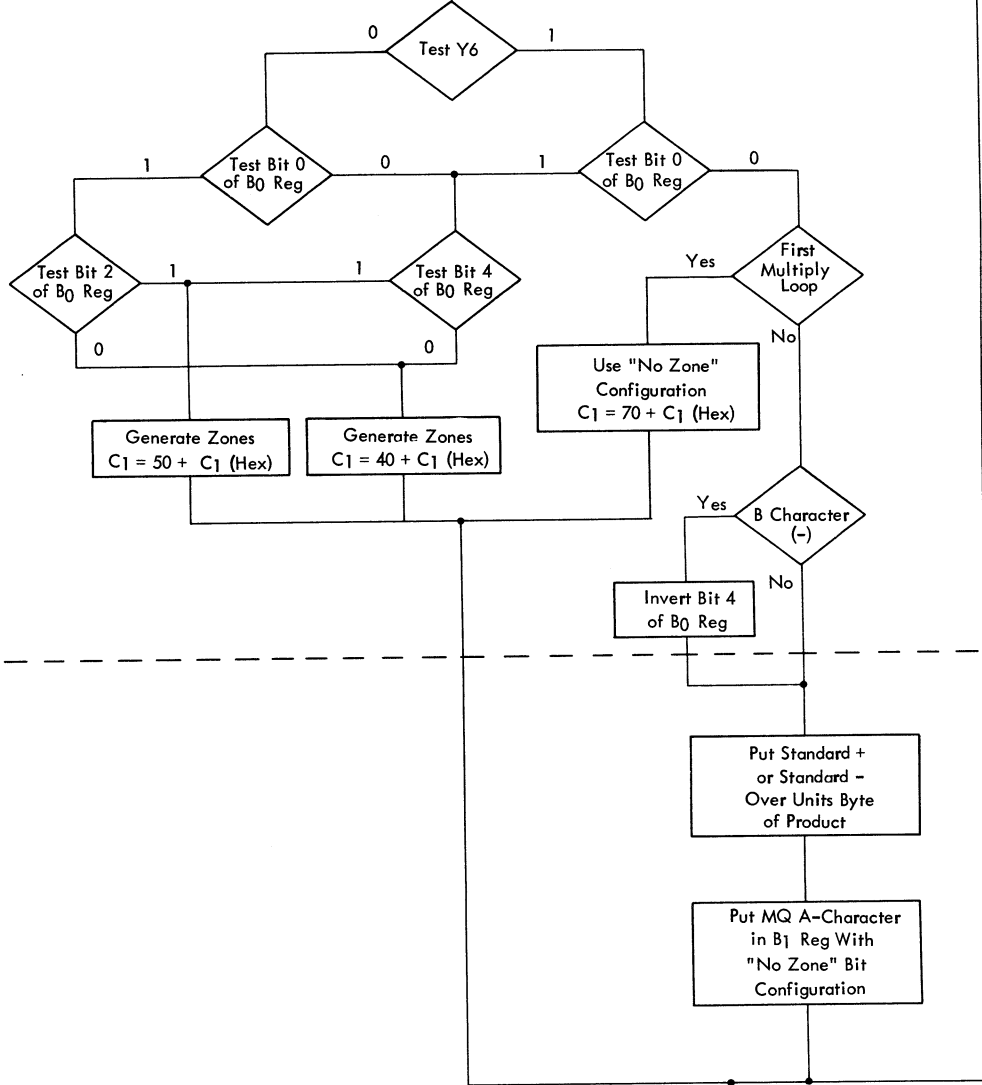
FIGURE 6208. 1401 MULTIPLY AND DIVIDE

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

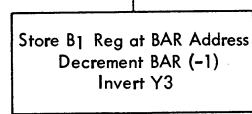
QT 214



QT 212



QT 210



QT 211

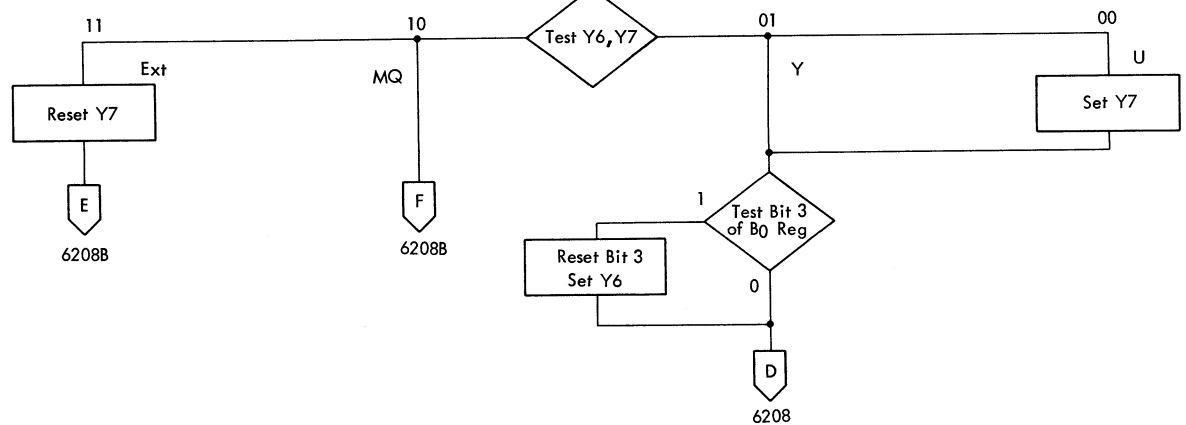
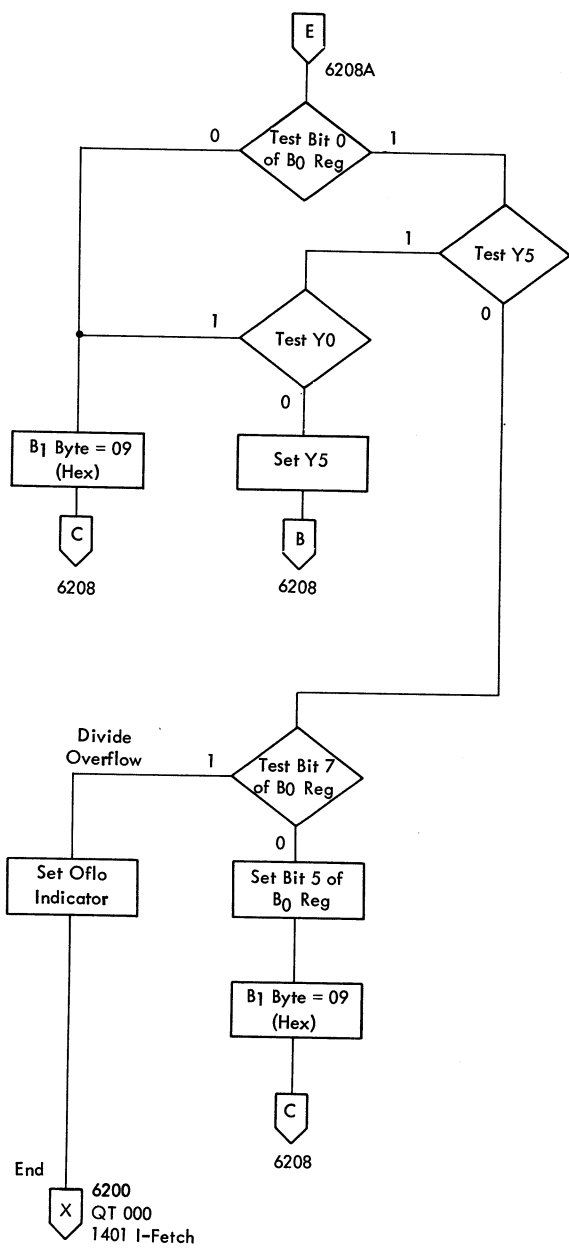
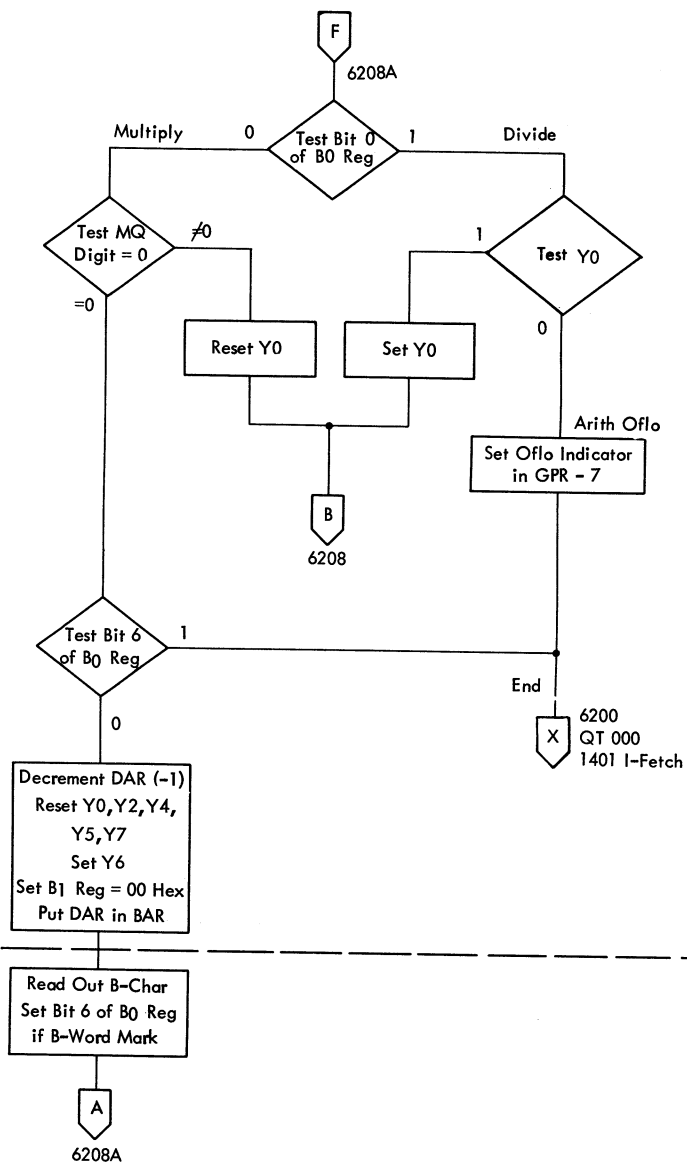


FIGURE 6208A. 1401 MULTIPLY AND DIVIDE

QT 211



QT 215



QT 214

B0 Byte		Y Stats	
Bit	Means	Y	Stats
0	Divide	0	Carry
2	Remainder Sign	2	First Scan
3	A WM	3	BAR Odd
4	Minus Sign	4	AAR Odd
5	RC MQ	5	Comp Add
6	B WM (Multiply)	6	X or MQ
7	IZT Zero (Divide)	7	X or Y

FIGURE 6208B. 1401 MULTIPLY AND DIVIDE



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

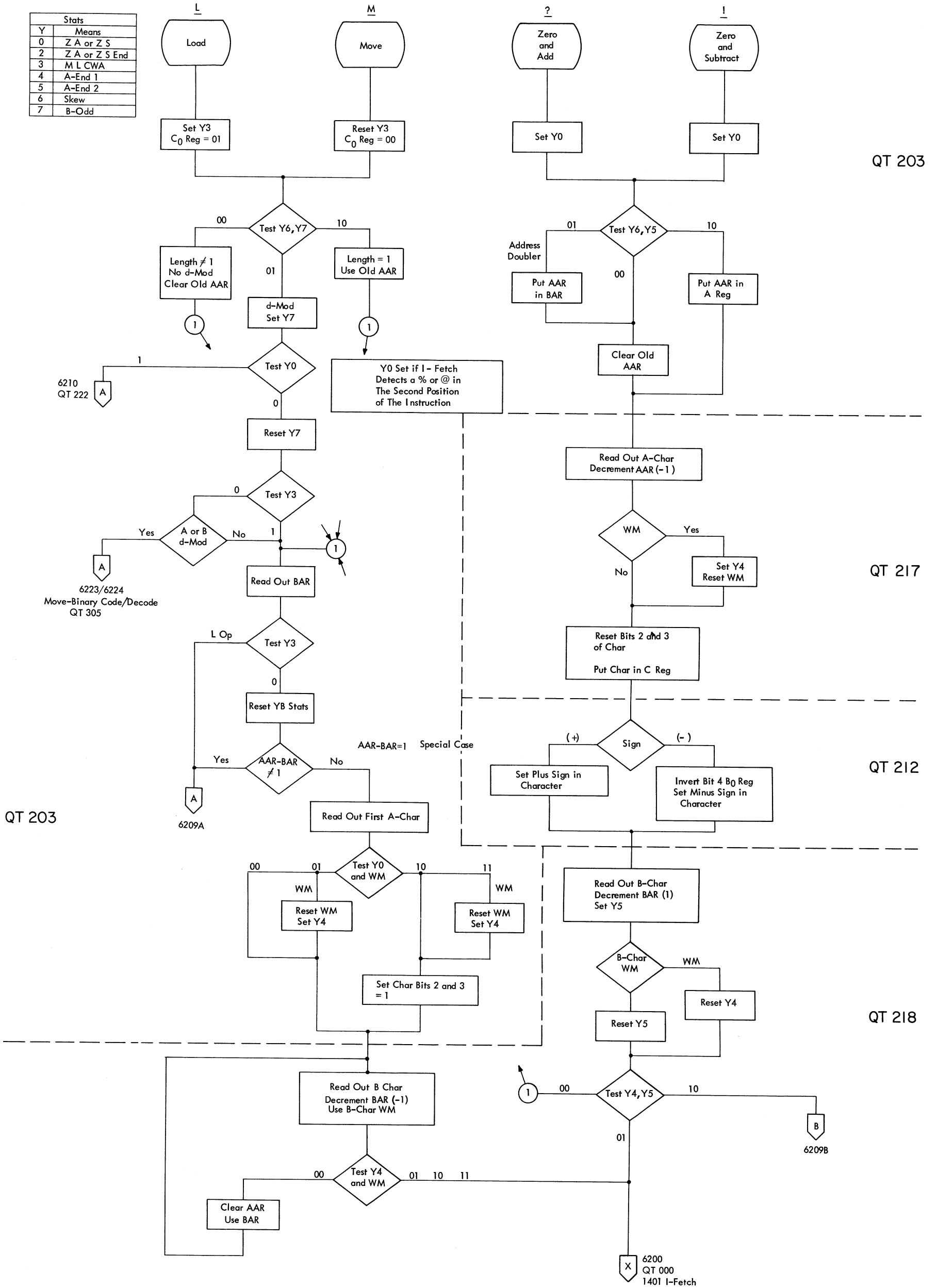


FIGURE 6209. 1401 MOVE, LOAD, ZERO AND ADD, ZERO AND SUBTRACT

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

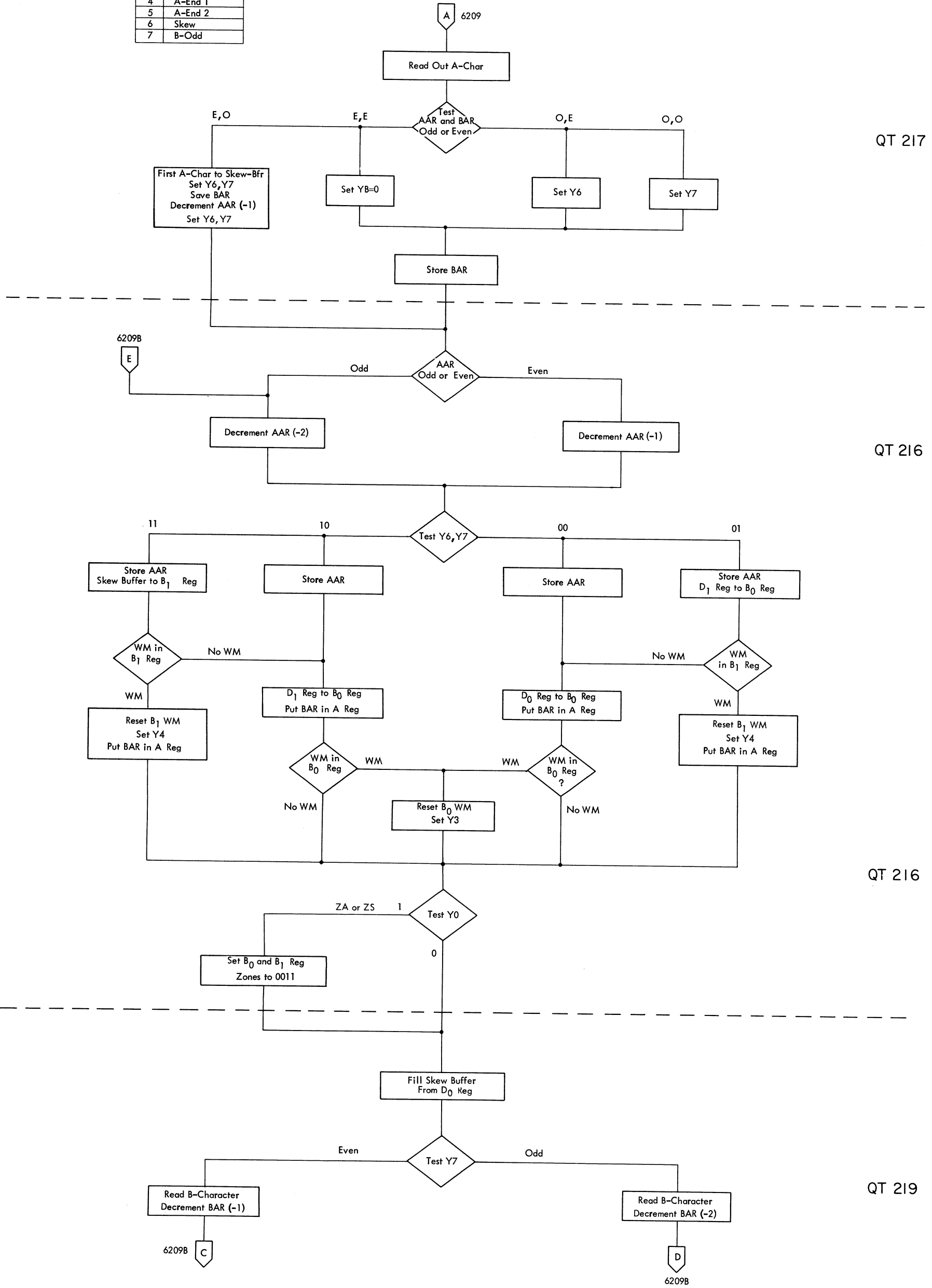


FIGURE 6209A. 1401 MOVE, LOAD, ZERO AND ADD, ZERO AND SUBTRACT

QT 219

QT 219

QT 220

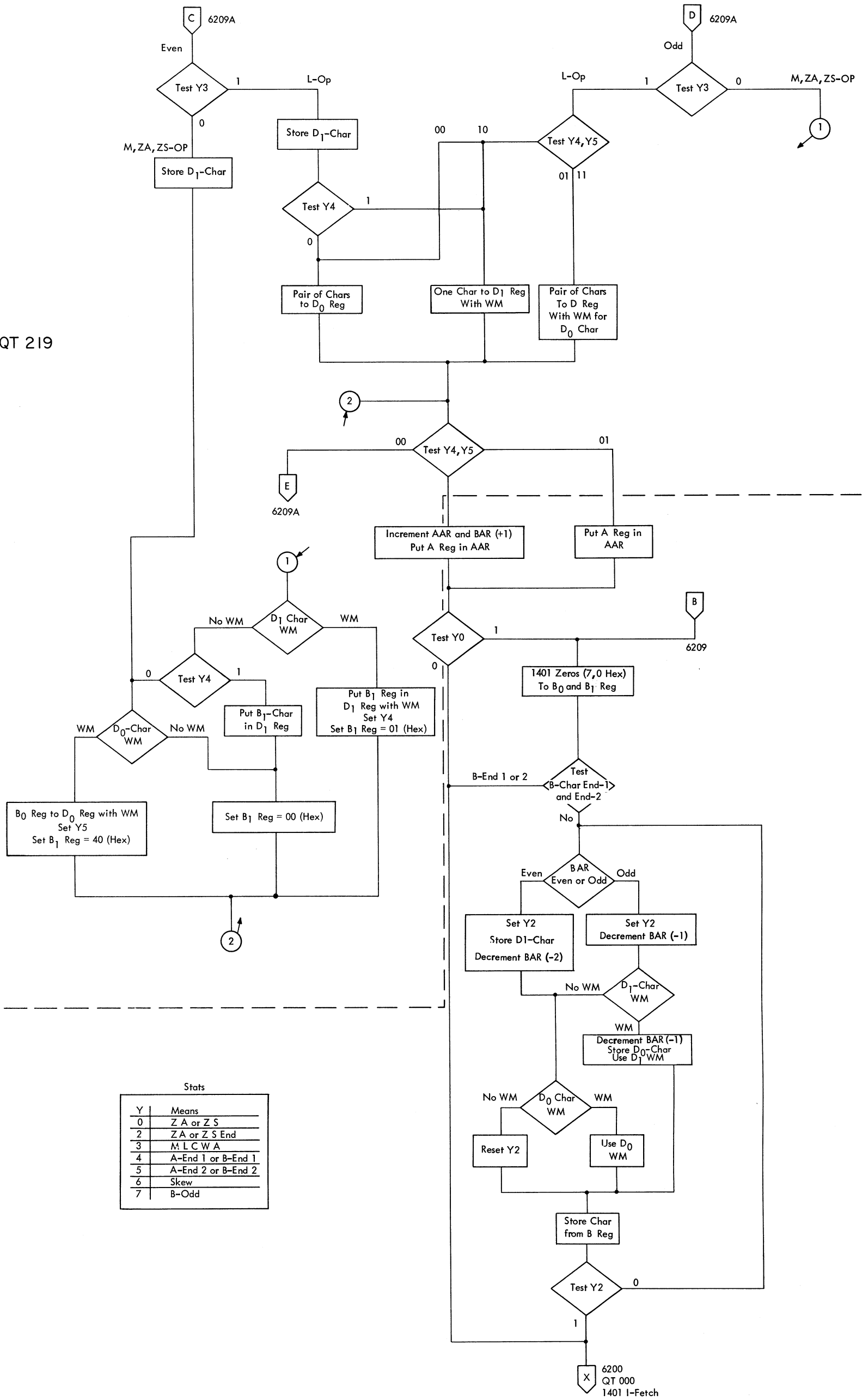
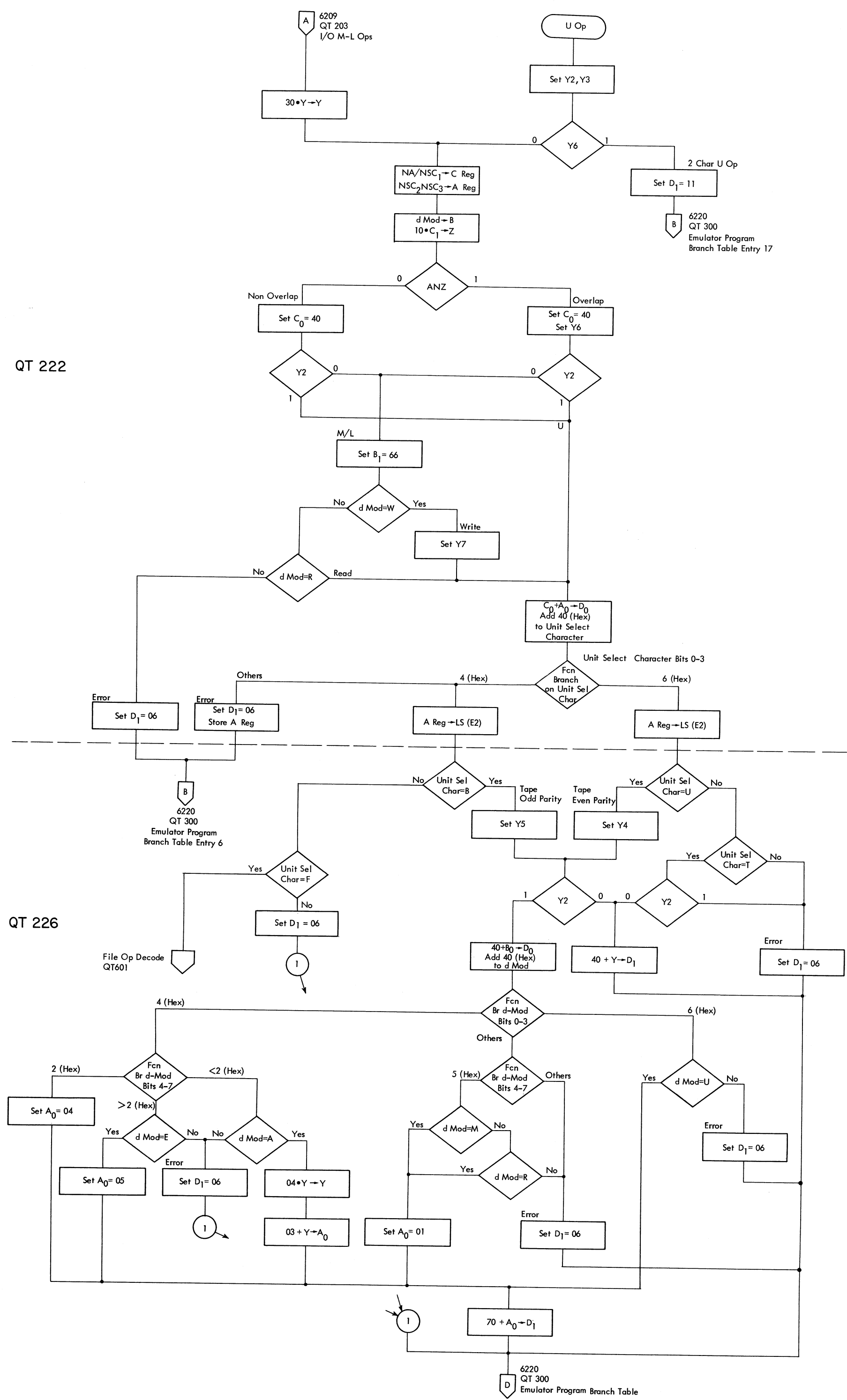


FIGURE 6209B. 1401 MOVE, LOAD, ZERO AND ADD, ZERO AND SUBTRACT



QT 222

QT 226

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

QT 223

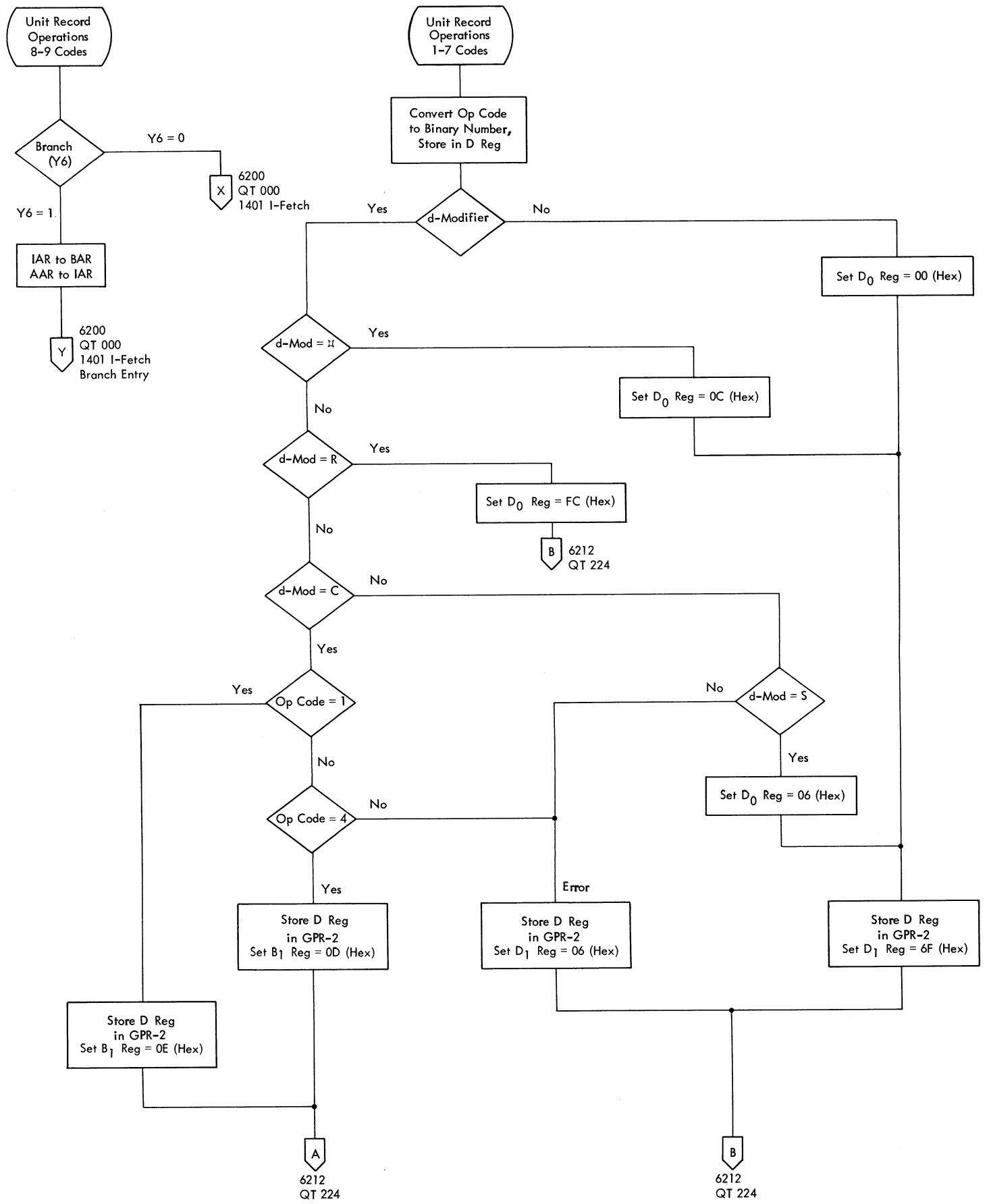
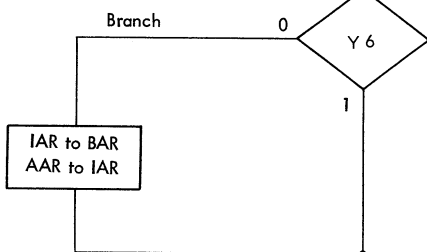
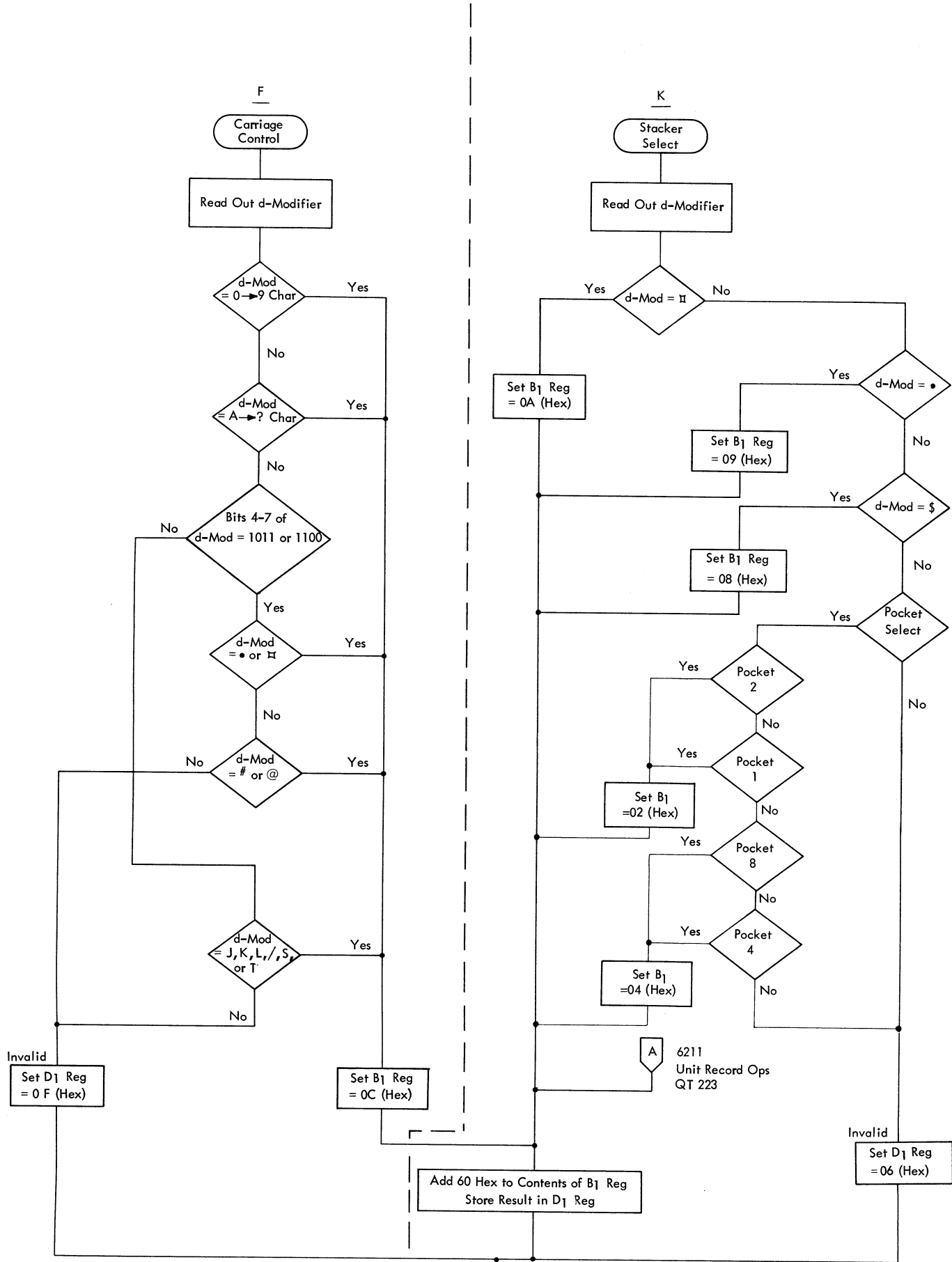


FIGURE 6211. 1401 UNIT RECORD OPERATIONS

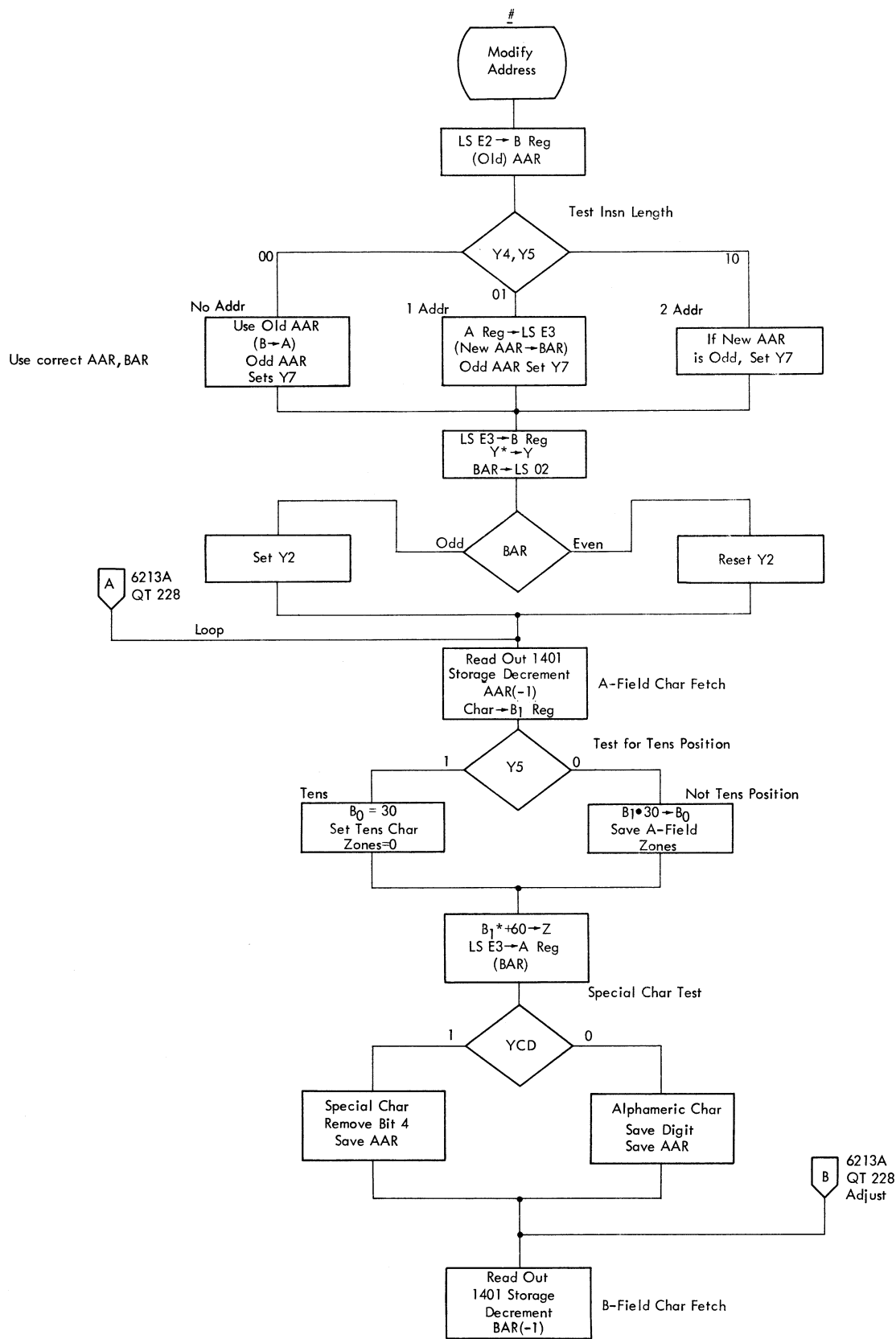
QT 225

QT 224



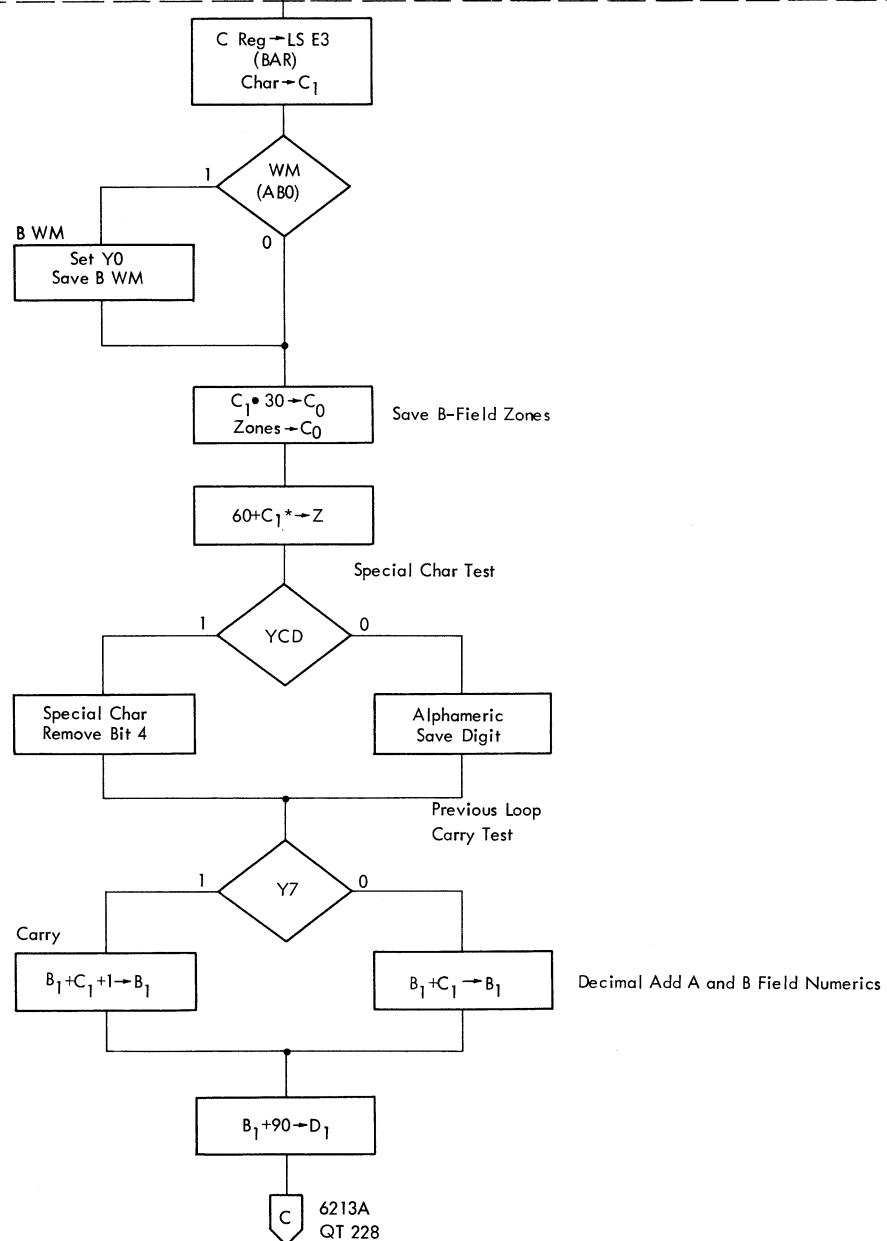
6220  
 QT 300  
 Emulator Program  
 Branch Table

FIGURE 6212. 1401 CARRIAGE CONTROL AND STACKER SELECT



QT 227

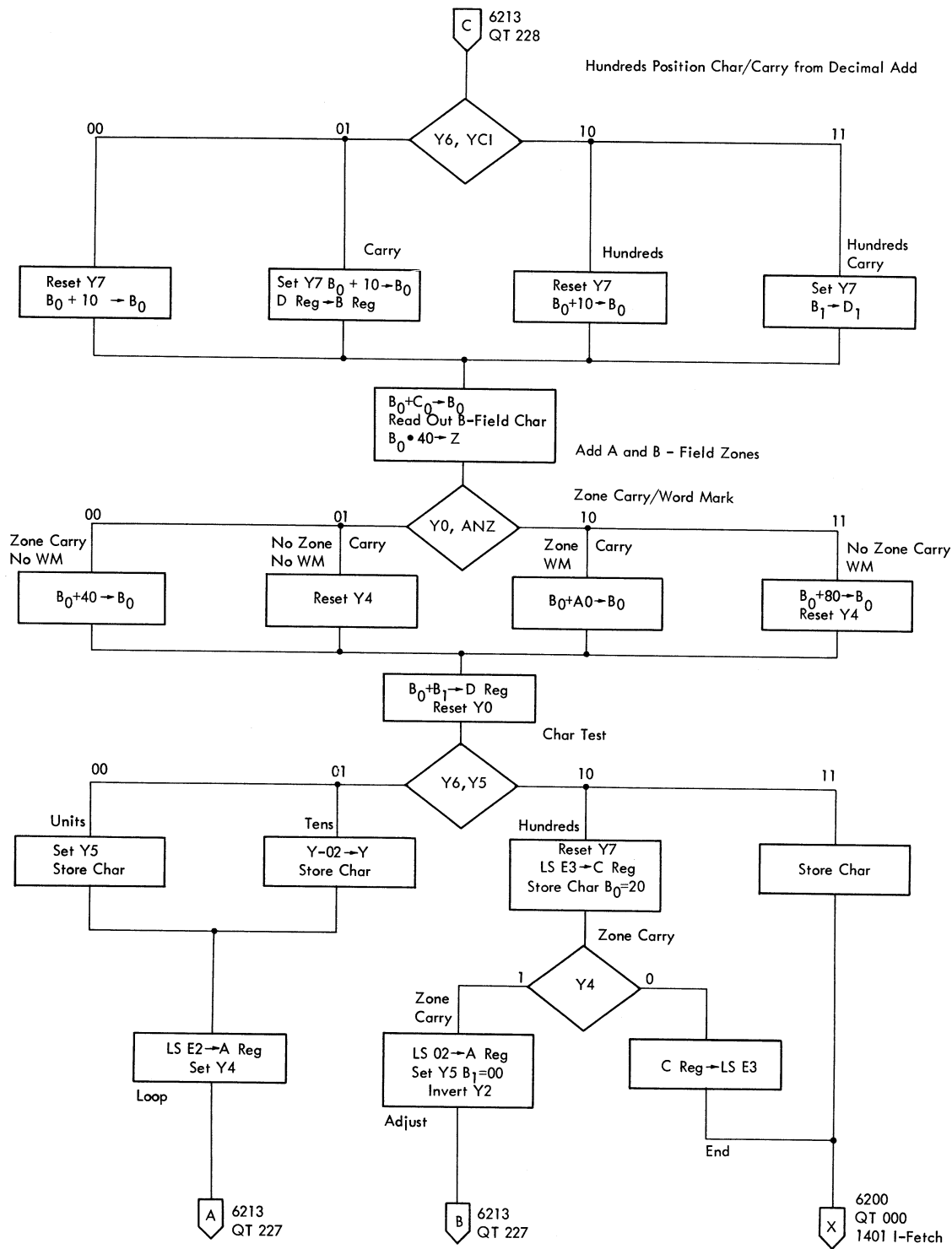
Y	Means
0	B-Field WM
2	BAR (Mod 2)
3	AAR (Mod 2)
4	Zone Carry
5	Tens
6	Hundreds
7	Numeric Carry



QT 228

FIGURE 6213. 1401 ADDRESS MODIFY

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



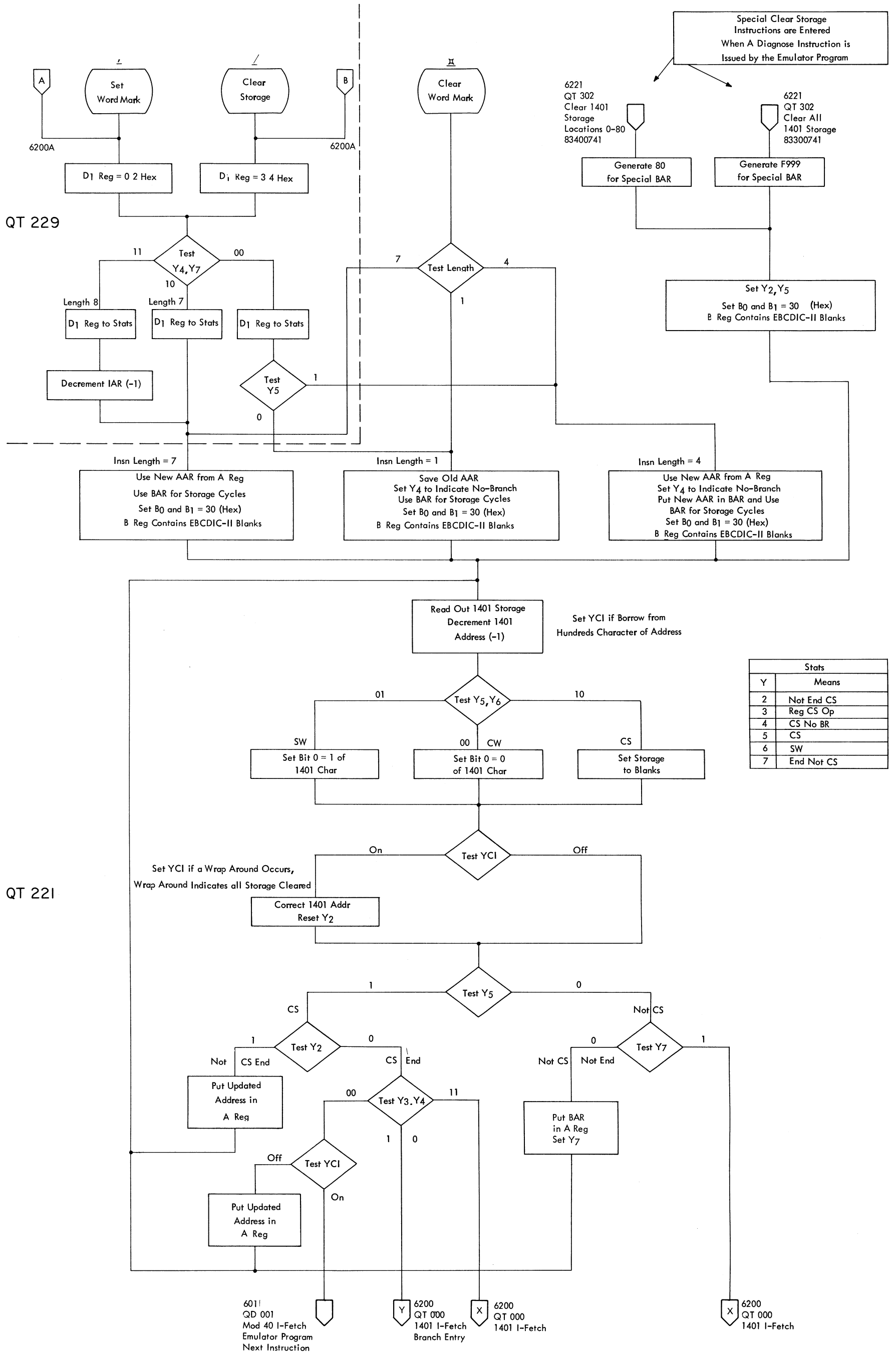
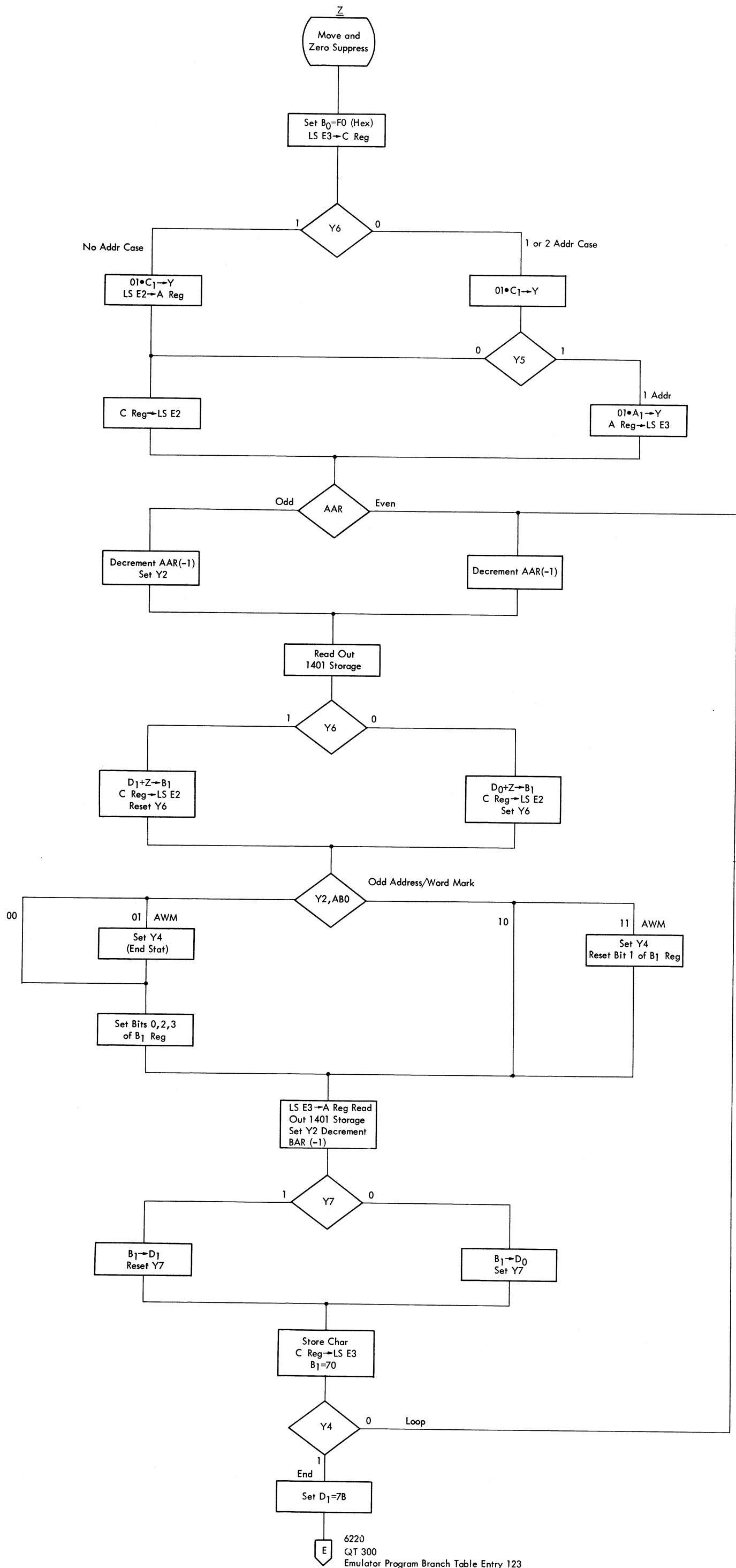


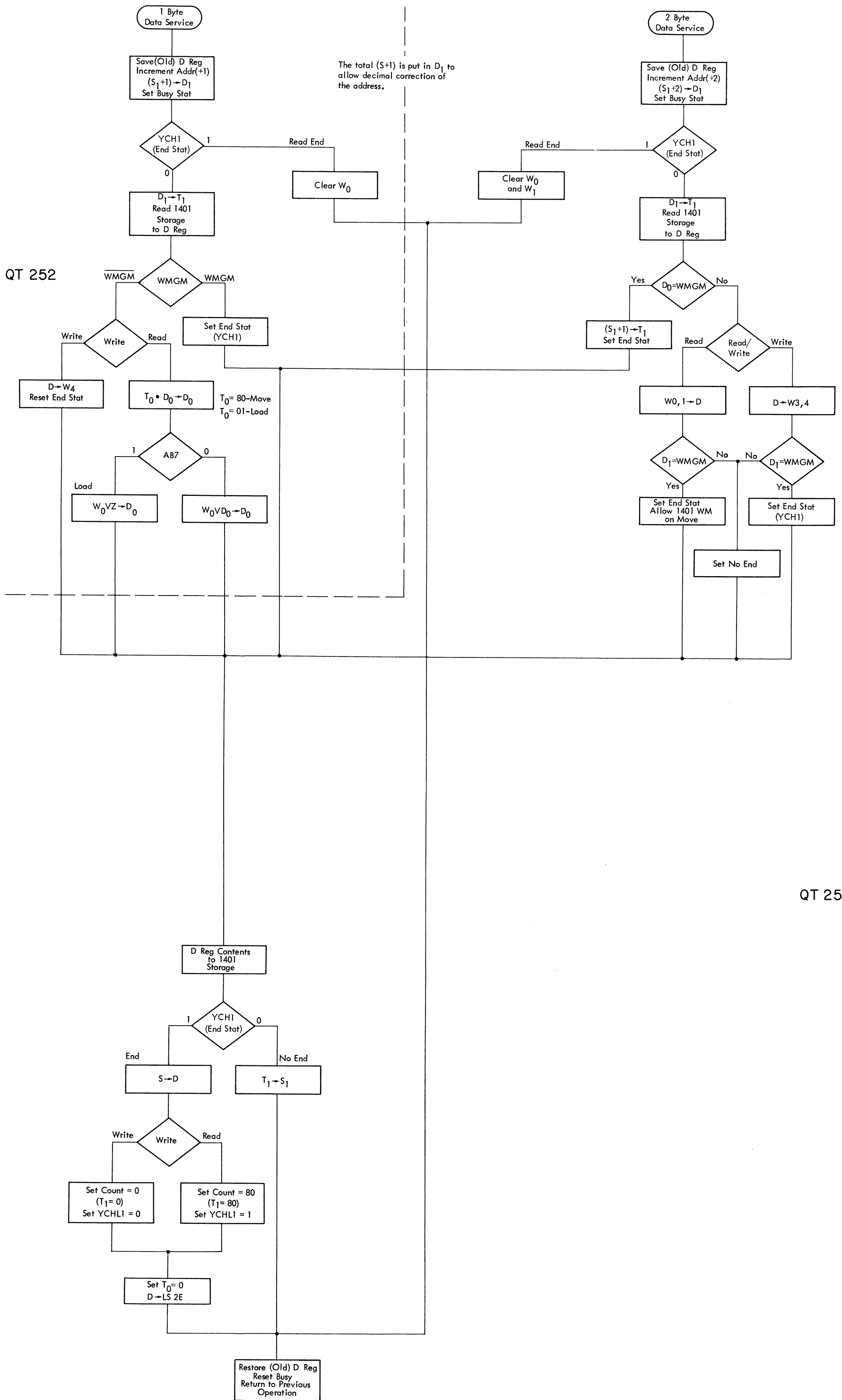
FIGURE 6214. 1401 SET WORD MARK, CLEAR WORD MARK, CLEAR STORAGE, AND SPECIAL CLEARS

QT 229



6220  
QT 300  
Emulator Program Branch Table Entry 123

FIGURE 6215. 1401 MOVE CHARACTERS AND SUPPRESS ZEROS



QT 252

QT 251

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

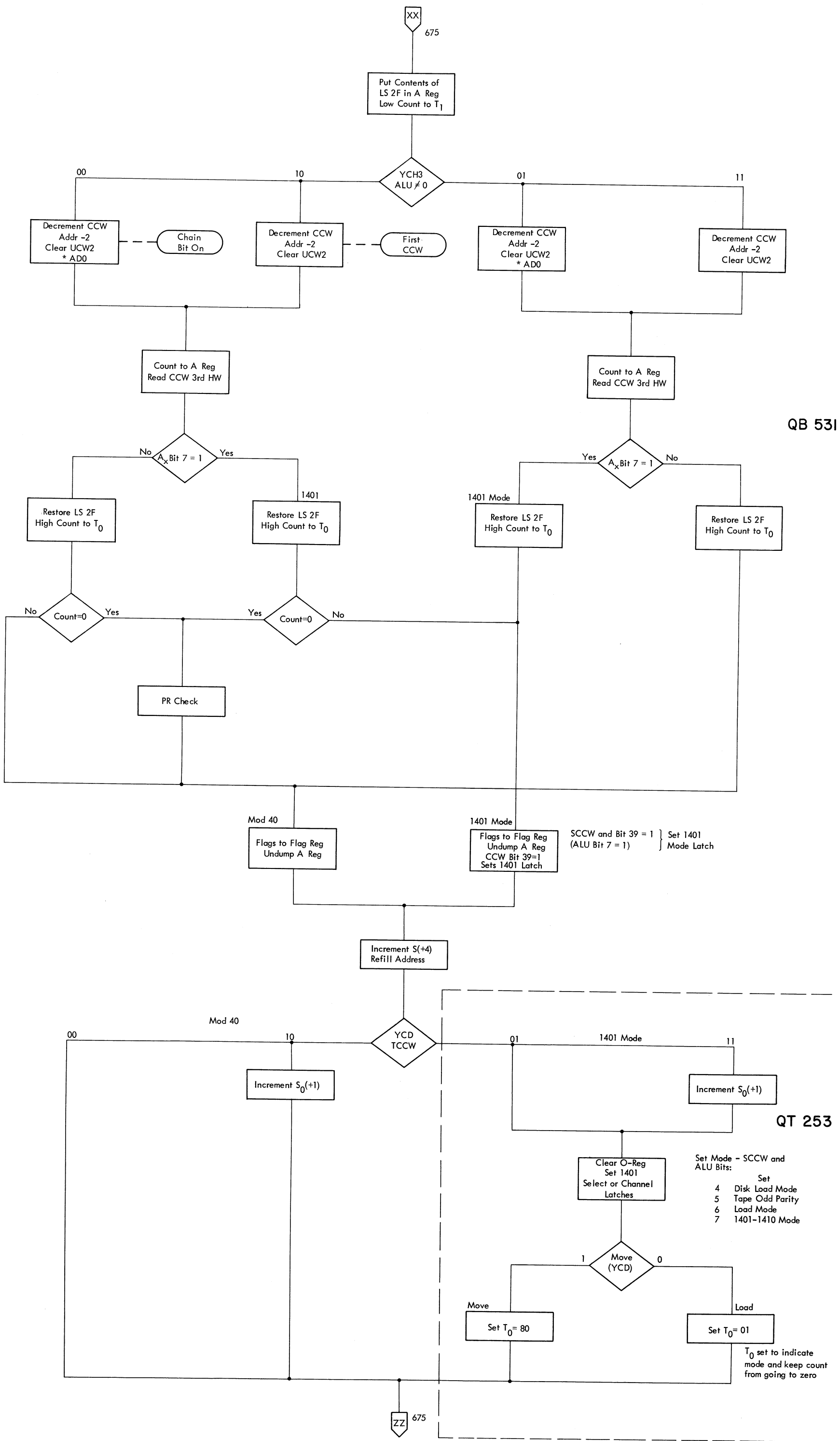


FIGURE 6217. SET SELECTOR CHANNEL TO 1401 MODE

QT 256

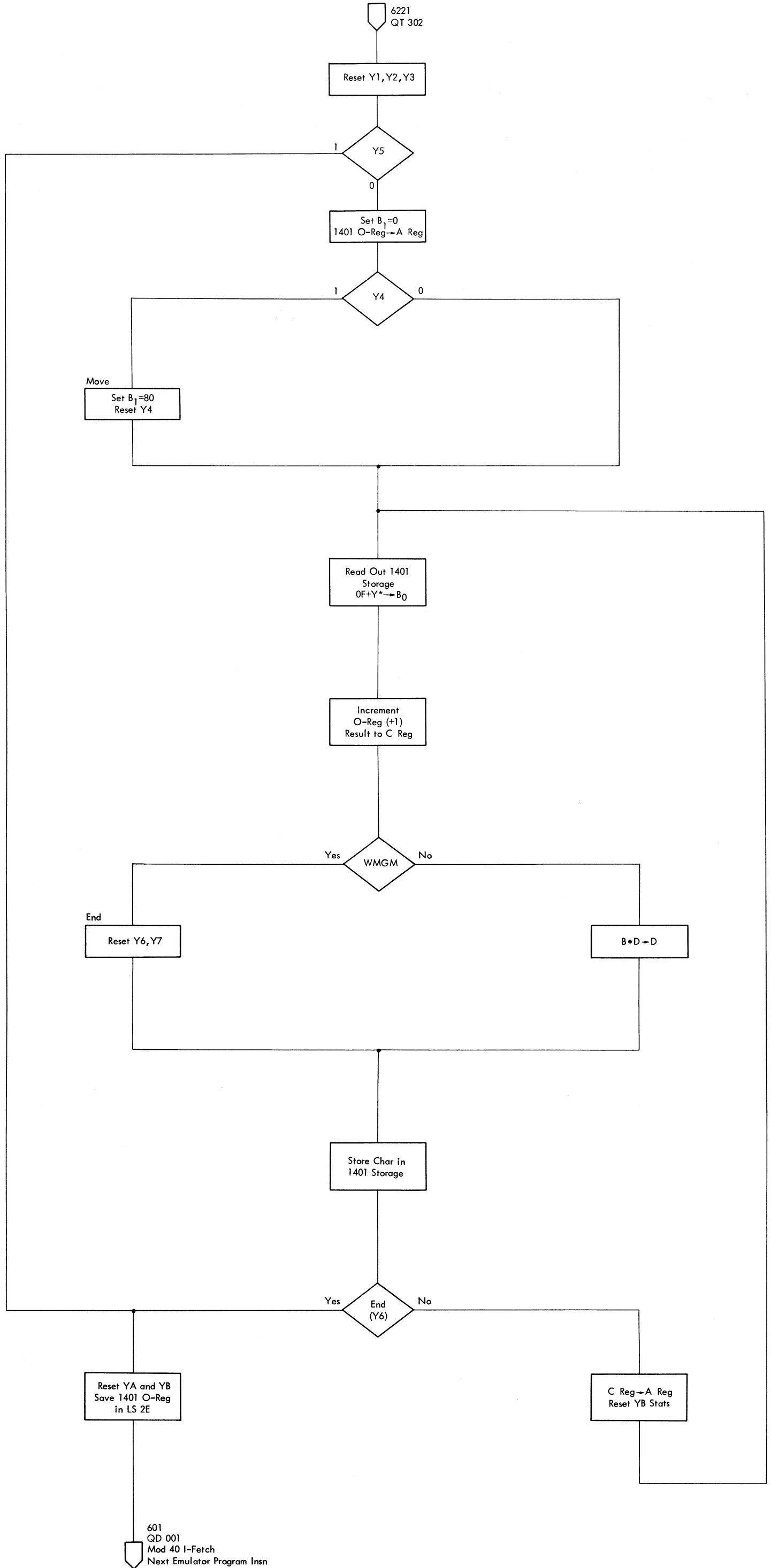
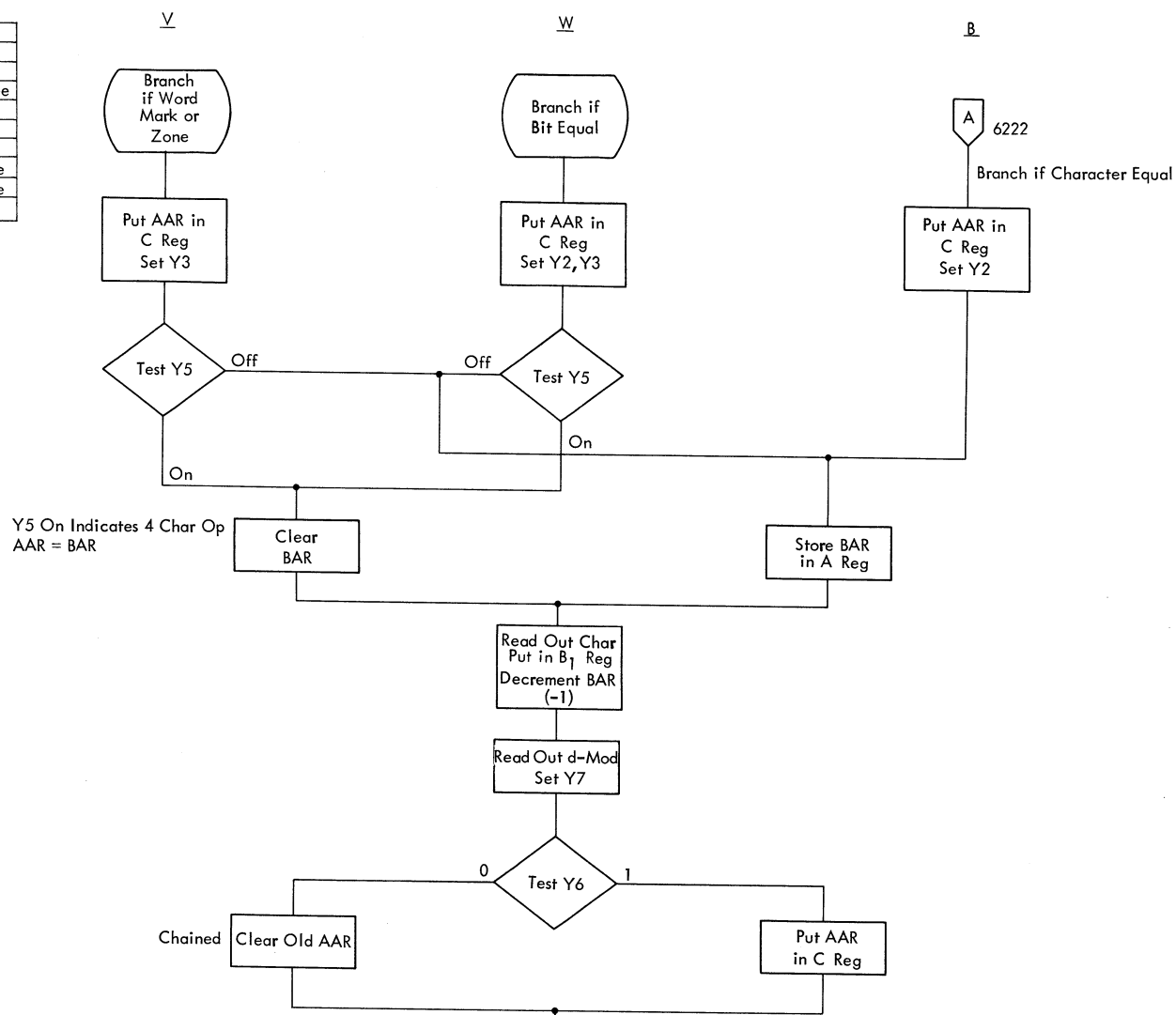


FIGURE 6218. 1401 TAPE OPERATIONS

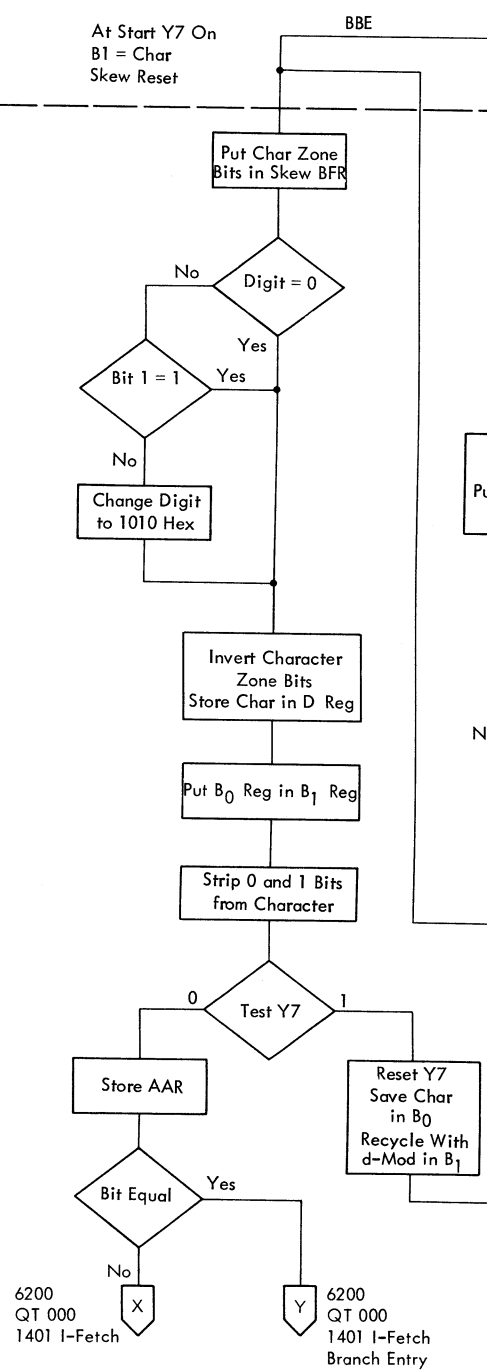
V (1) (B) d

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

QT 300



QT 301



QT 302

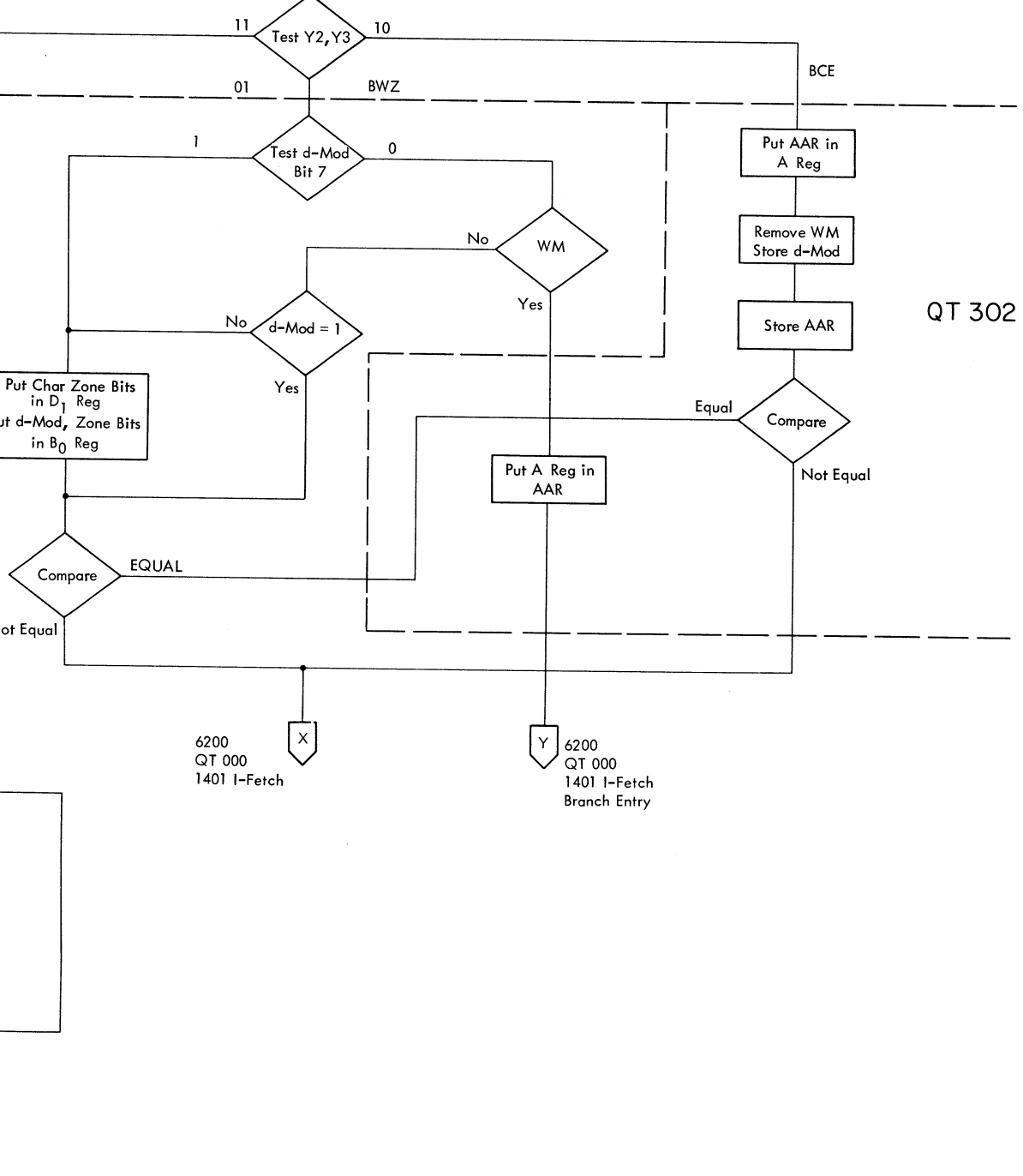
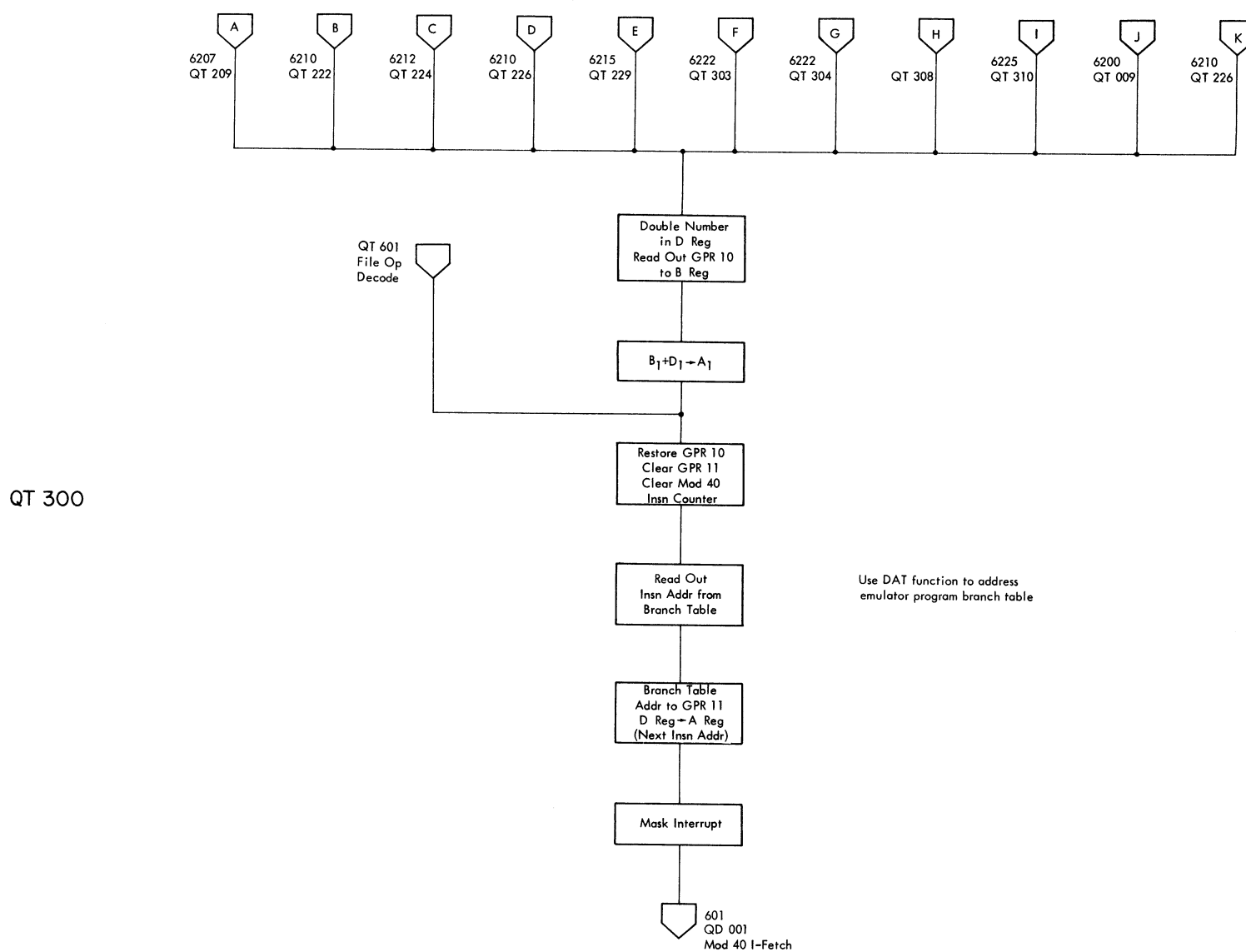
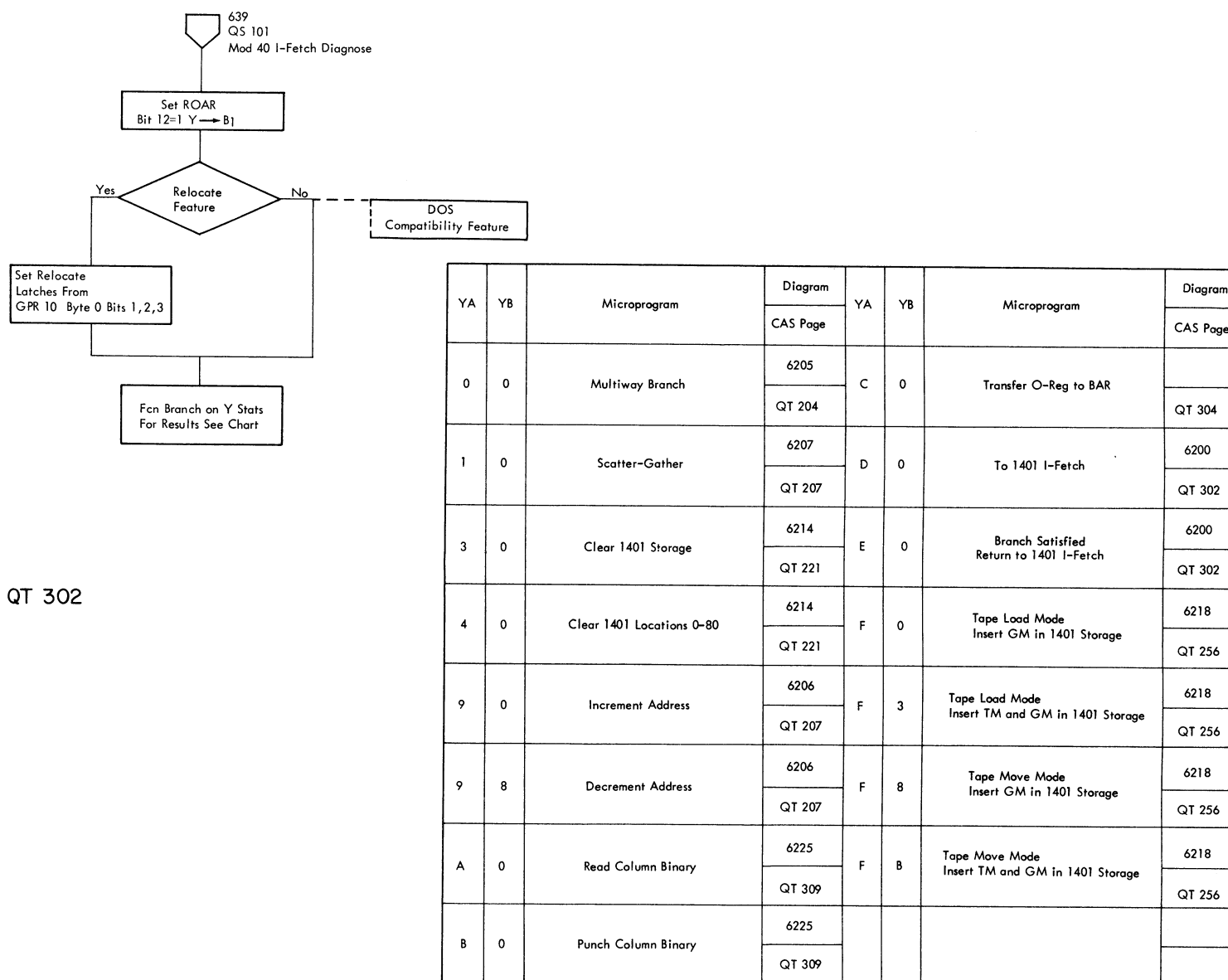


FIGURE 6219. 1401 BRANCH IF: WORD MARK OR ZONE, BIT EQUAL, OR CHARACTER EQUAL



QT 300

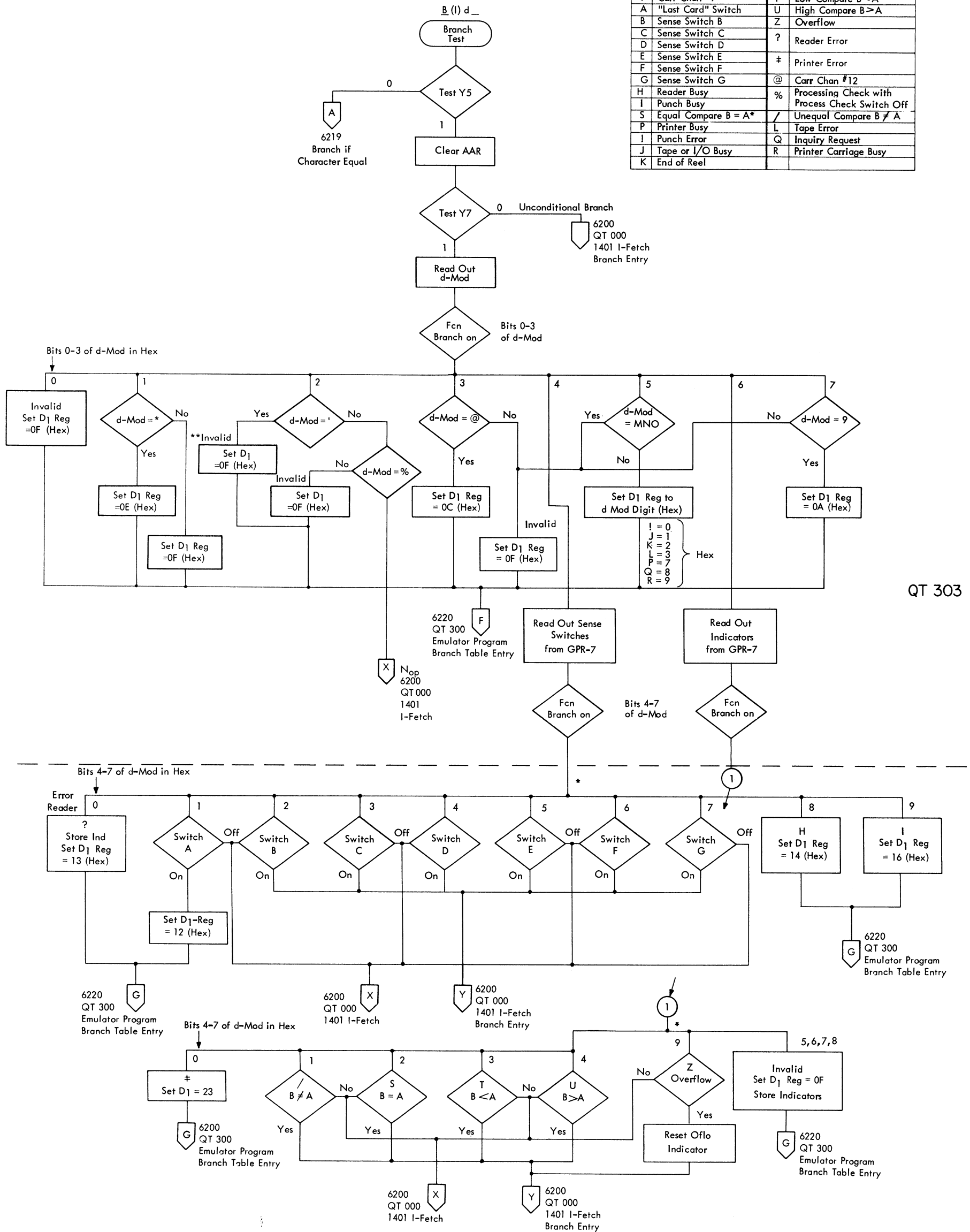
FIGURE 6220. 1401 EMULATOR PROGRAM ENTRY



QT 302

● 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
J	Punch Error	Q	Inquiry Request
K	Tape or I/O Busy	R	Printer Carriage Busy



QT 303

QT 304

FIGURE 6222. 1401 BRANCH TESTS



QT 305

QT 306

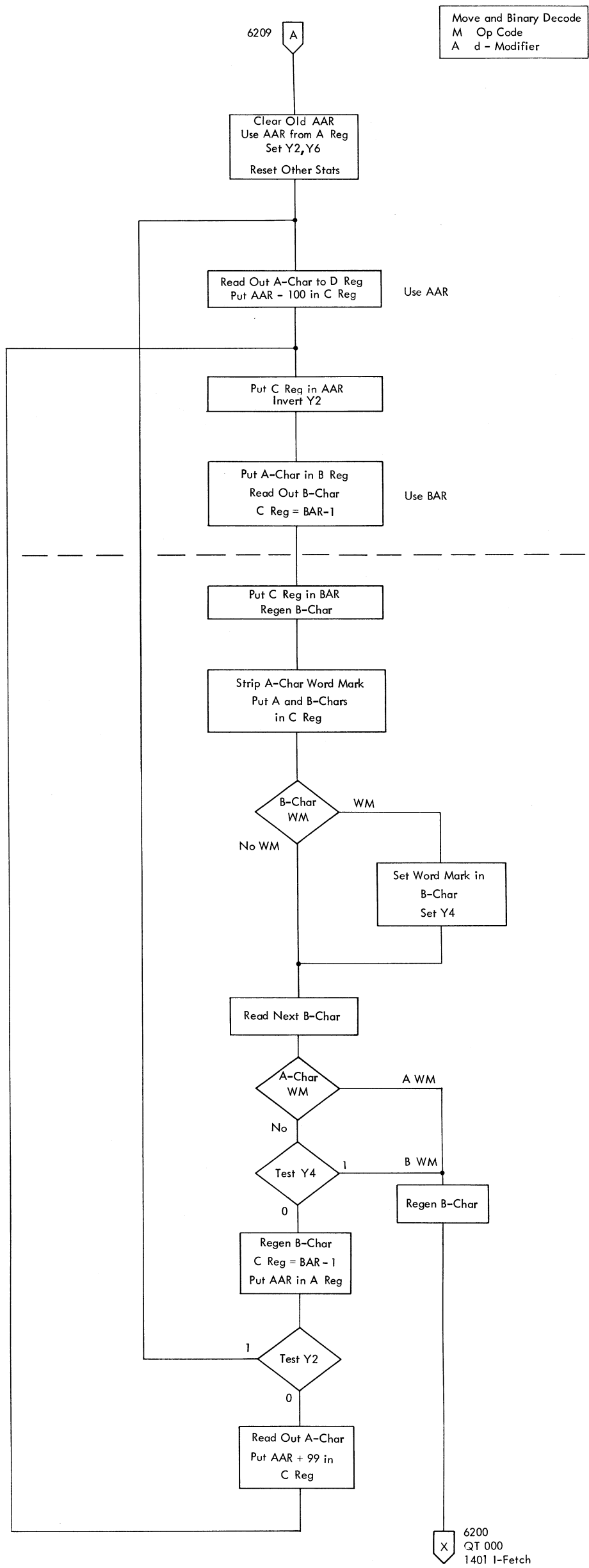
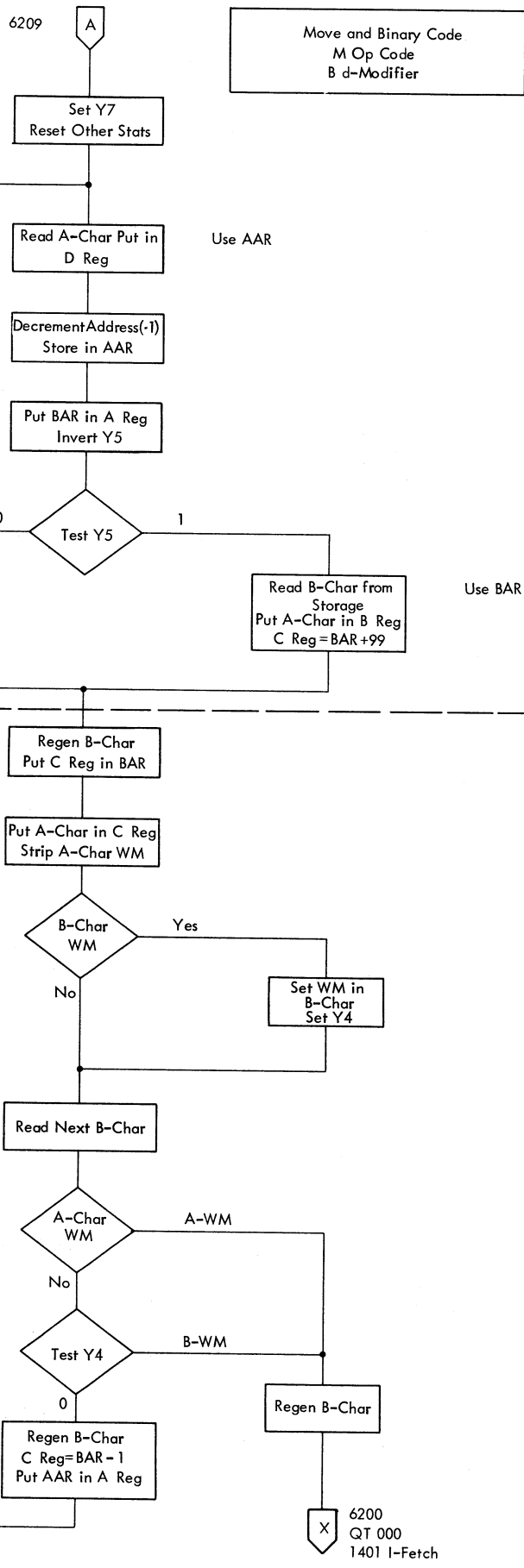


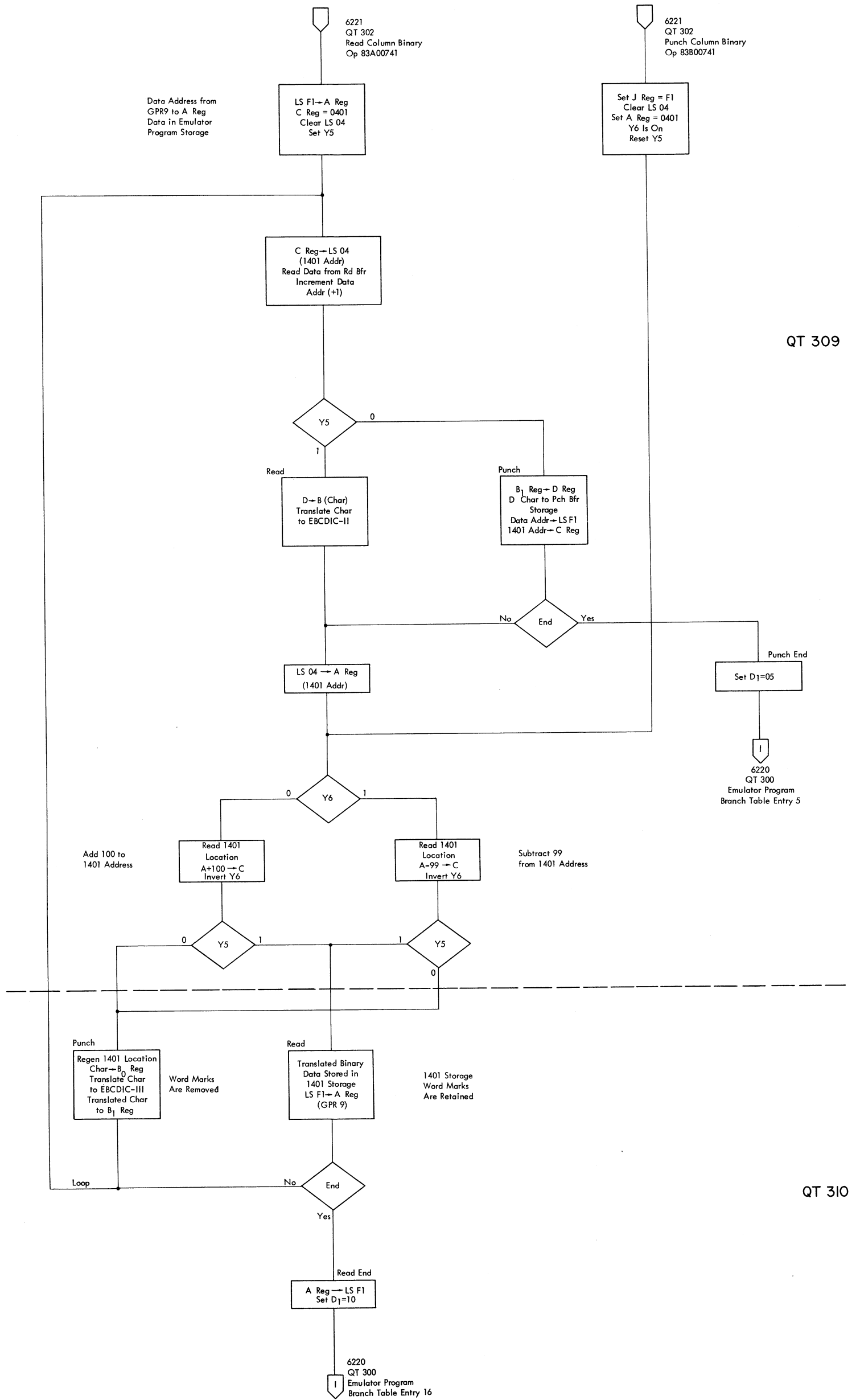
FIGURE 6223. 1401 MOVE AND BINARY DECODE



QT 305

QT 306

FIGURE 6224. 1401 MOVE AND BINARY CODE

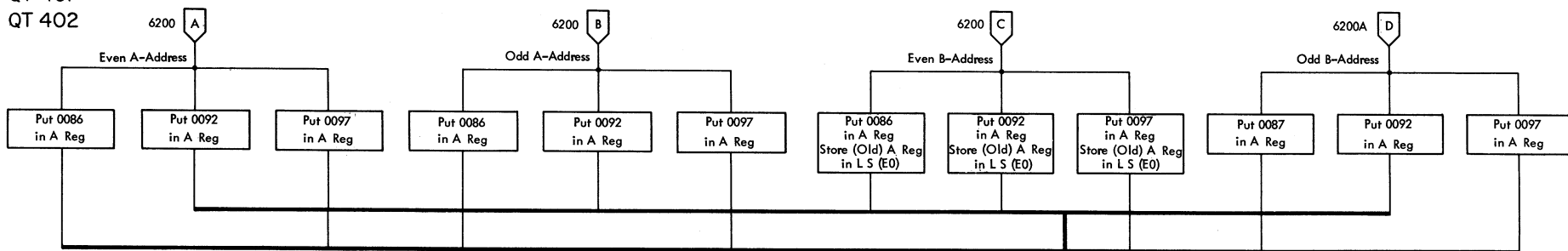


QT 309

QT 310

FIGURE 6225. 1401 READ AND PUNCH COLUMN BINARY

QT 401  
QT 402

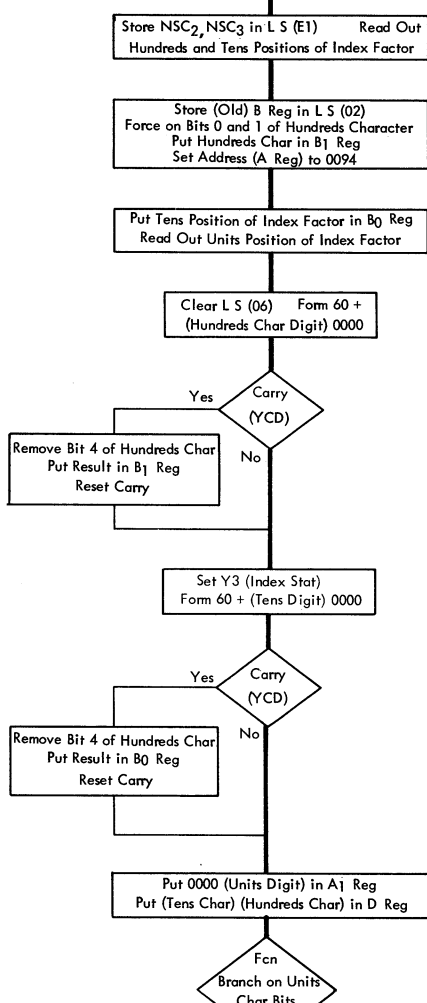
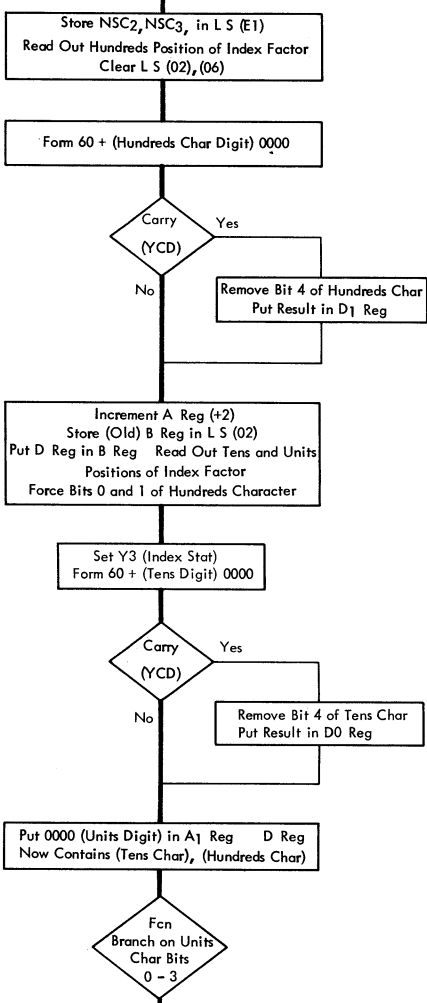


A Reg = 86, 87, 97

A Reg = 86, 87, 92, 97

QT 404

QT 403



Bits 0 - 3 in Hex

Bits 0 - 3 in Hex

QT 404  
QT 405



QT 405

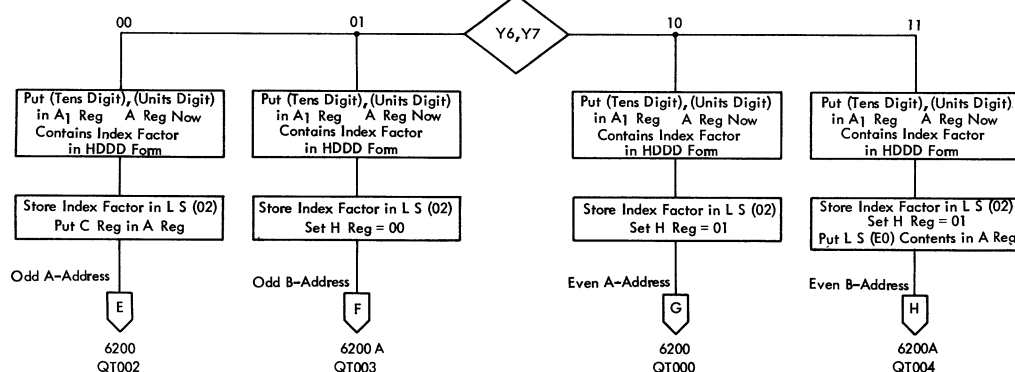
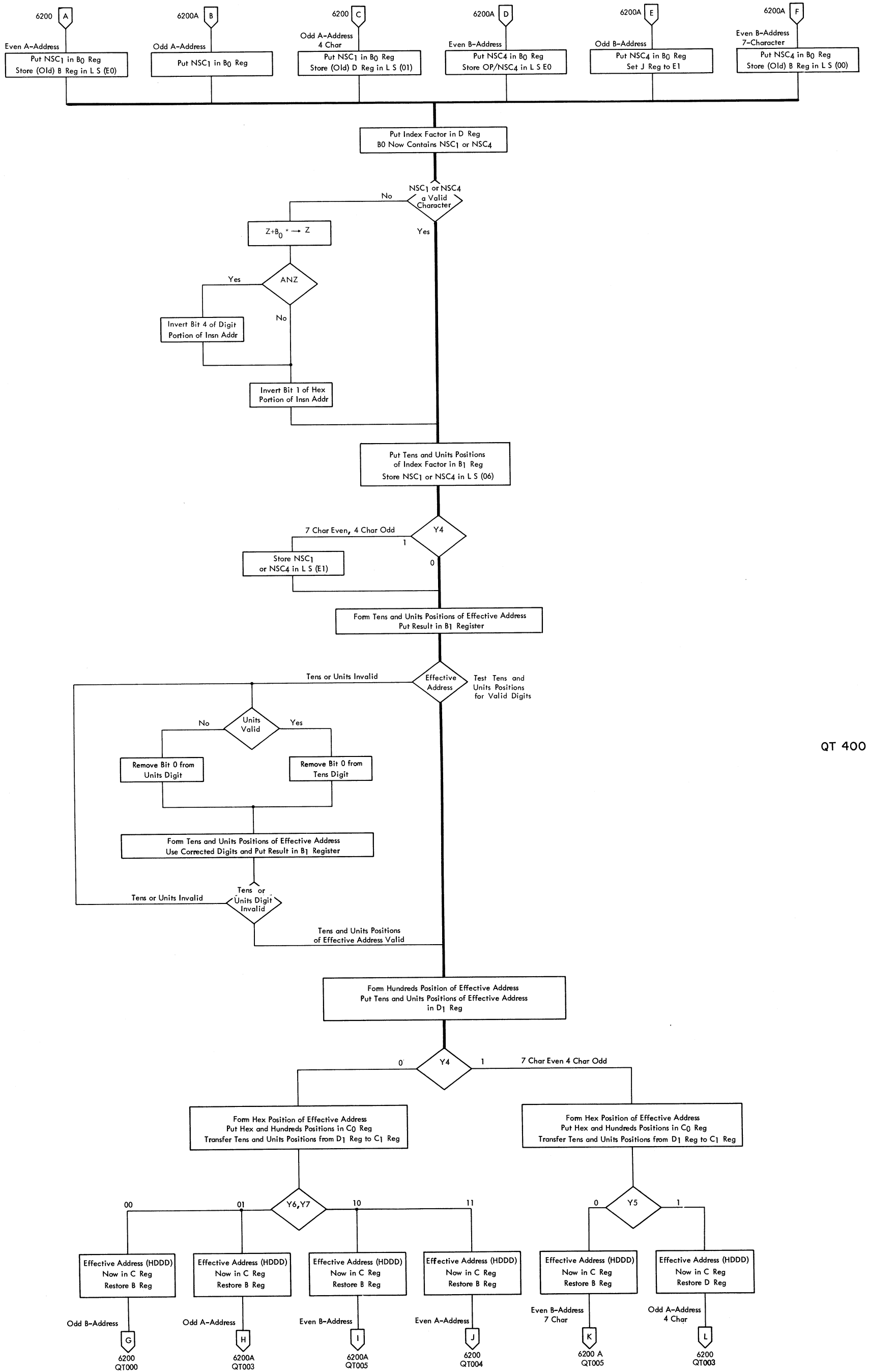


FIGURE 6226. 1401 INDEX FACTOR FETCH



QT 400

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
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

Stats	
Y2	Odd Address
Y4	G Op Code
Y5	Second Address

Local Storage Contents	
LS 01	Working Instruction Counter
LS 02	Translated Op Code, TTh Position
LS 05	Odd Character

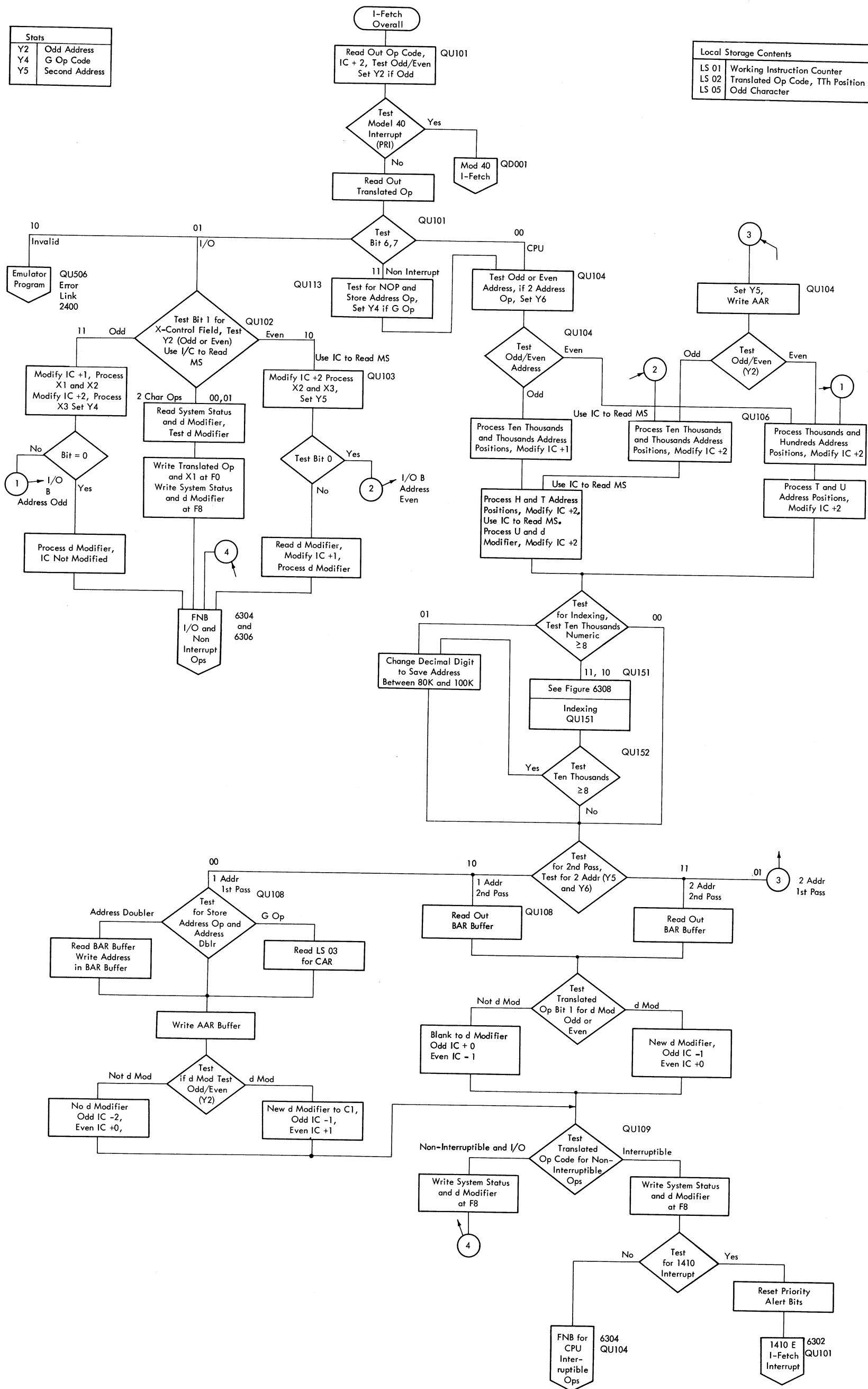


FIGURE 6301. 1410 E INSTRUCTION-FETCH OVERALL

Stats	
Y1	First Return from Diagnose
Y2	Odd Start Address
Y6	2 Address Type Ops

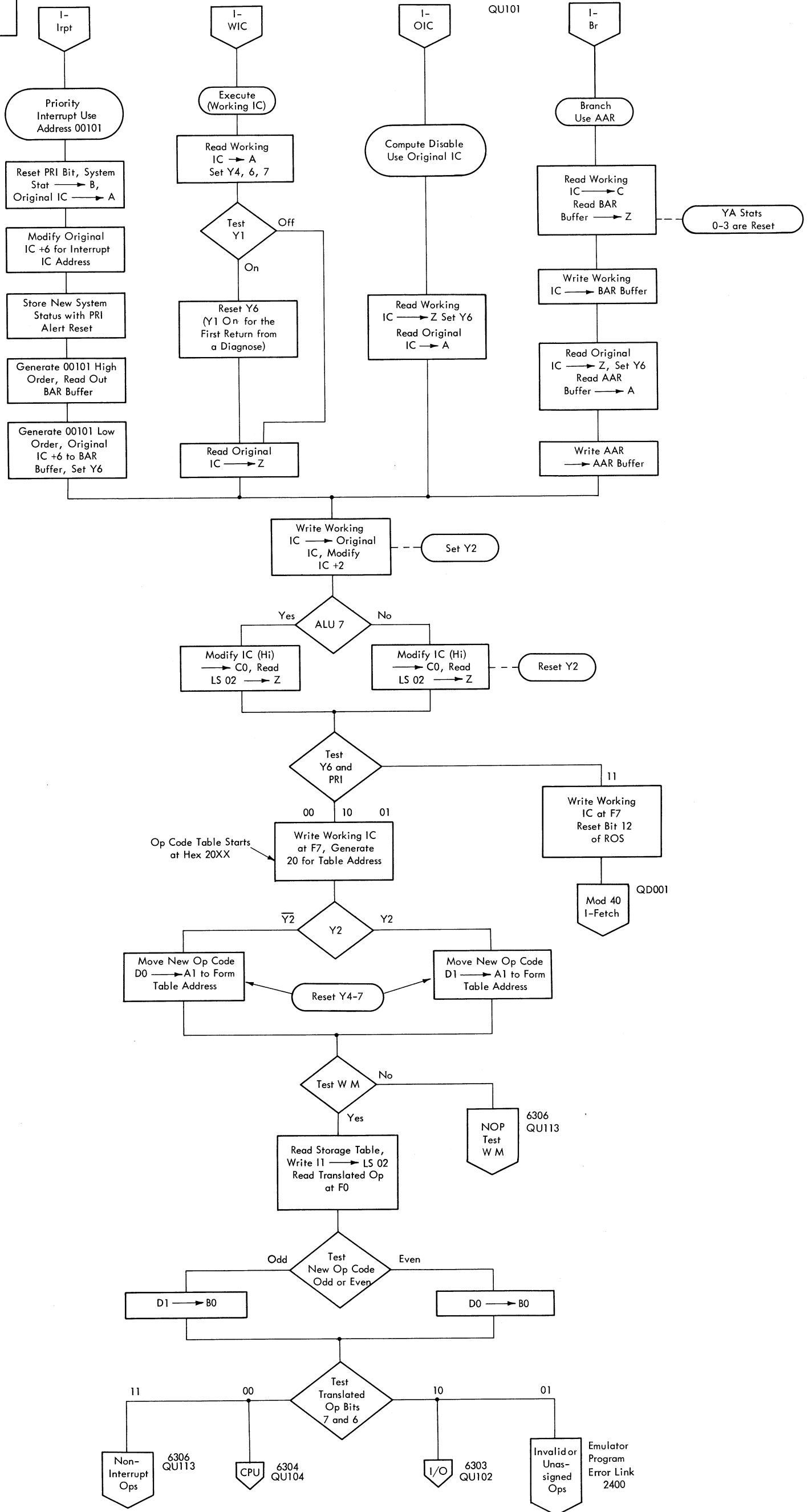


FIGURE 6302. 1410 E INSTRUCTION-FETCH START



Stats	
Y2	Odd Start Address
Y5	2nd Pass of Address
Y6	2 Address Type Ops

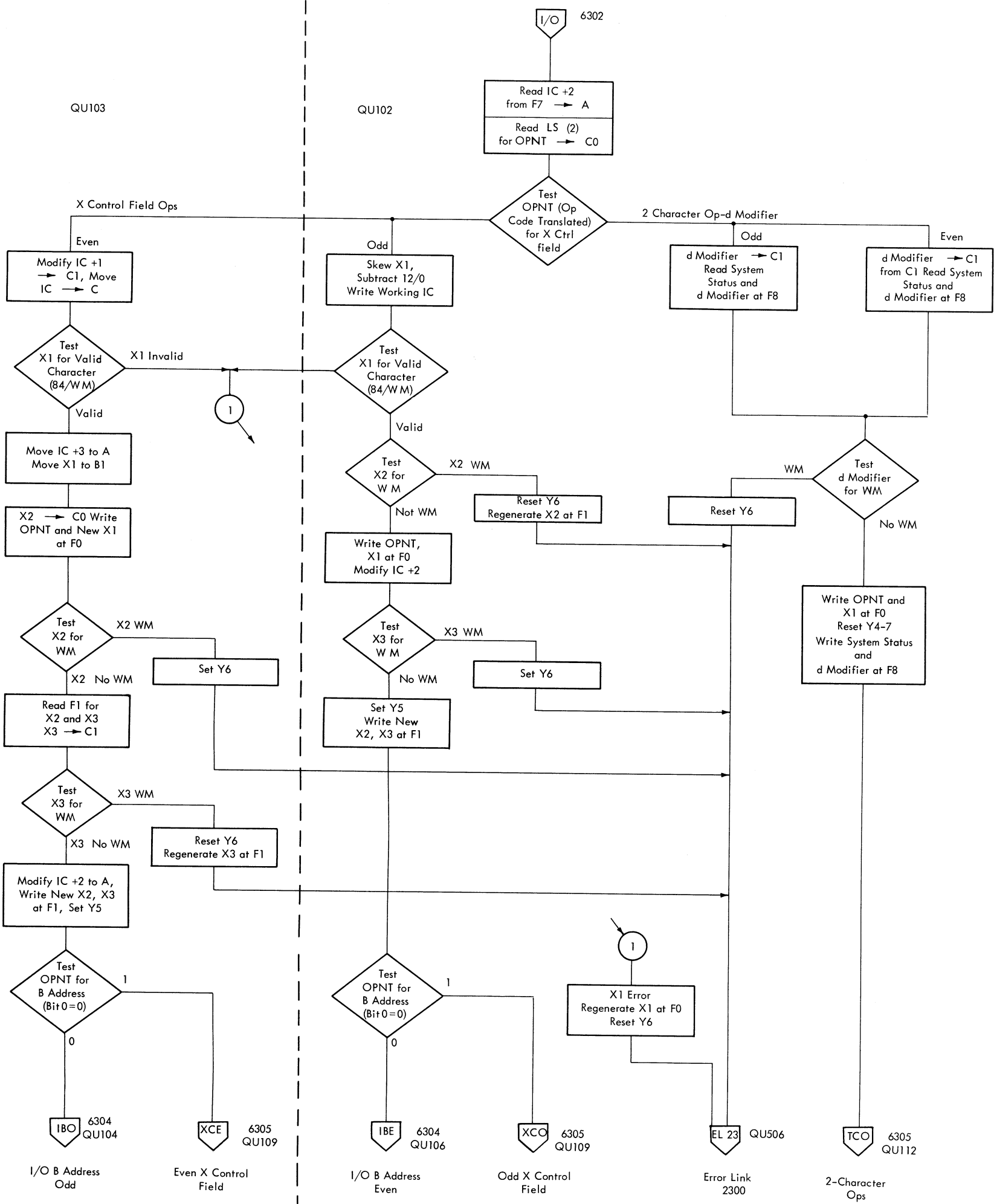
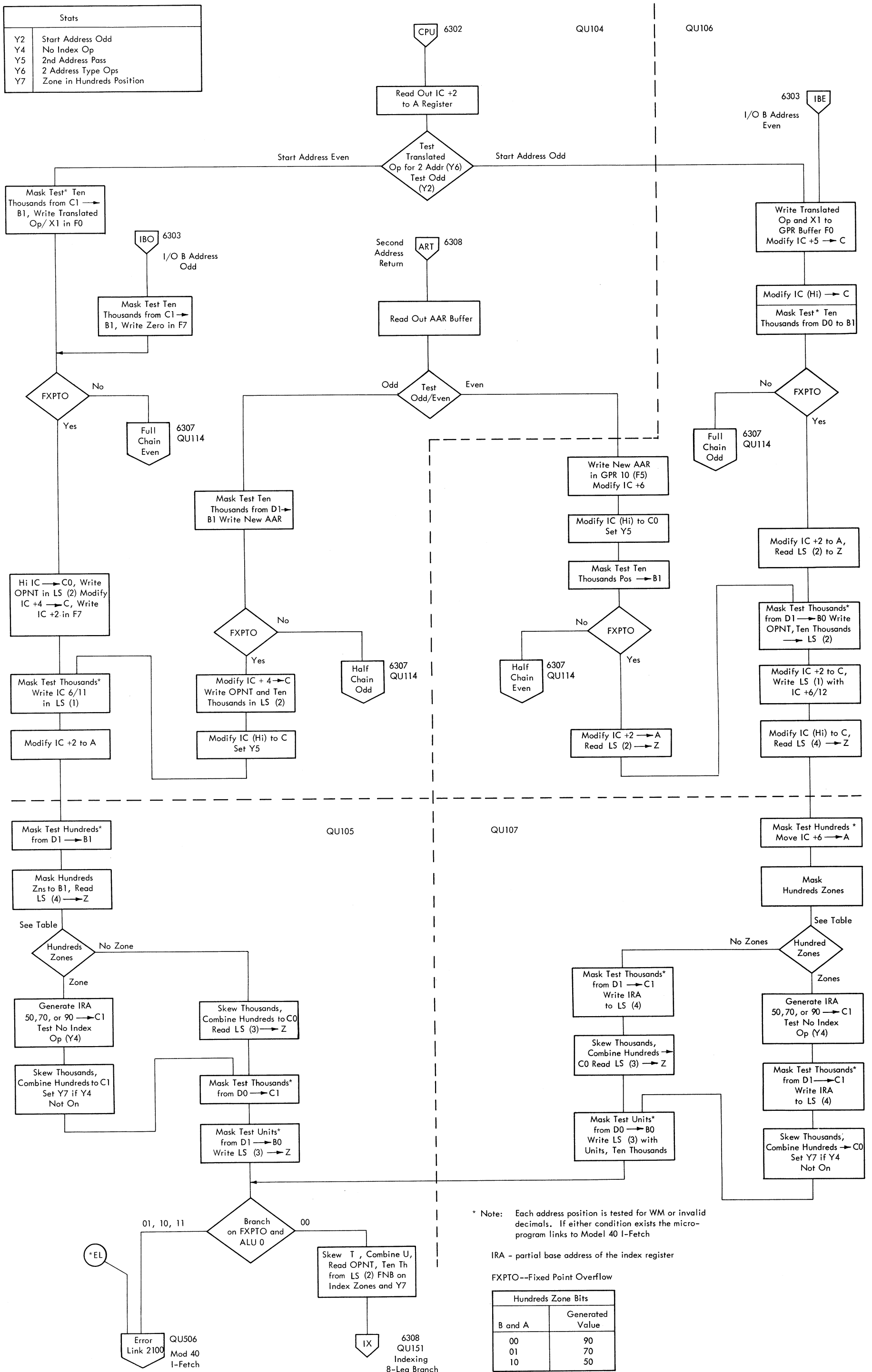


FIGURE 6303. 1410 E X CONTROL FIELD READOUT



Stats	
Y2	Start Address Odd
Y4	No Index Op
Y5	2nd Address Pass
Y6	2 Address Type Ops
Y7	Zone in Hundreds Position

\* 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. 1410E ADDRESS READOUT

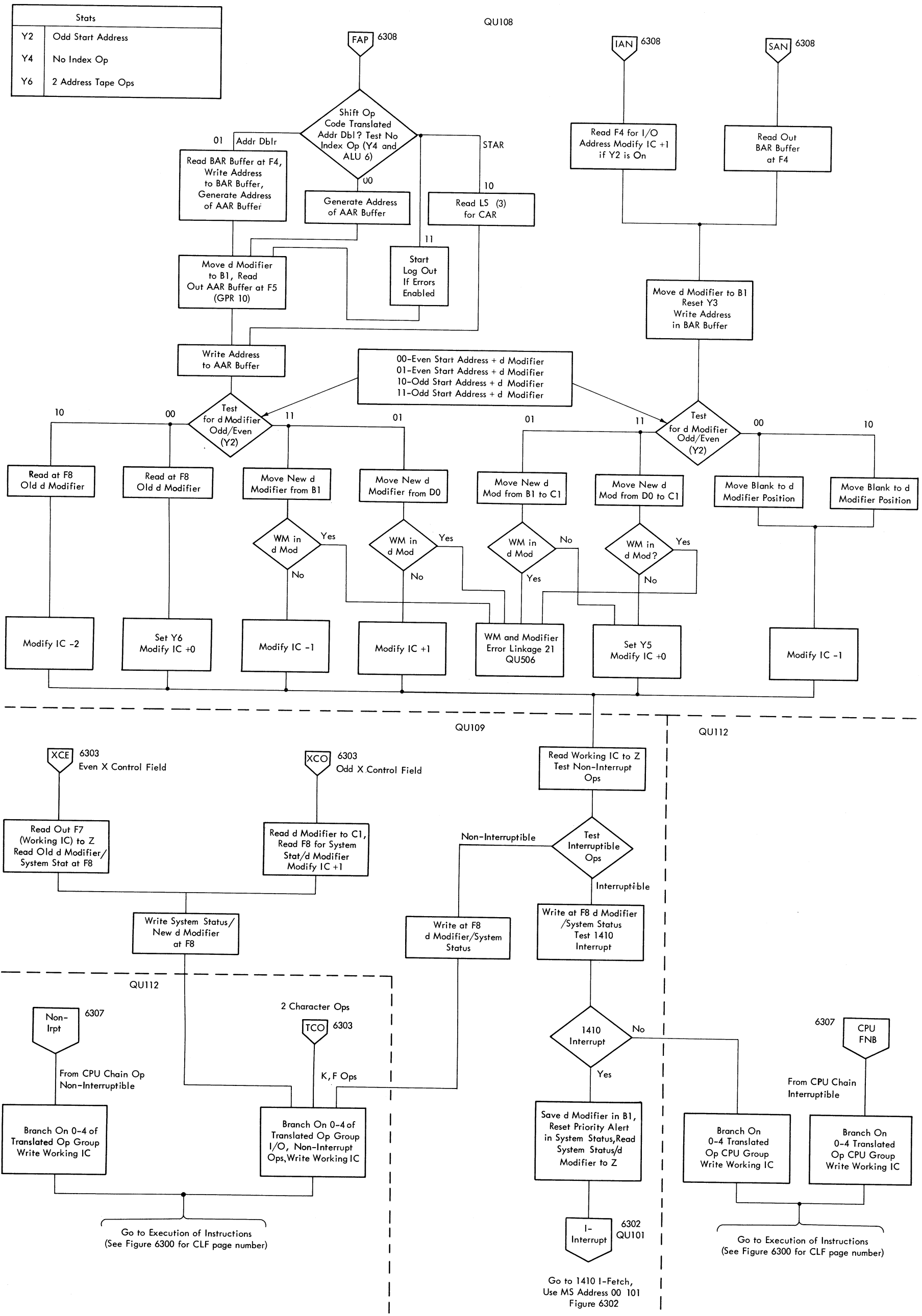


FIGURE 6305. 1410 E INSTRUCTION FETCH ENDING

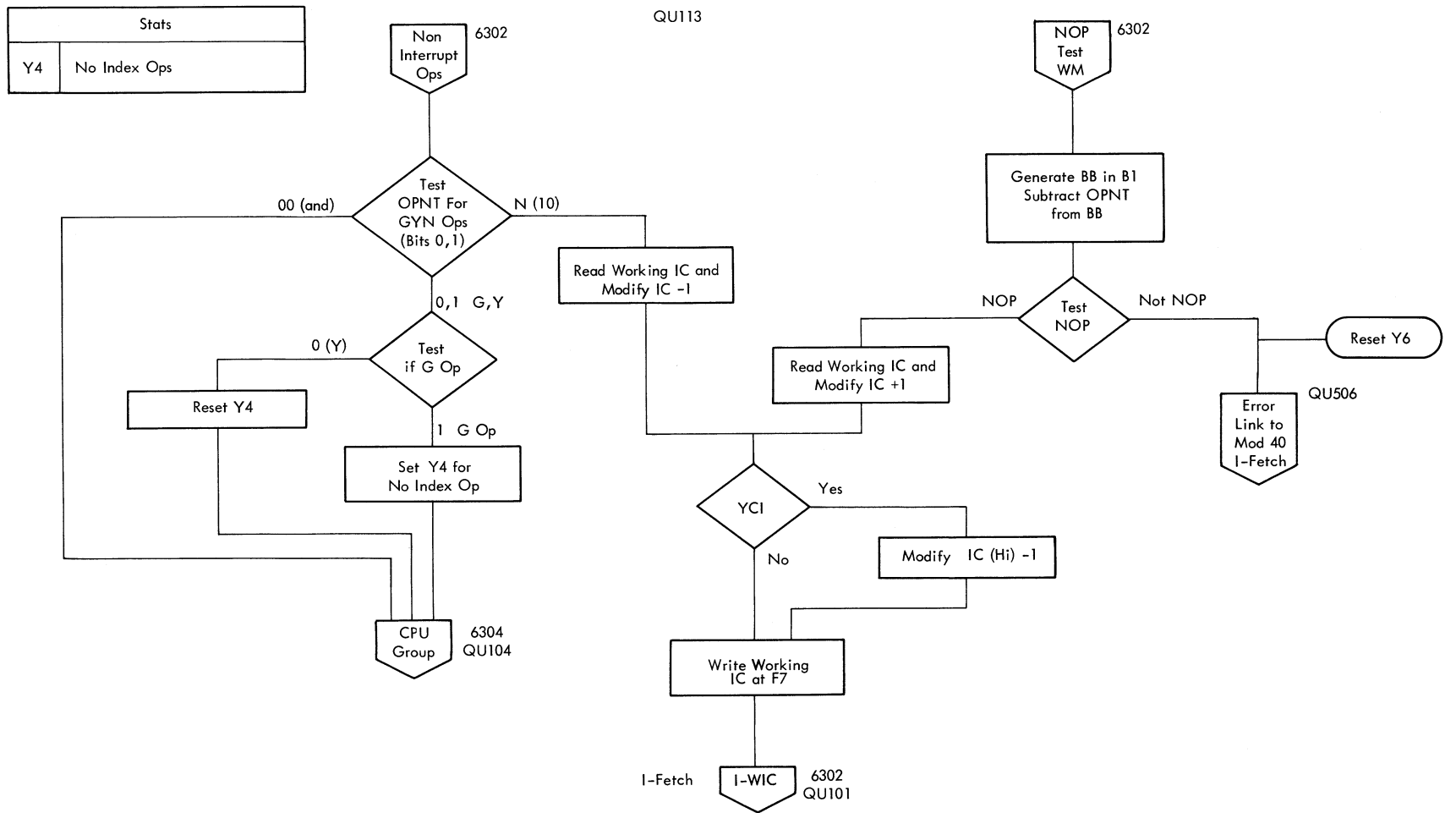


FIGURE 6306. 1410 E NOP AND NON-INTERRUPTIBLE OP CODES

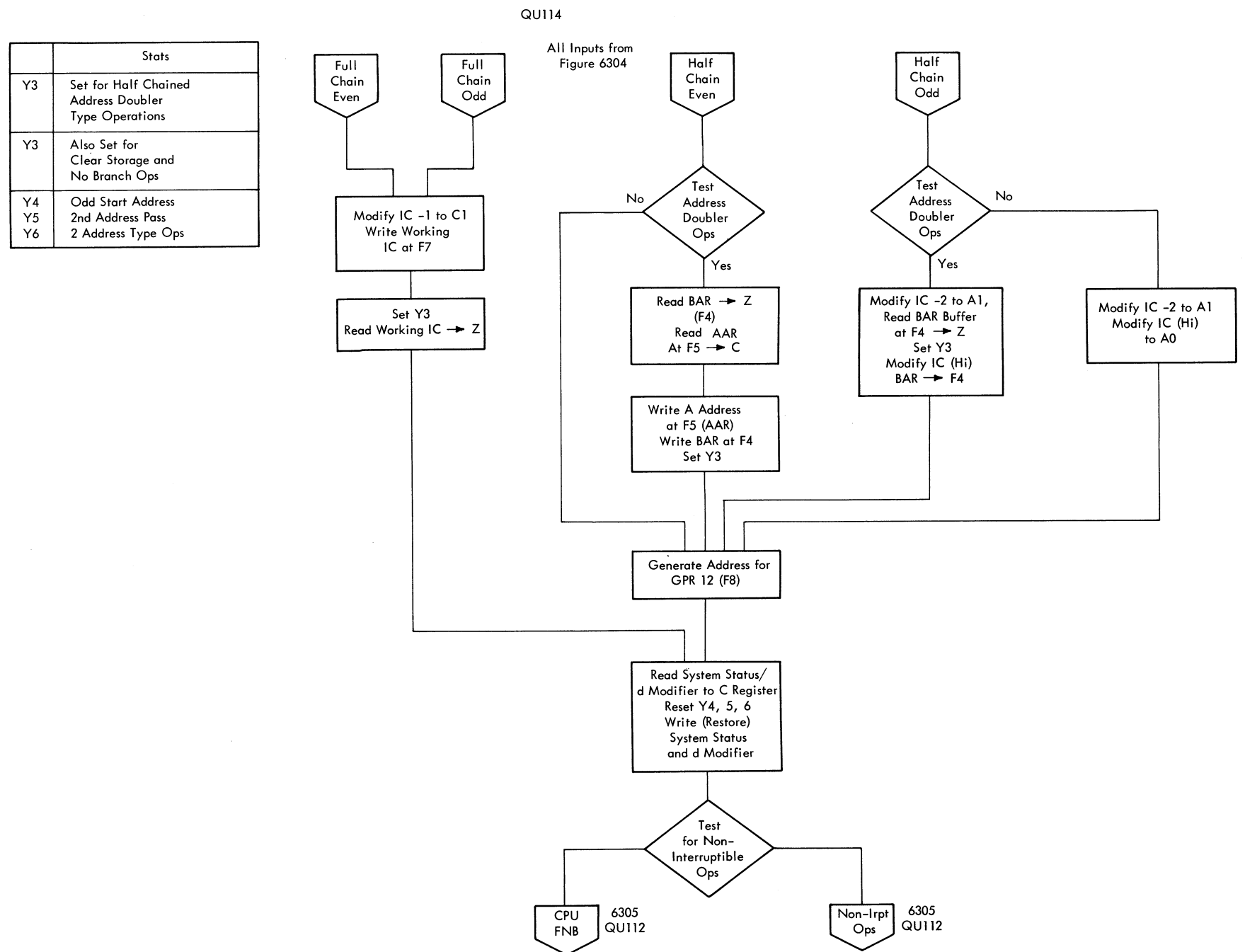


FIGURE 6307. 1410E CHAINING

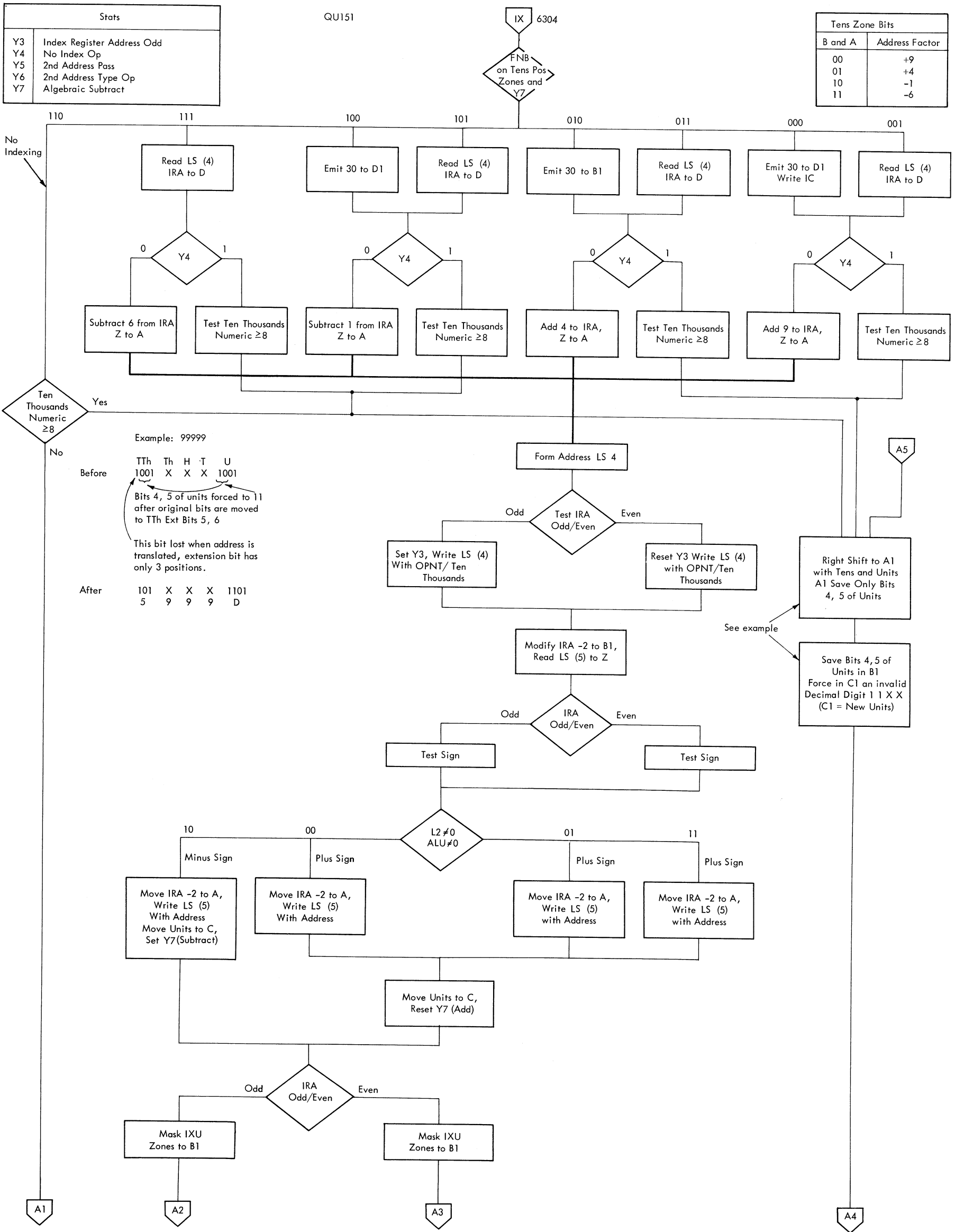


FIGURE 6308. 1410 E INDEXING (SHEET 1 OF 2)

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

QU152

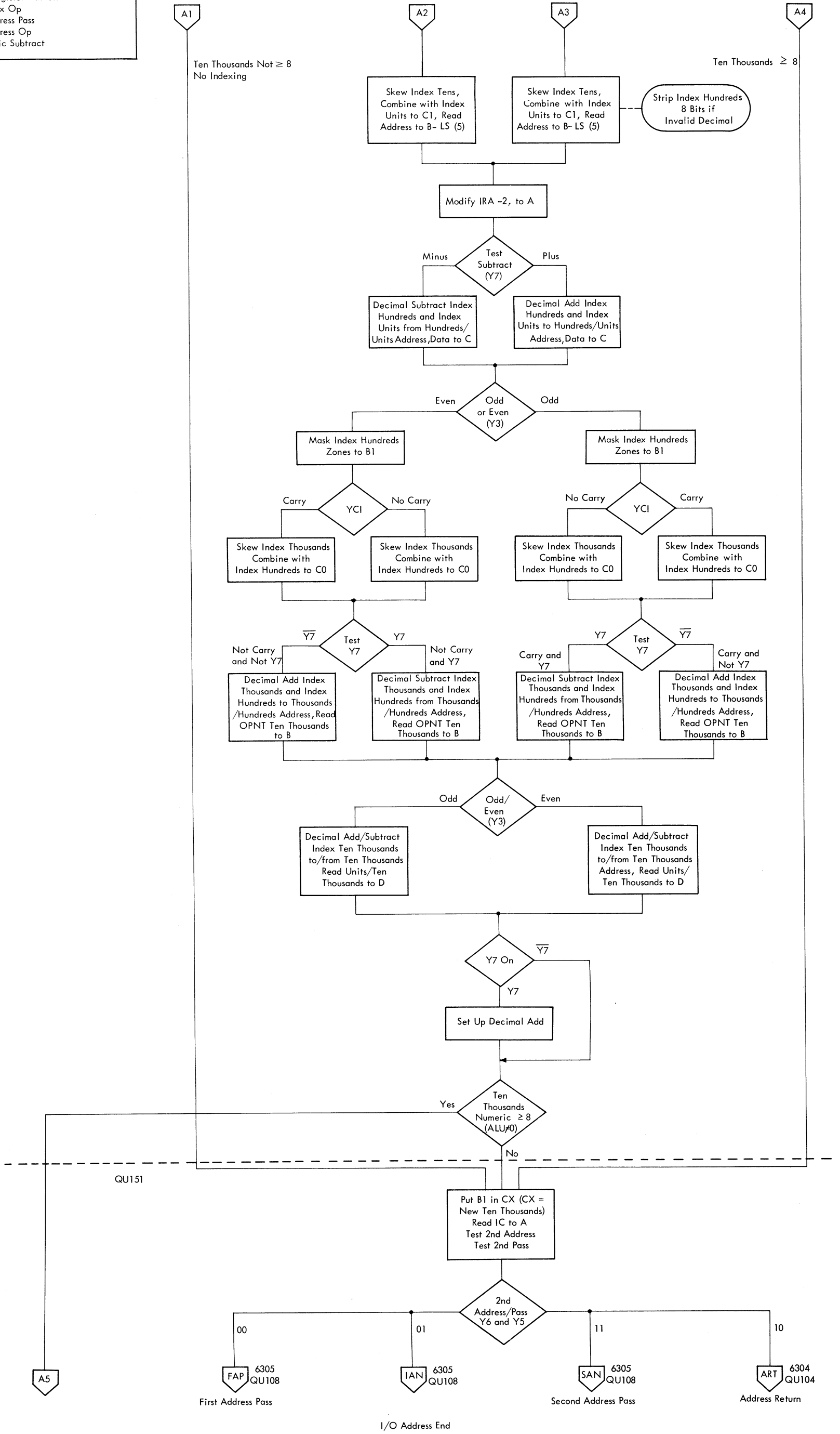


FIGURE 6308. 1410E INDEXING (SHEET 2 OF 2)

Stats	
Y6	Branch Ch 2 - X Op

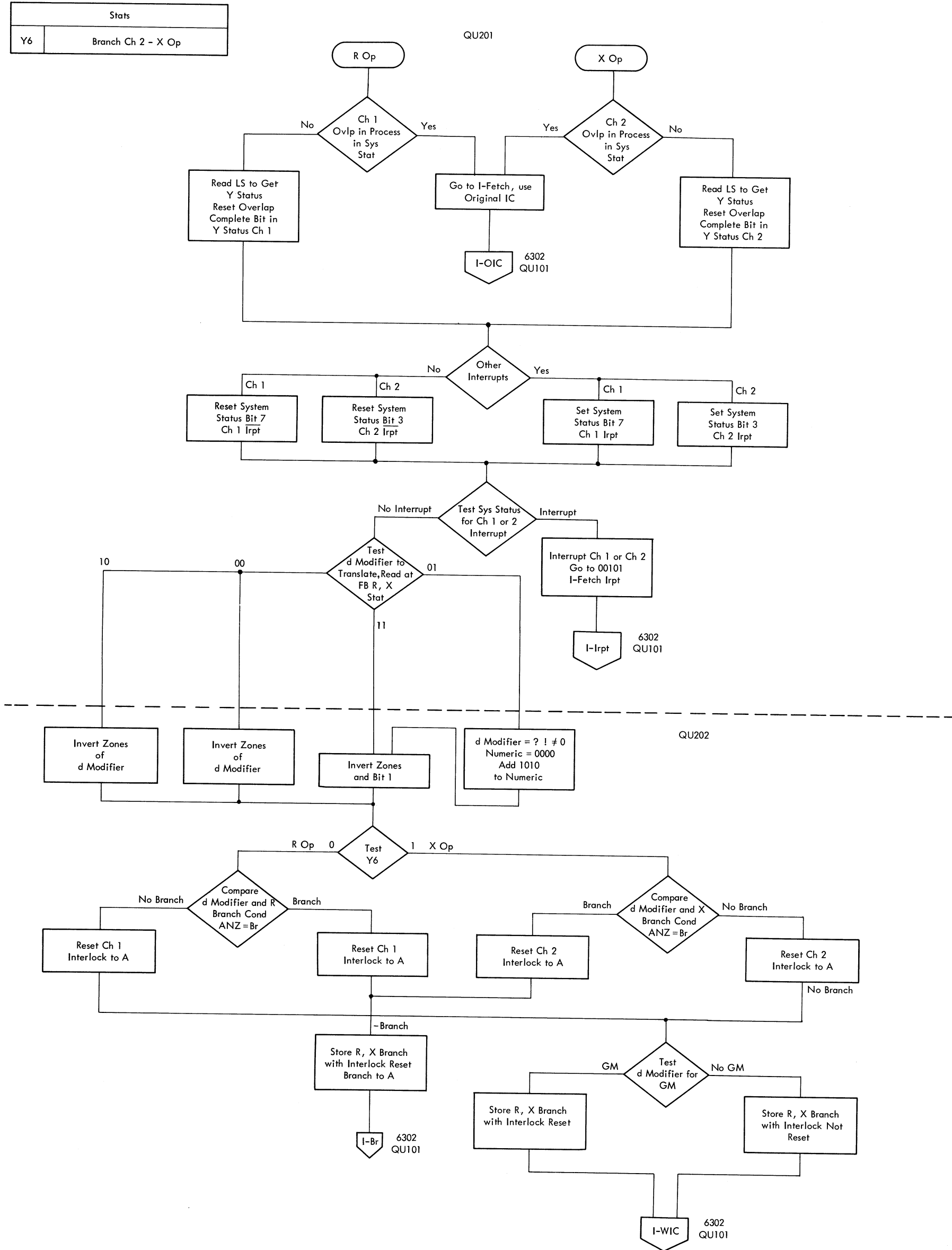


FIGURE 6309. 1410E BRANCH ON CHANNEL STATUS

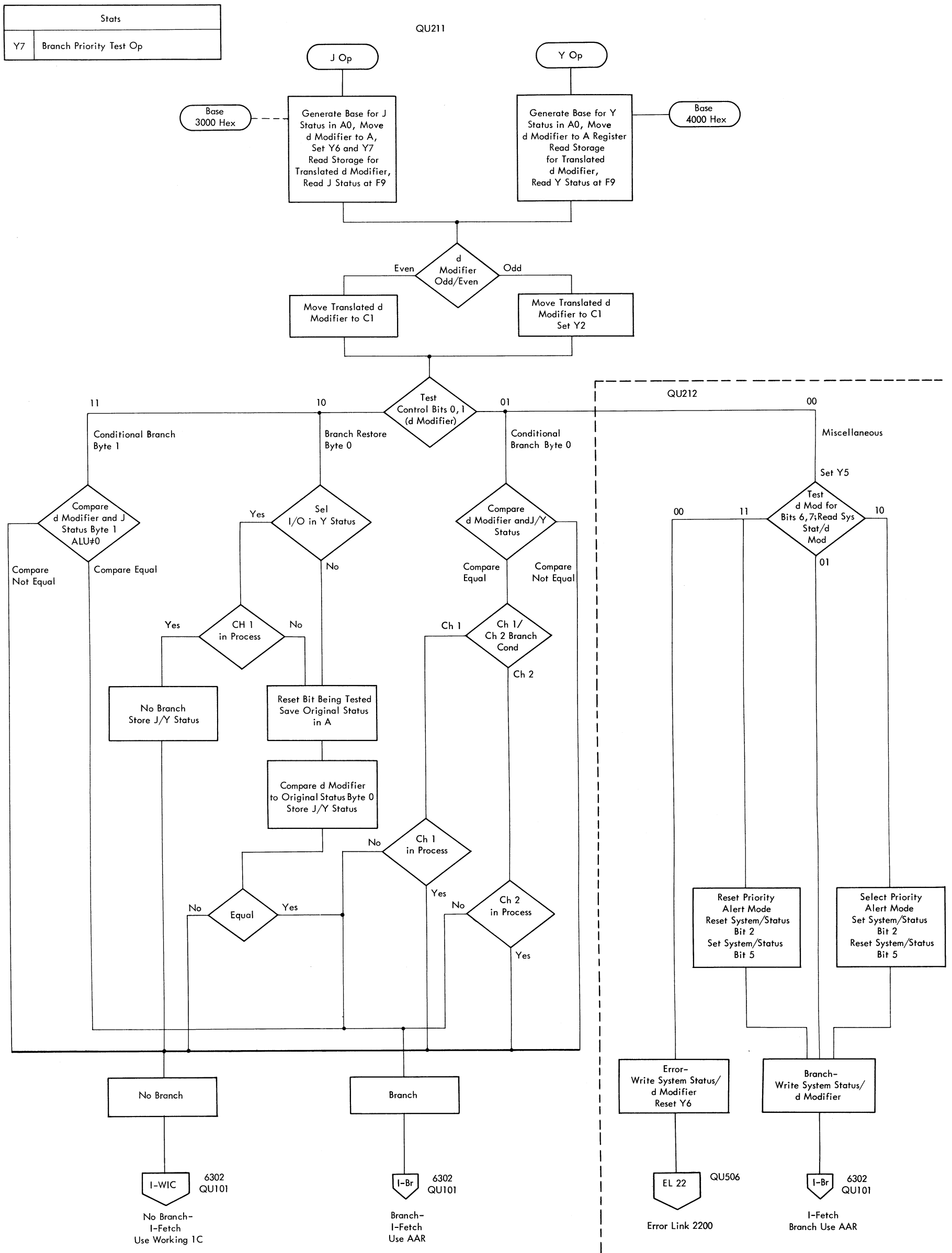


FIGURE 6310. 1410E PRIORITY TEST AND BRANCH, BRANCH ON INTERNAL INDICATOR



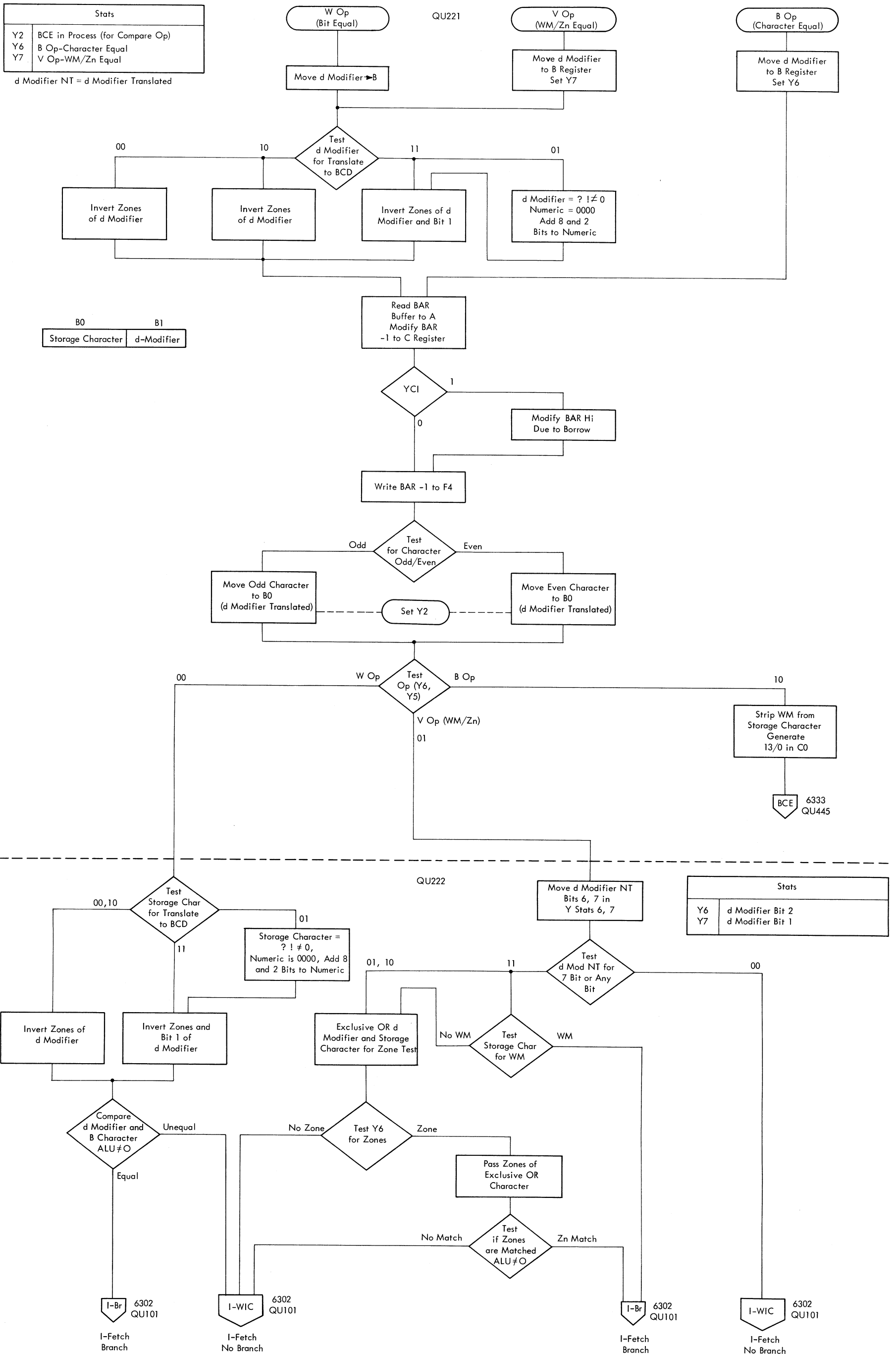


FIGURE 6311. 1410E BRANCH IF: CHARACTER, WM/ZONE, OR BIT EQUAL

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

QU251

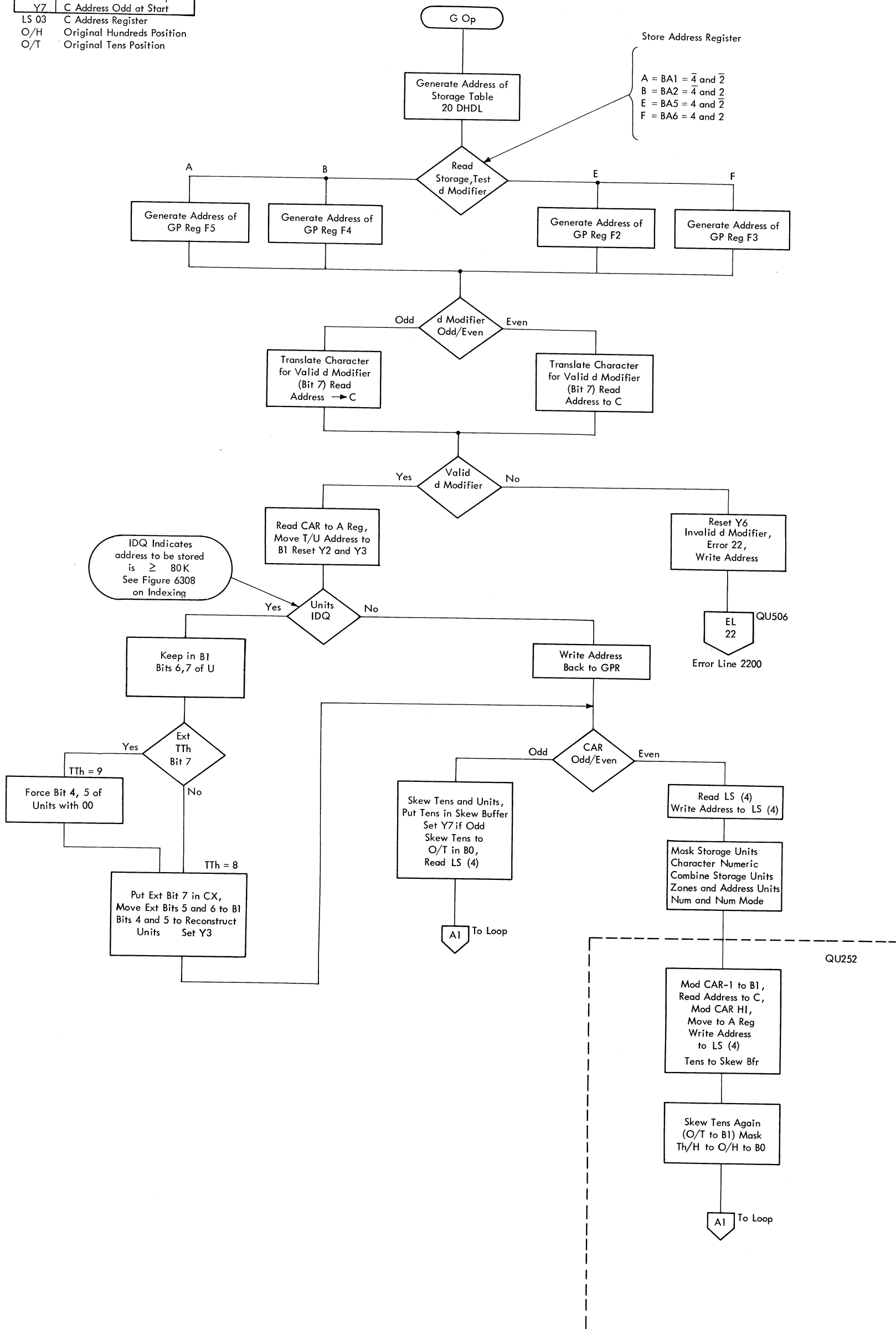


FIGURE 6312. 1410E STORE ADDRESS REGISTER (SHEET 1 OF 2)

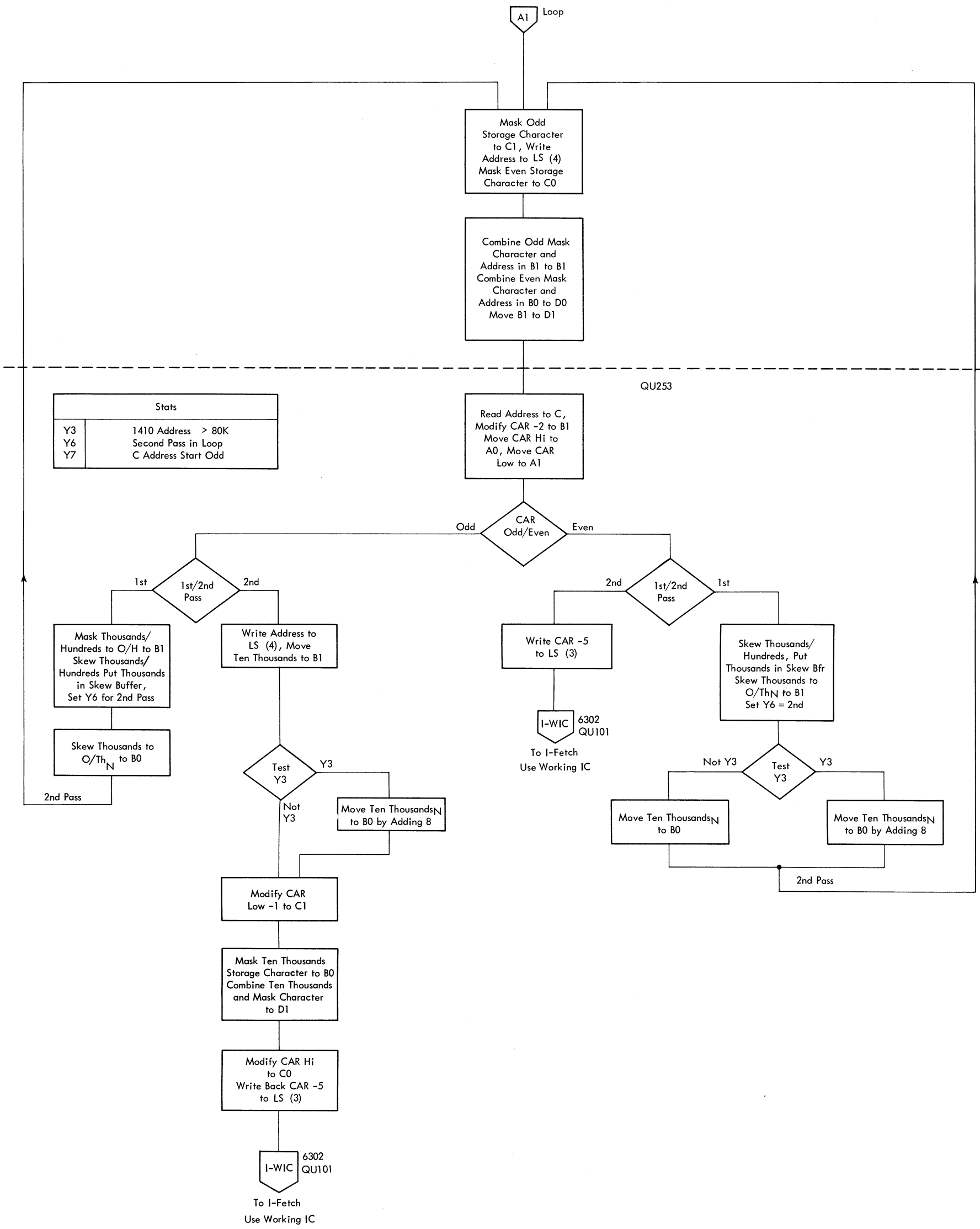
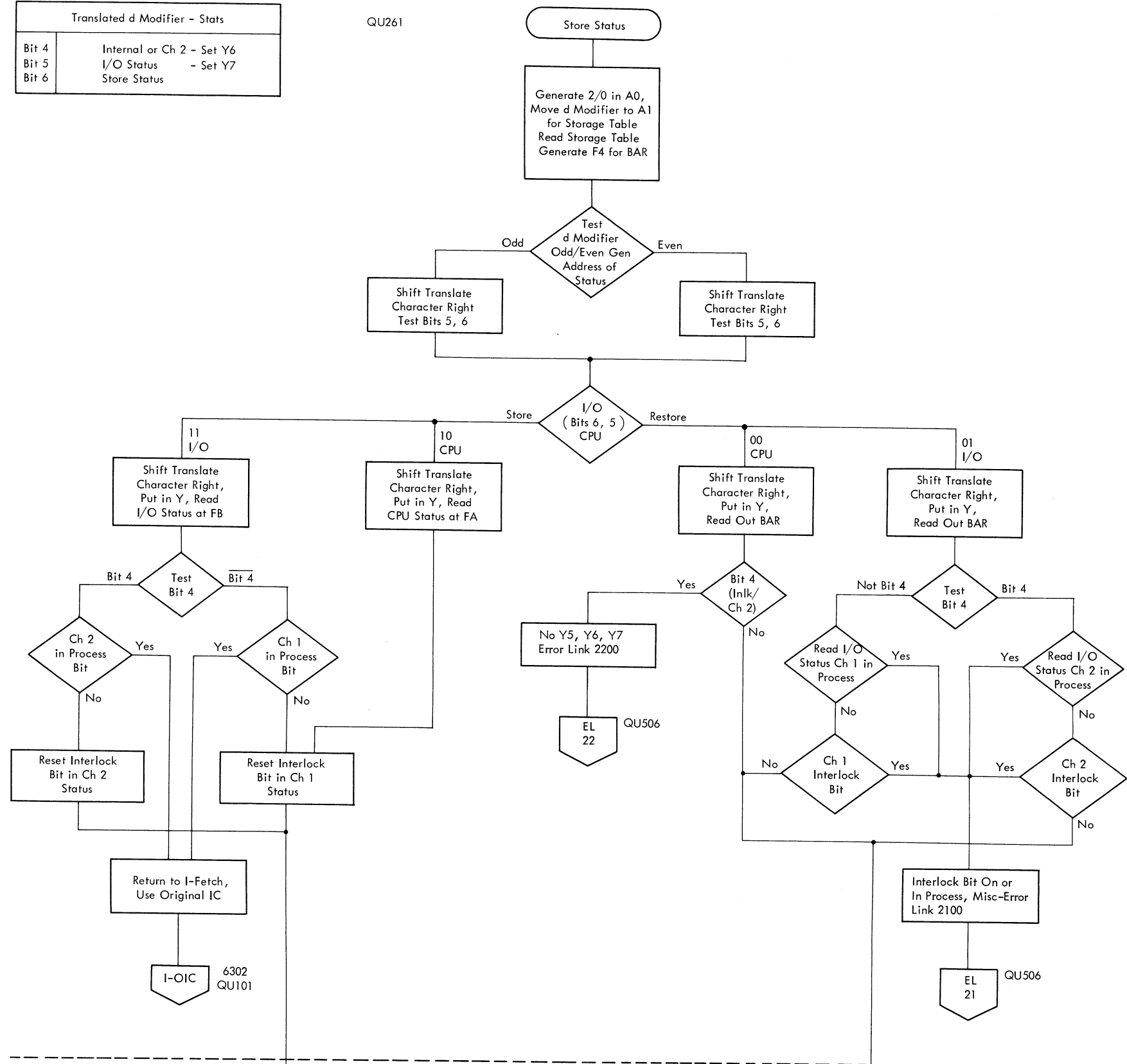


FIGURE 6312. 1410E STORE ADDRESS REGISTER (SHEET 2 OF 2)

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

QU261



QU262

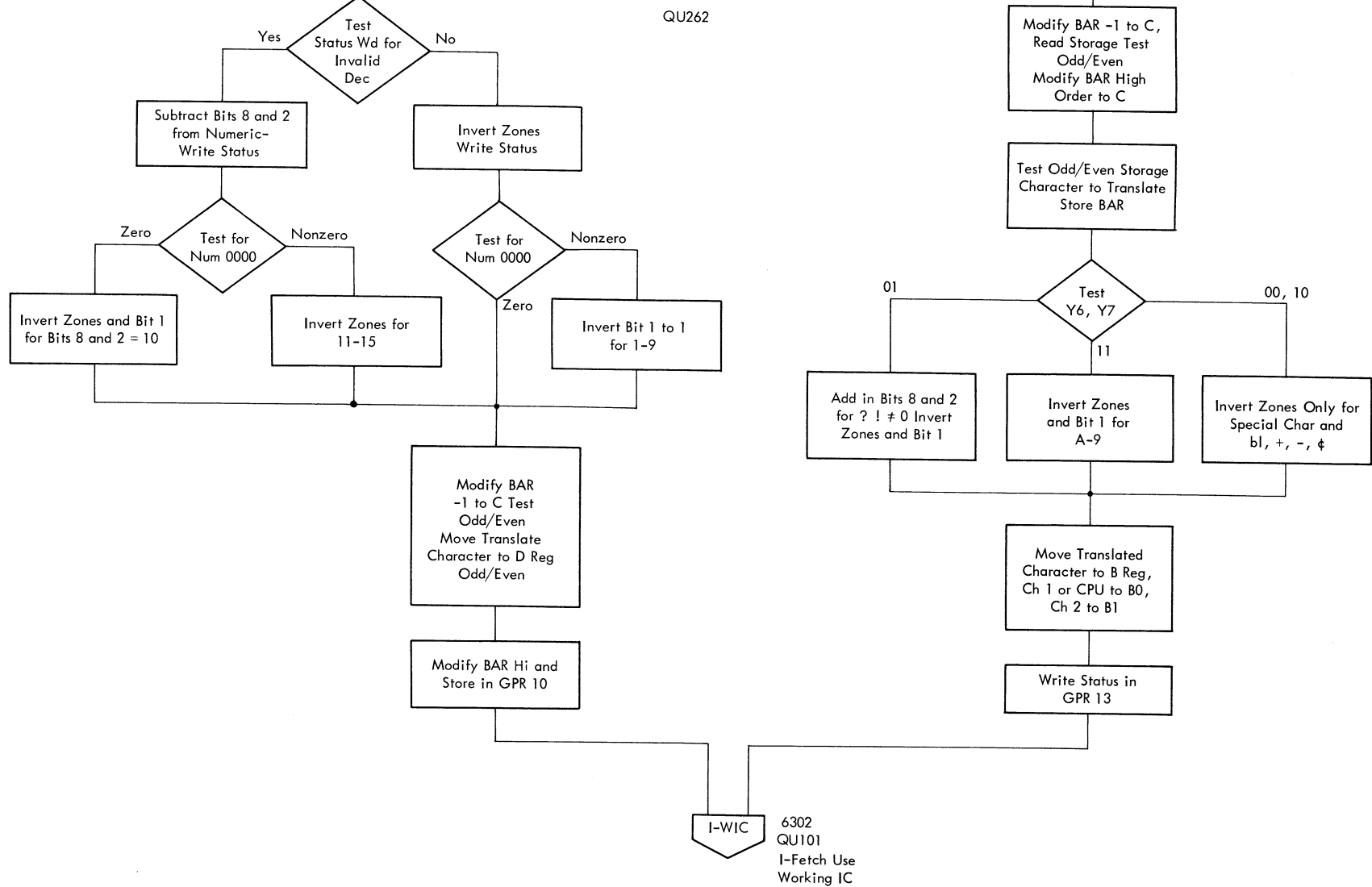


FIGURE 6313. 7010E STORE AND RESTORE STATUS

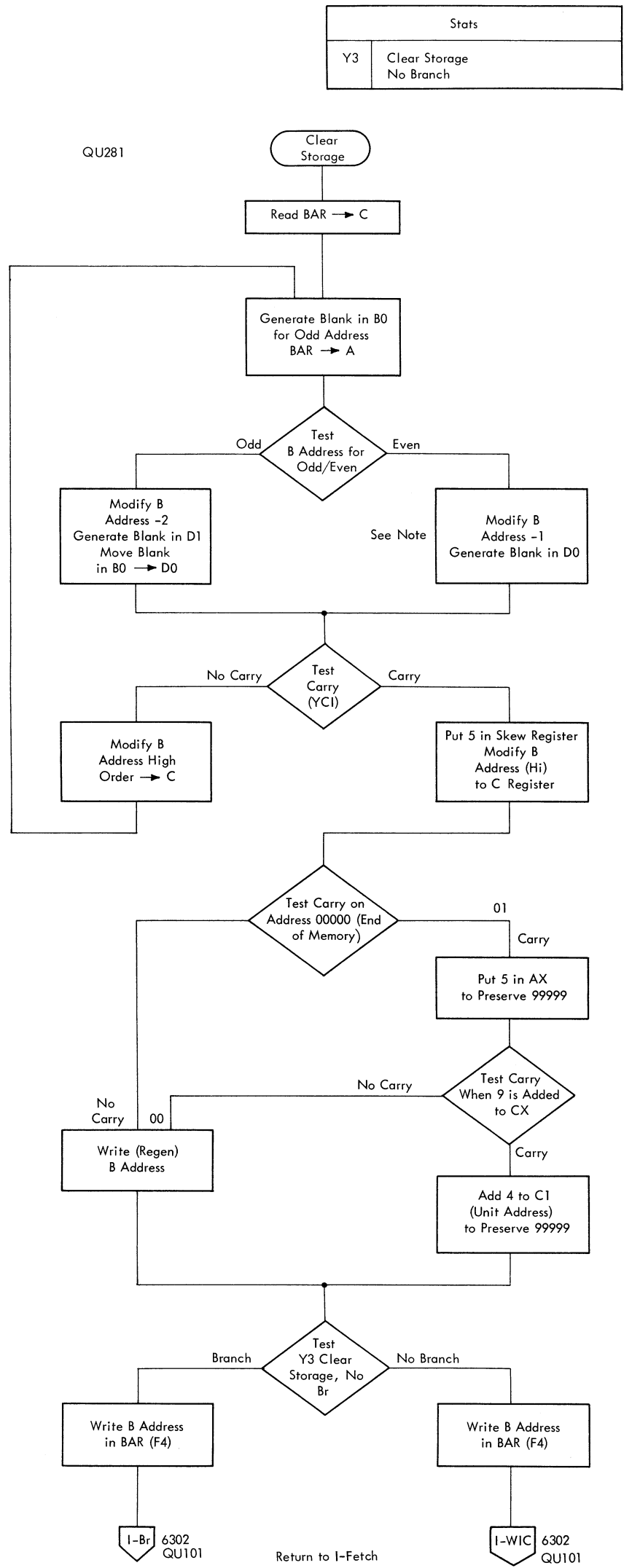
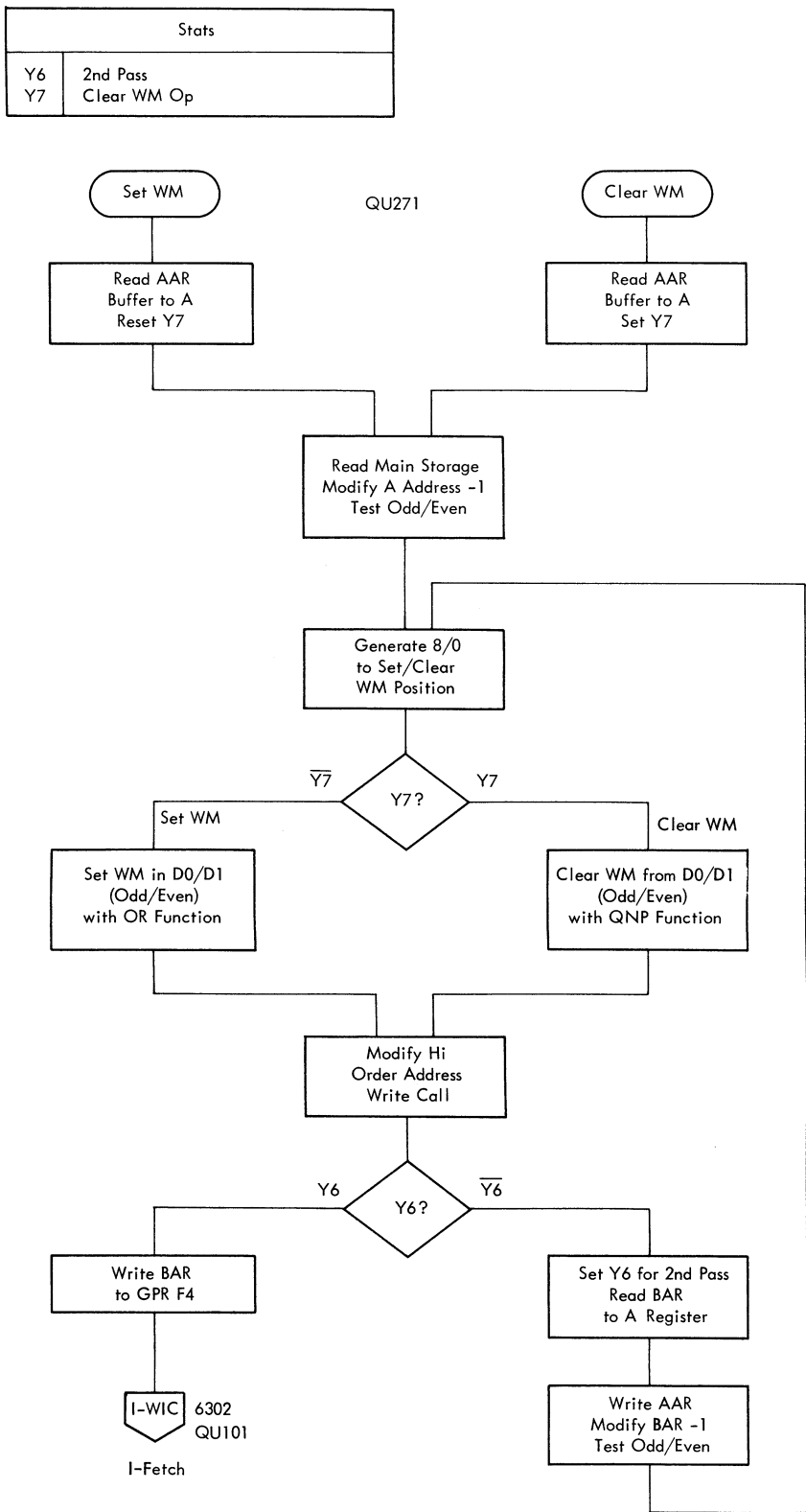


FIGURE 6314. 1410 E SET/CLEAR WM AND CLEAR STORAGE

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  $\leq 79,999$ , addresses  $\geq 80,000$  must be converted. See example on Figure 6308.

Stats	
Y6	Not End of Table

Table Look-up Start

QU291

6333 Table Look-up Exit from Compare (Y4=1)

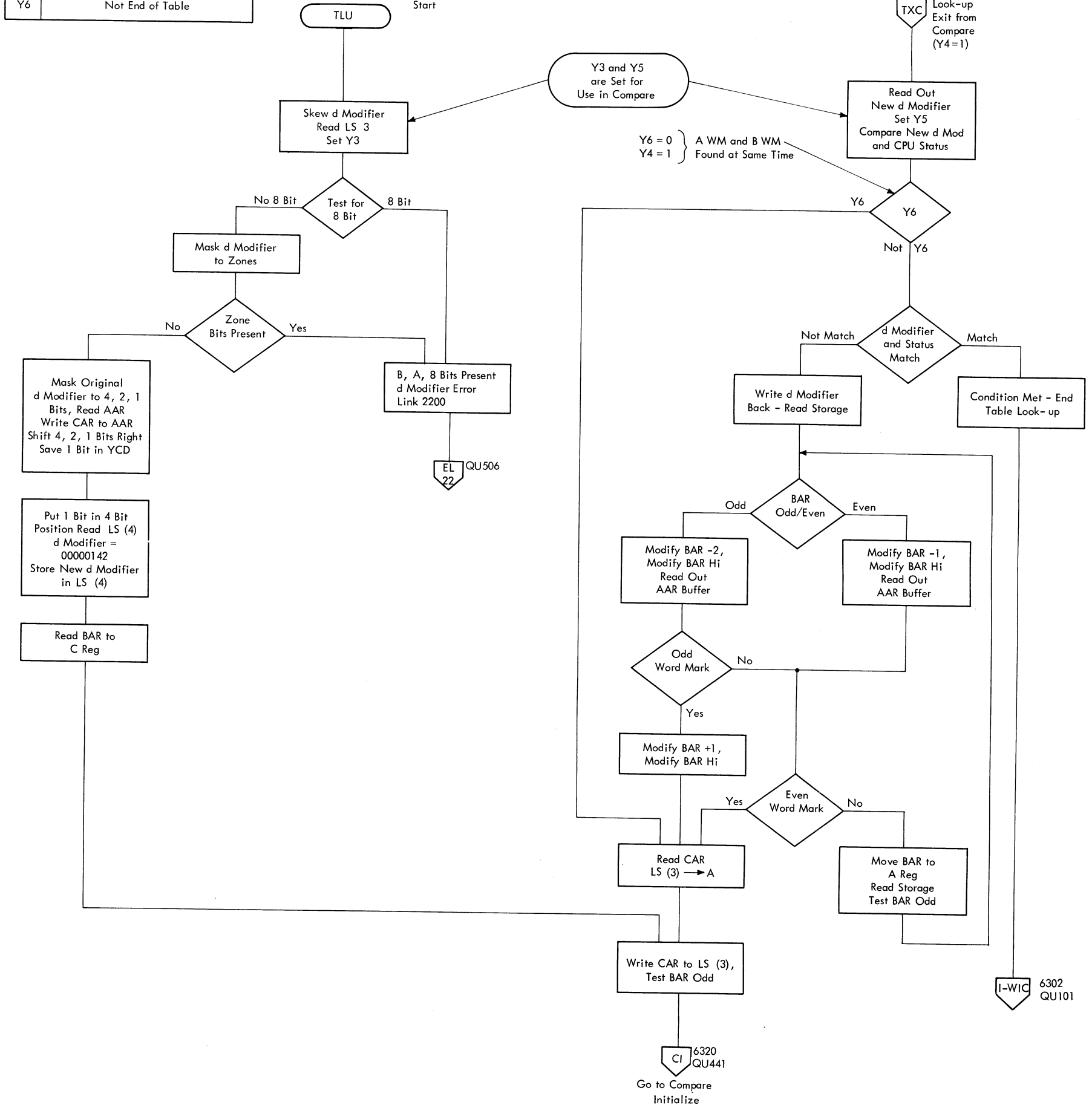
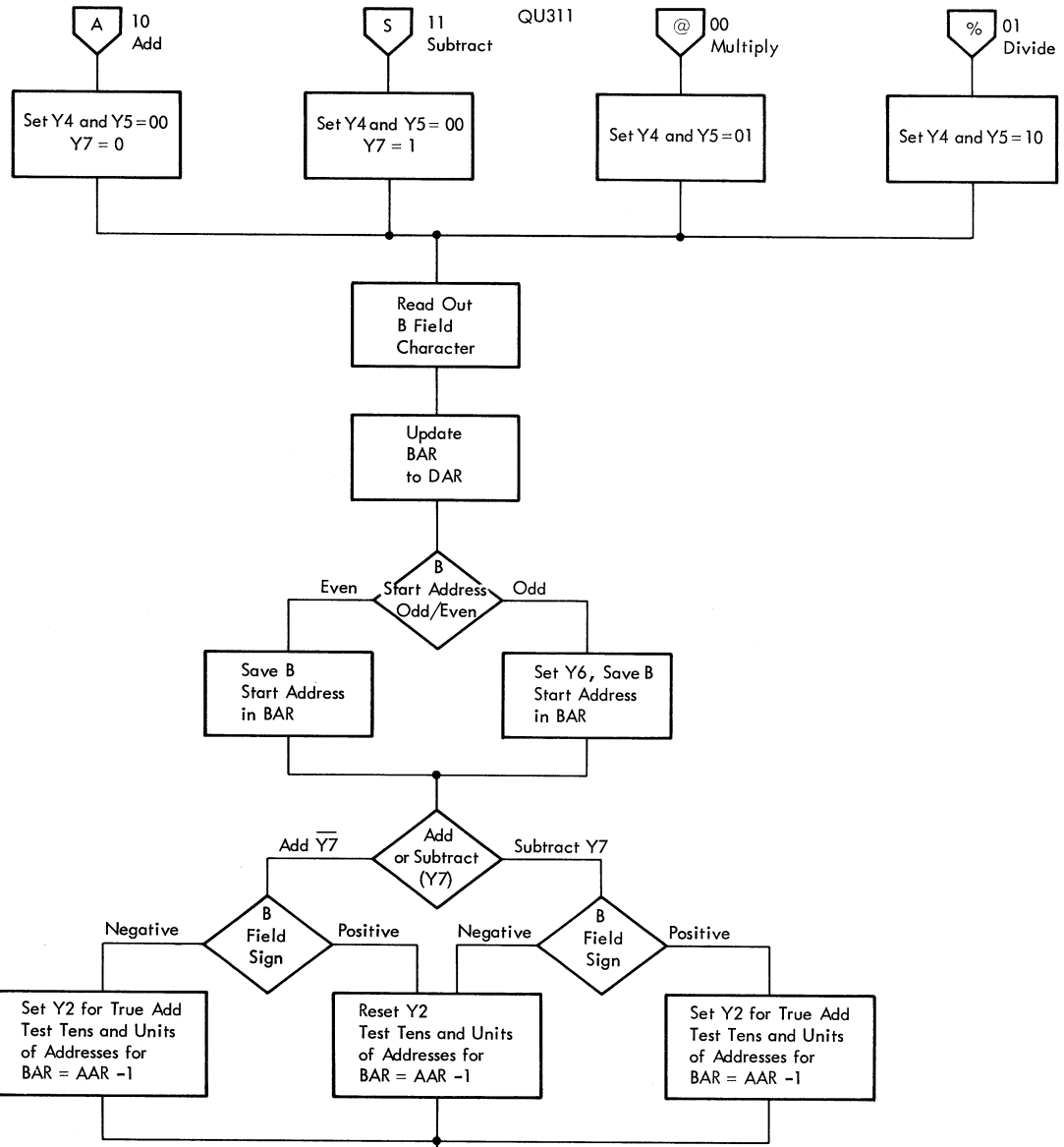


FIGURE 6315. 1410E TABLE LOOKUP

Stats	
Y4 and Y5	00 for @ 01 for % 10 for A 11 for S
Y6	B Field Start Addr Odd
Y7	Subtract



Stats	
Y2	A Field Positive (@,%) True Add for A/S
Y3	Skew Case 00 = A/S
Y4	01 = @
Y5	10 = Invalid Exit 11 = %
Y6	B Field Start Addr Odd
Y7	Overlap Field Case (BAR = AAR -1)

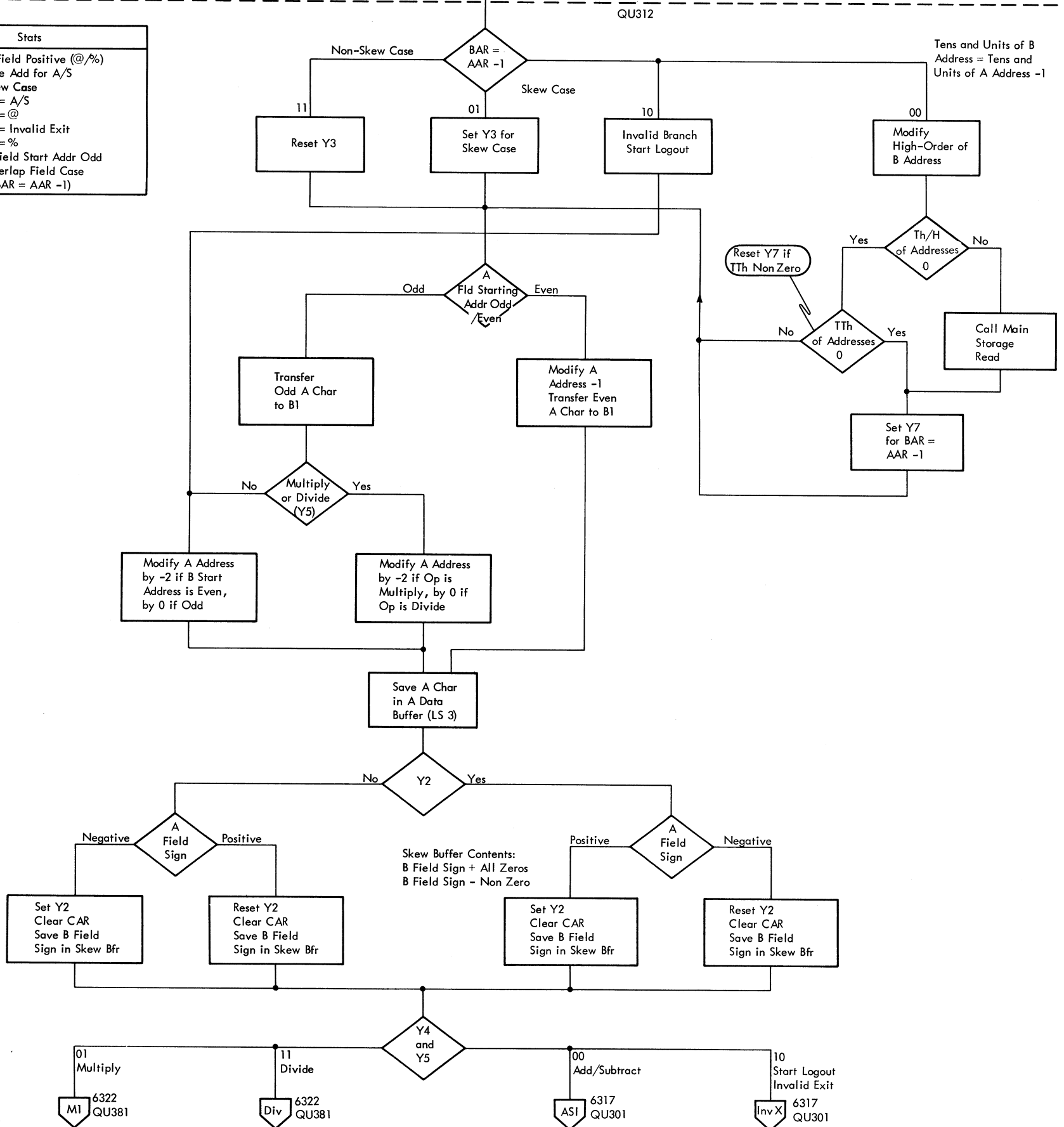


FIGURE 6316. 1410E ADD, SUBTRACT, MULTIPLY, DIVIDE

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)

QU301

Inv X From Invalid Exit, Common Routine Start Logout 6316

ASI From Add/Subtract Entry 6316

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.

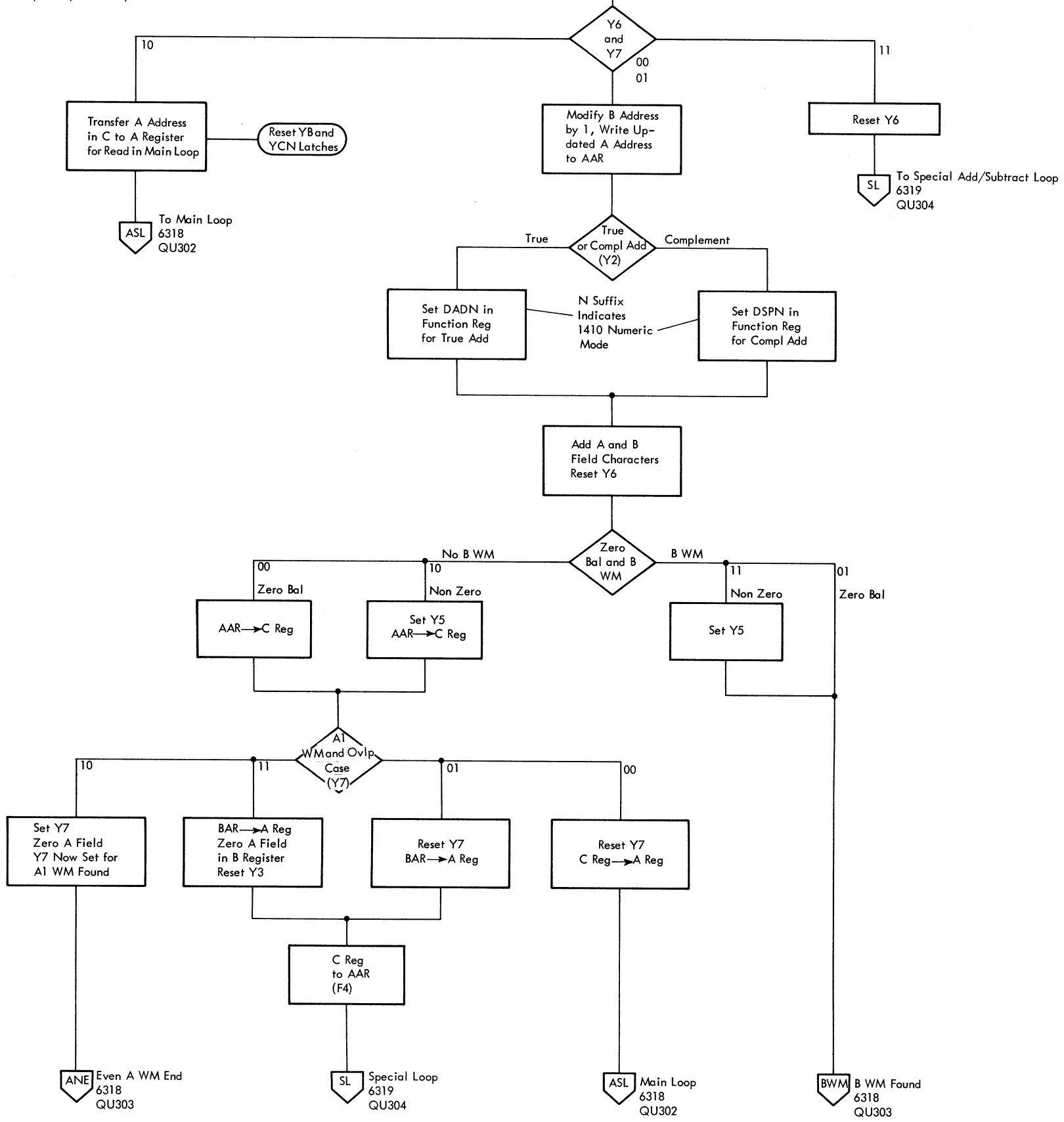
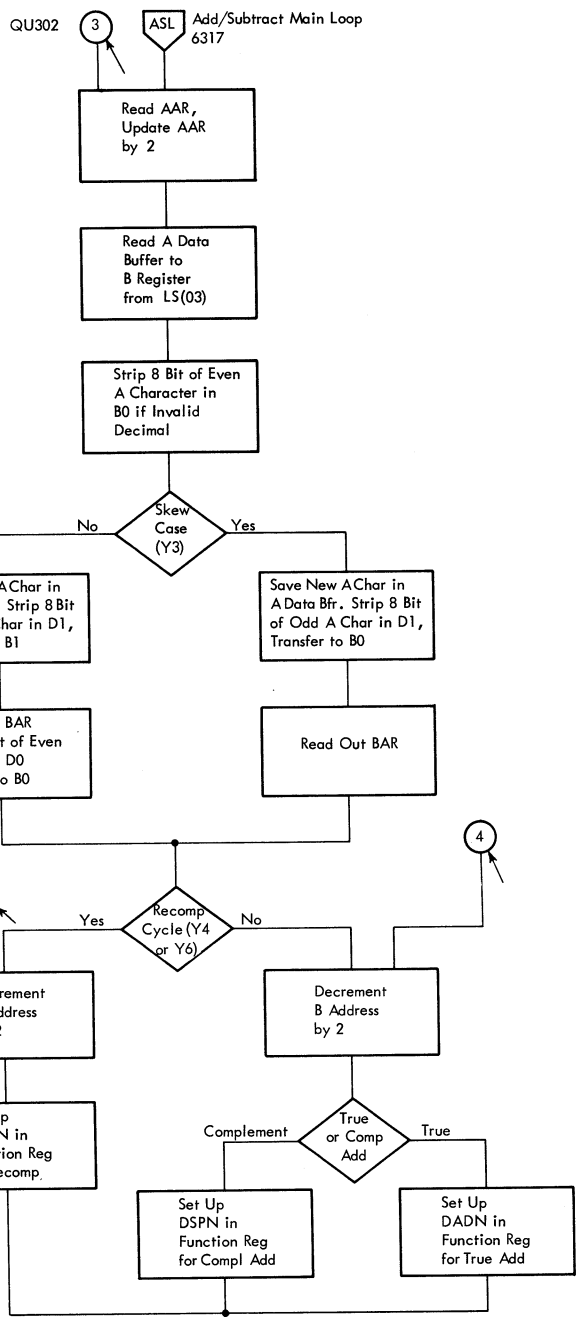


FIGURE 6317. 1410E ADD, SUBTRACT-INITIAL LOOP



Stats	
Y2	True Add
Y3	Skew Case
Y4	Recomplement Cycle
Y5	Numeric Result Non Zero
Y7	A Field WM Found
Y6	Recomplement Cycle



QU303

Stats	
Y3	Skew Case
Y4	Recomplement Cycle
Y5	Not Zero Balance
Y6	Recomplement Cycle
Y7	A Address End Modification Done

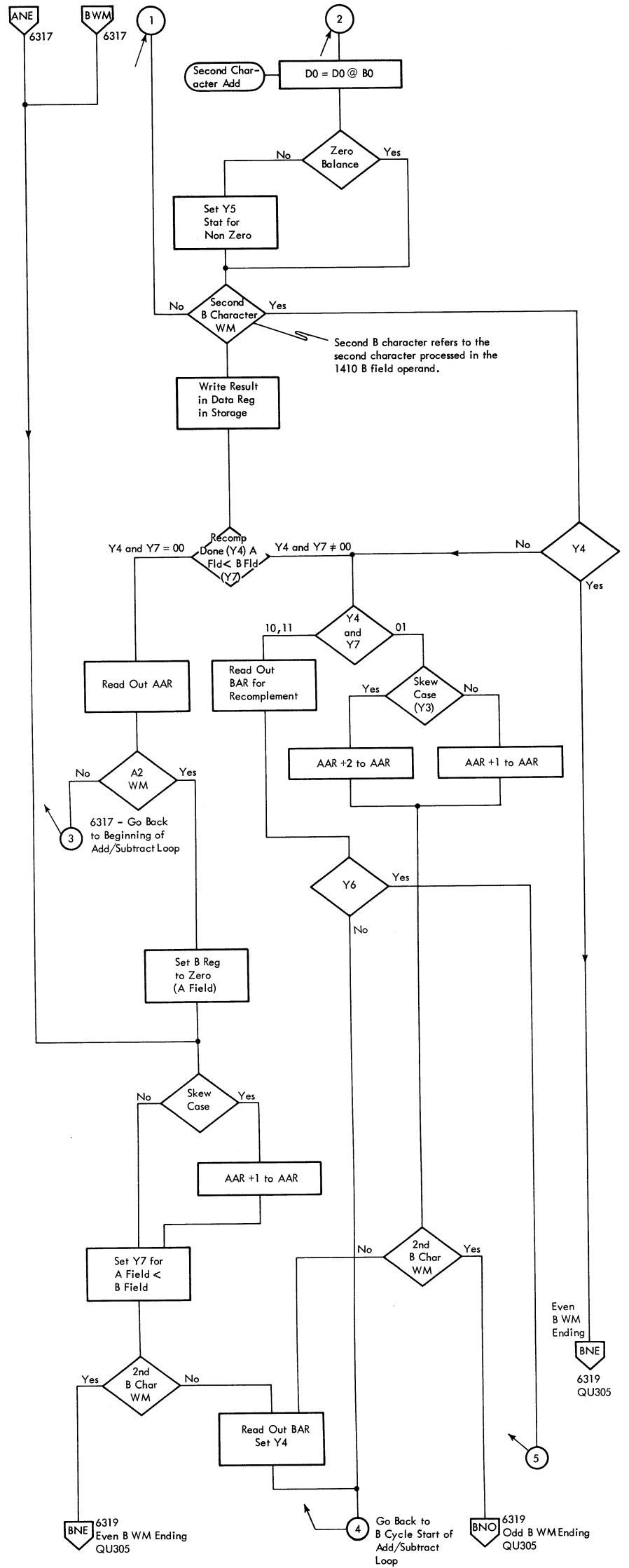
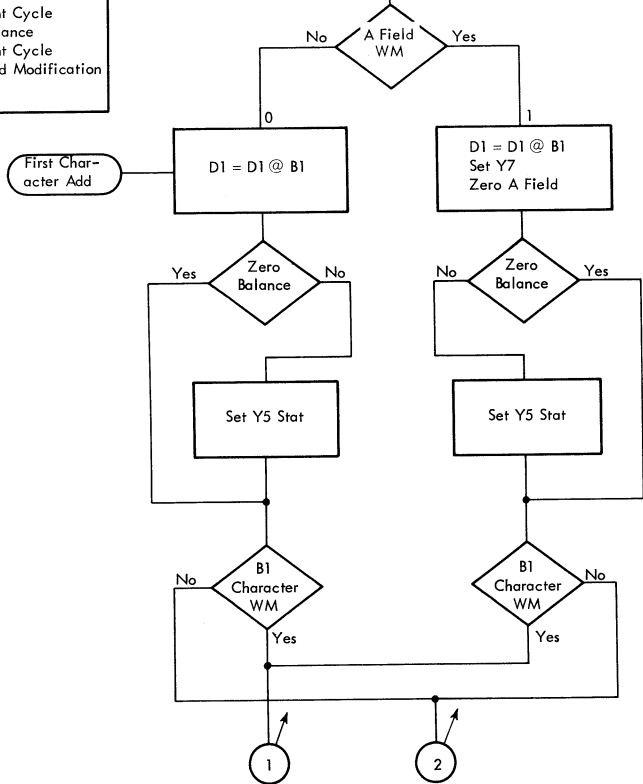


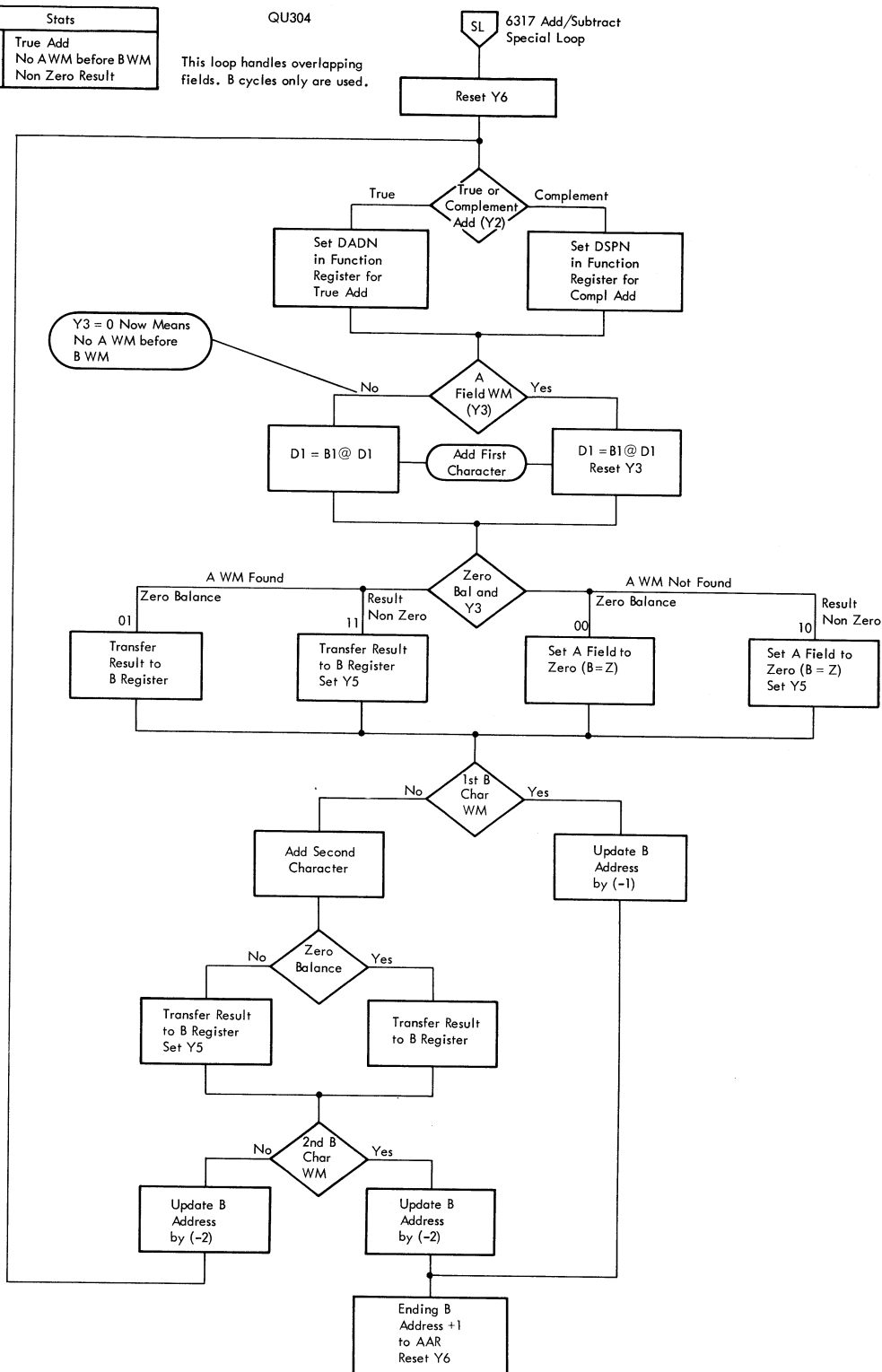
FIGURE 6318. 1410E ADD, SUBTRACT-MAIN LOOP

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

QU304

6317 Add/Subtract Special Loop

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



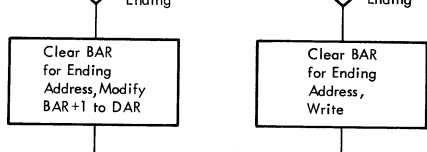
Stats	
Y2	True Add
Y3	Skew
Y4	Result has been recomplemented
Y5	Num Result Non Zero
Y6	Recomplement Done
Y7	A Field WM Found

MSD = Most Significant Digit  
ZBI = Zero Balance Indicator

QU305

6318 Odd B WM Ending

6318 Even B WM Ending



QU306

Set CPU Indicator Routine

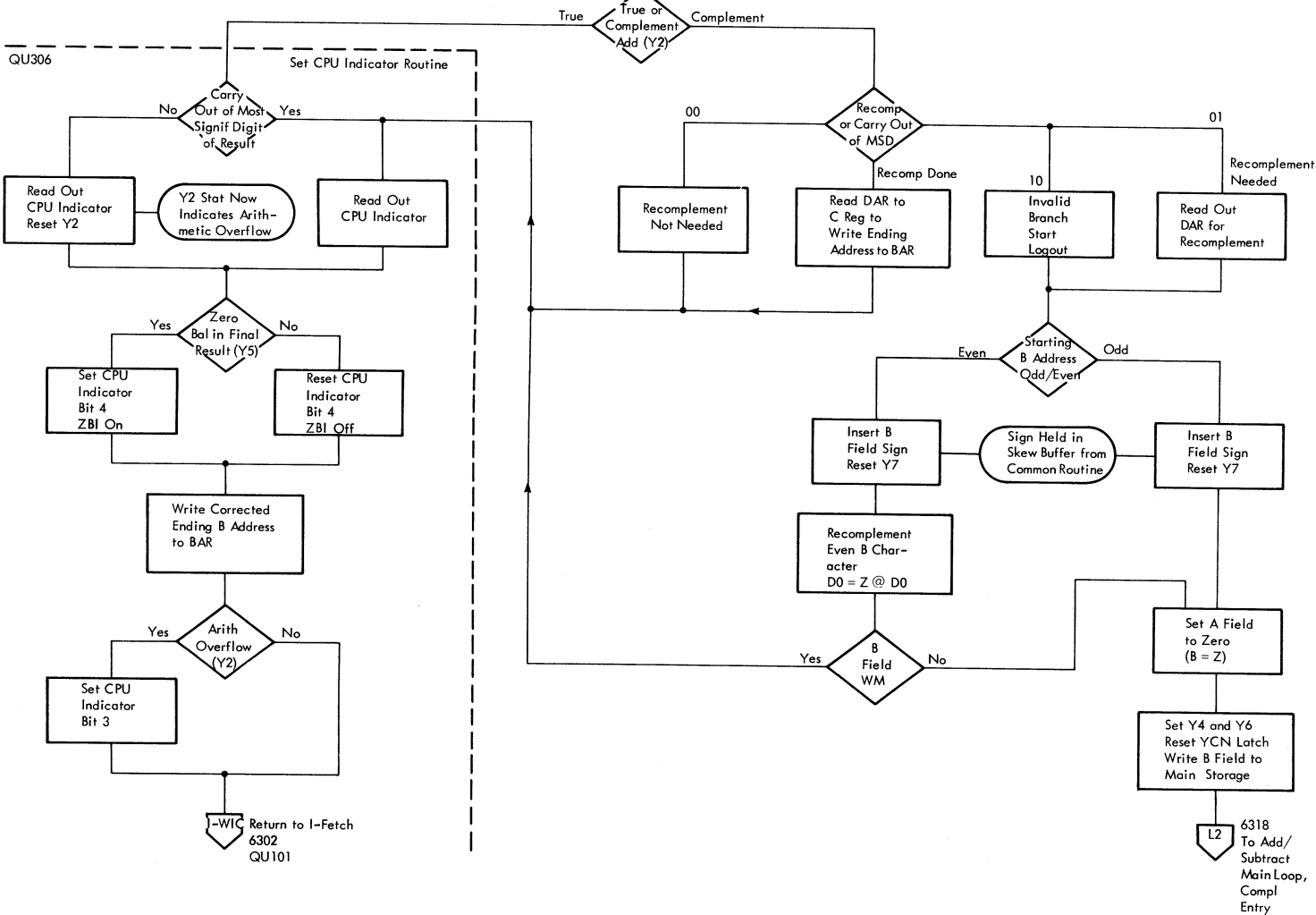


FIGURE 6319. 1401E ADD, SUBTRACT-SPECIAL LOOP AND ENDING

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

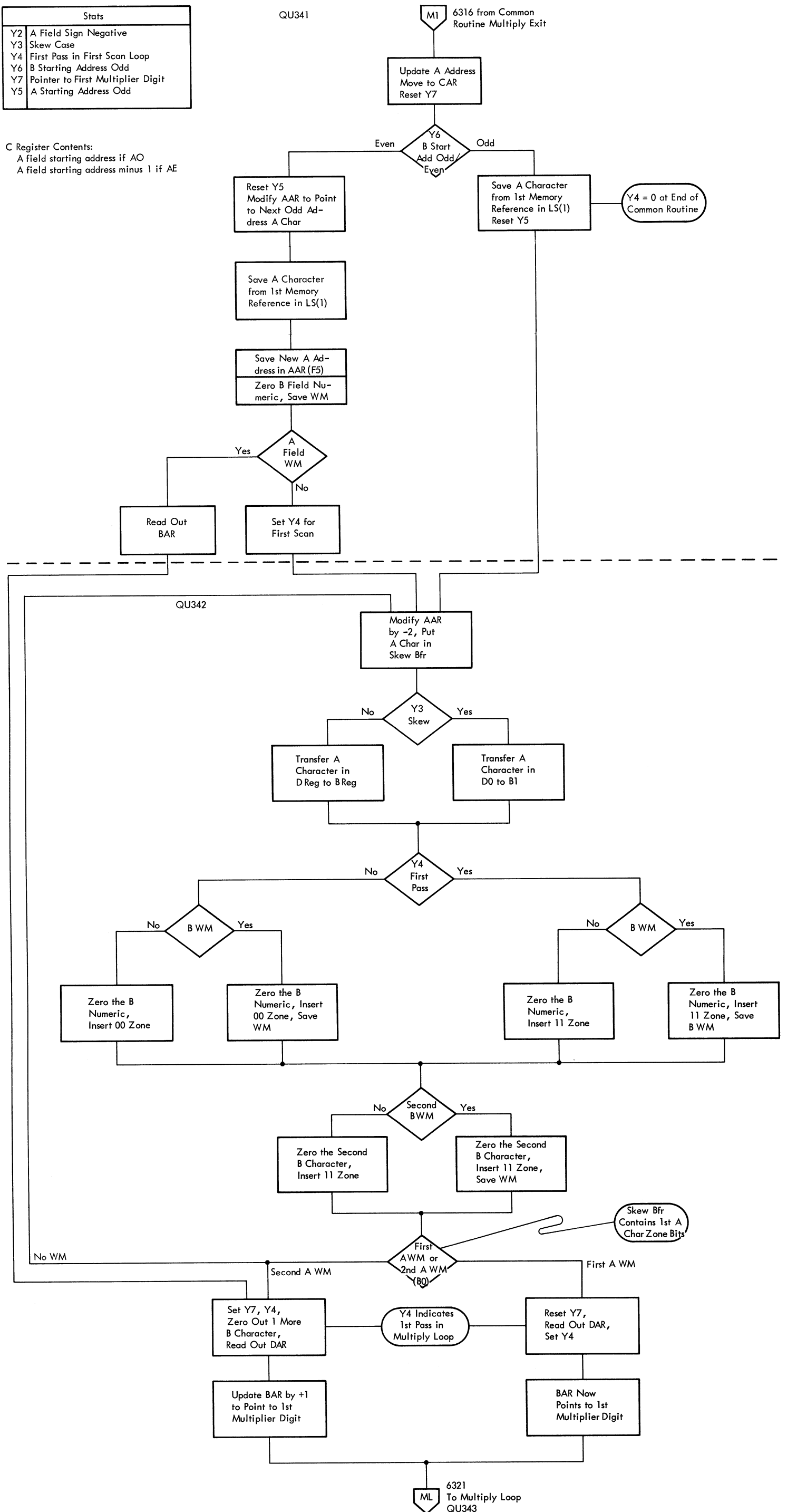
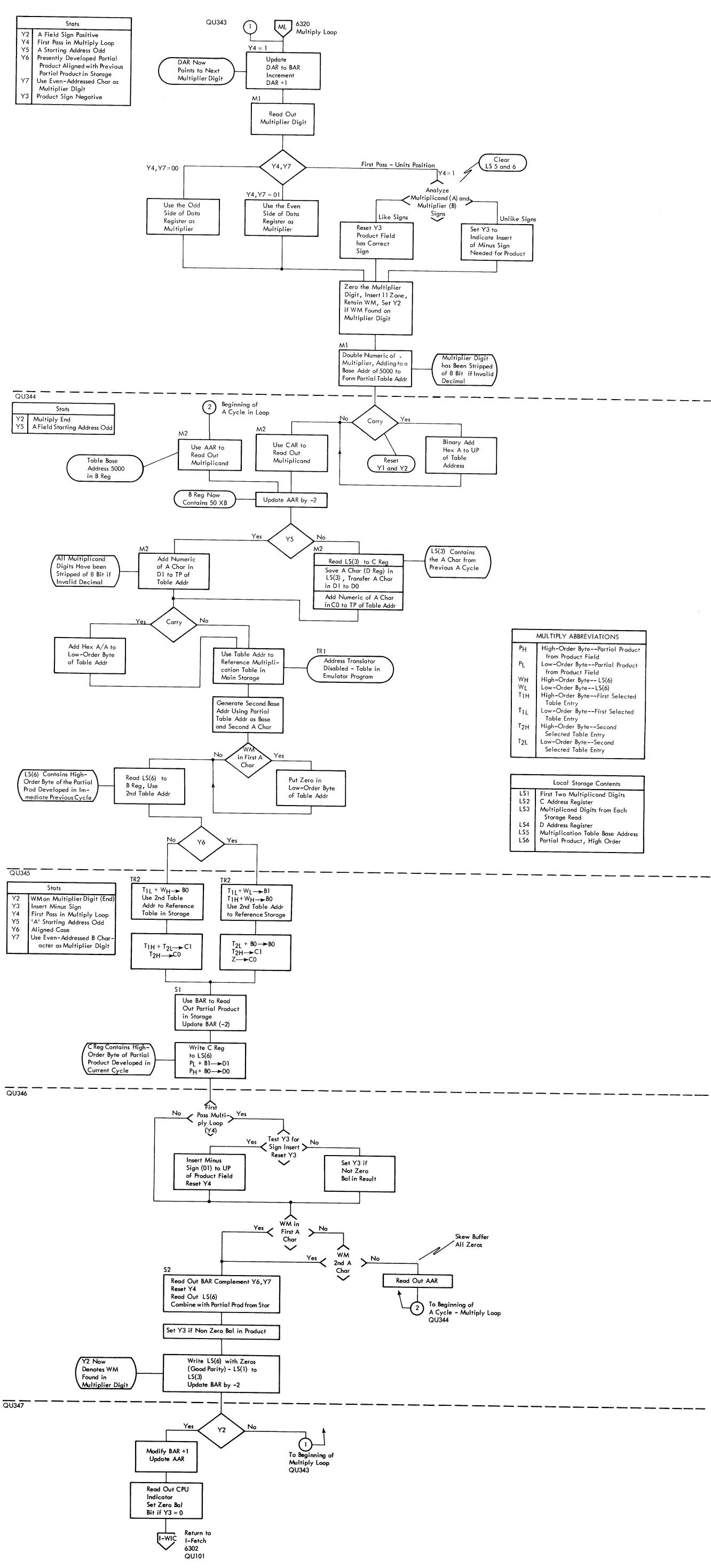


FIGURE 6320. 1410E MULTIPLY FIRST SCAN



MULTIPLY ABBREVIATIONS	
PH	High-Order Byte--Partial Product from Product Field
PL	Low-Order Byte--Partial Product from Product Field
WH	High-Order Byte--LS(6)
WL	Low-Order Byte--LS(6)
T1H	High-Order Byte--First Selected Table Entry
T1L	Low-Order Byte--First Selected Table Entry
T2H	High-Order Byte--Second Selected Table Entry
T2L	Low-Order Byte--Second Selected Table Entry

Local Storage Contents	
LS1	First Two Multiplicand Digits
LS2	C Address Register
LS3	Multiplicand Digits from Each Storage Read
LS4	D Address Register
LS5	Multiplication Table Base Address
LS6	Partial Product, High Order

FIGURE 6321. MULTIPLY LOOP

Stats	
Y2	MDL (Divide End)
Y3	Skew Case
Y4	Complement Add Cycle
Y5	Look for B Bit On B Char to Set MDL Latch
Y6	B Starting Addr Odd
Y7	WM Found on Even A Char

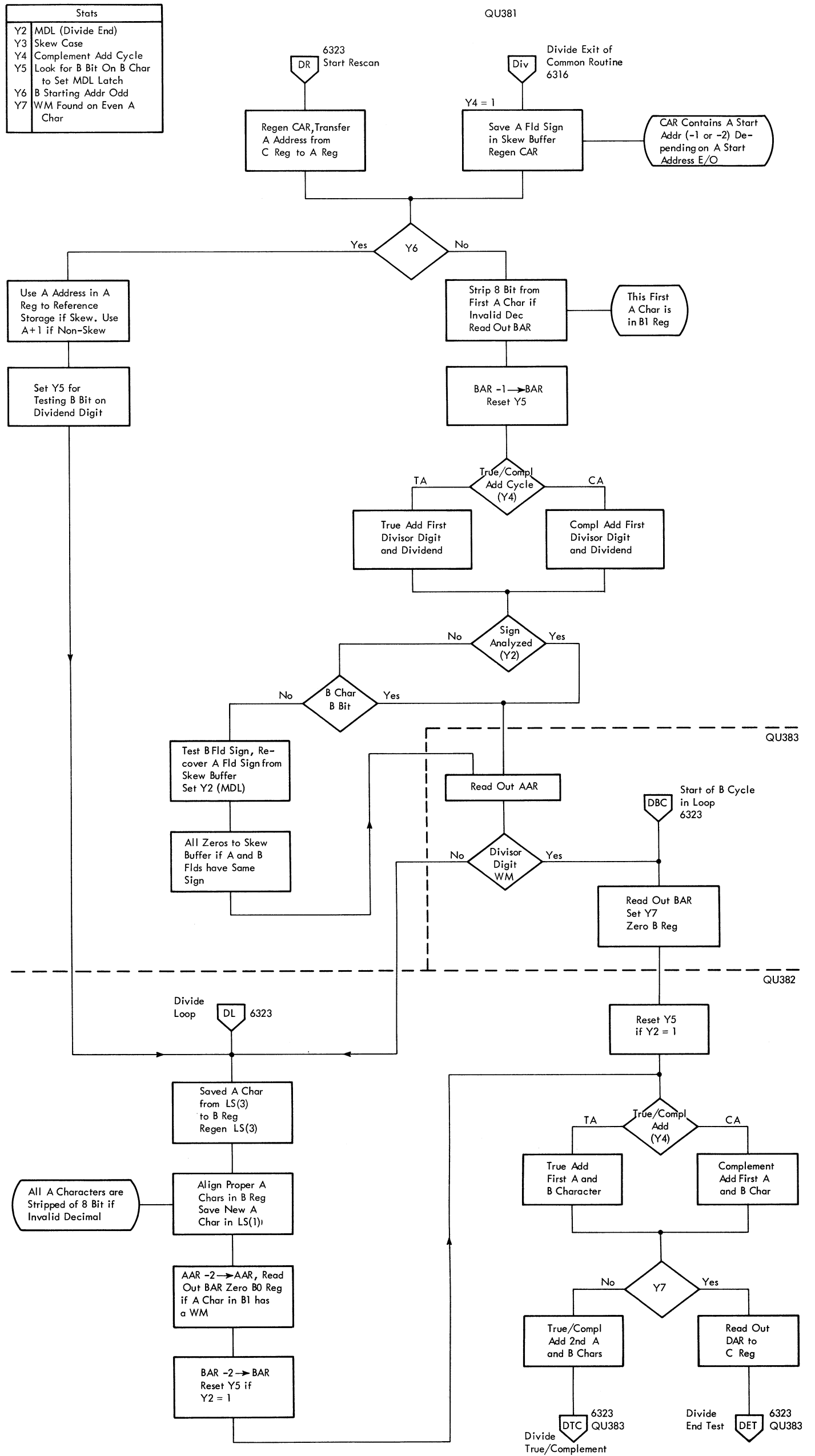


FIGURE 6322. 1410E DIVIDE INITIAL LOOP

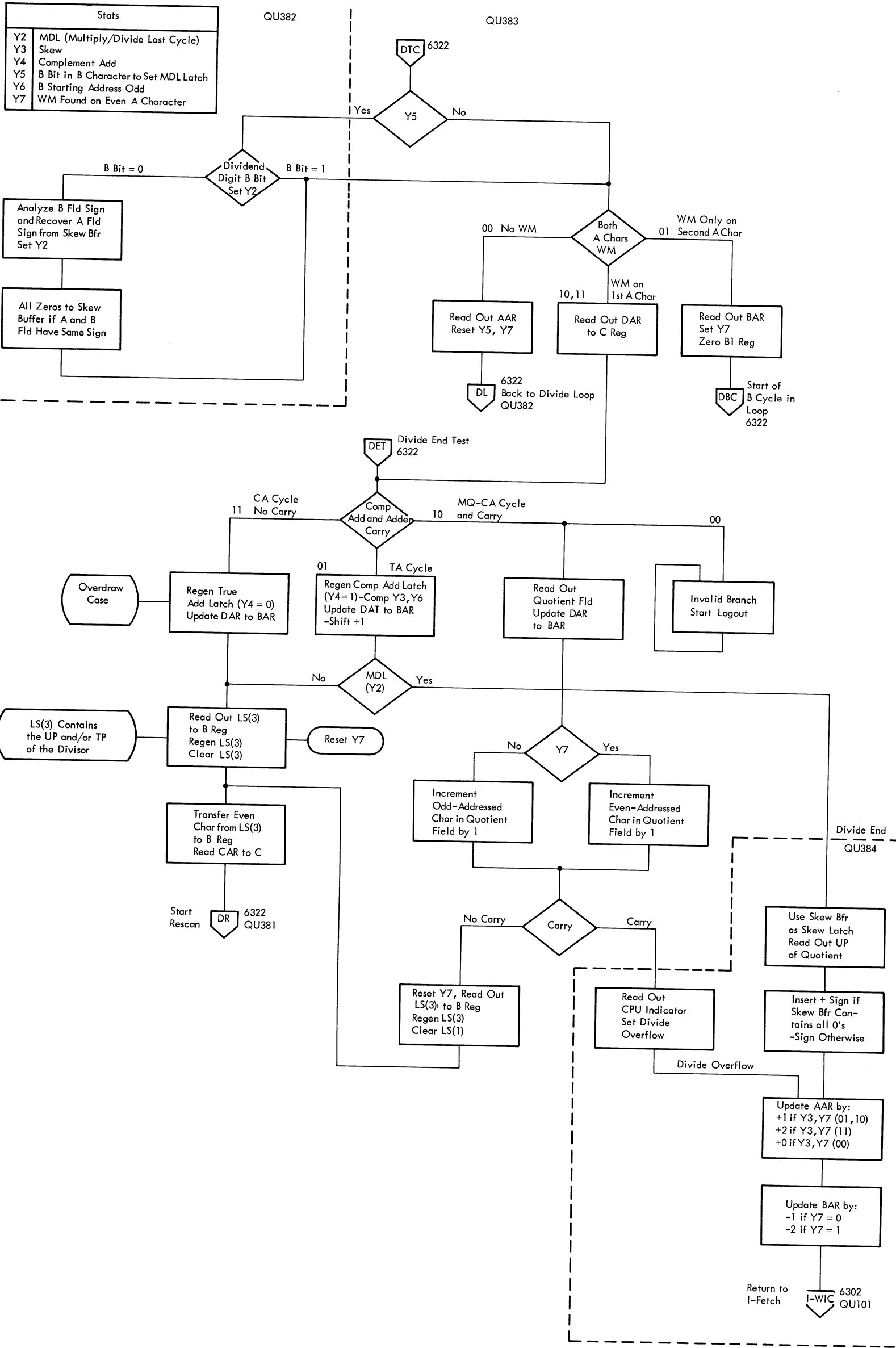


FIGURE 6323. 1410E DIVIDE LOOP AND ENDING

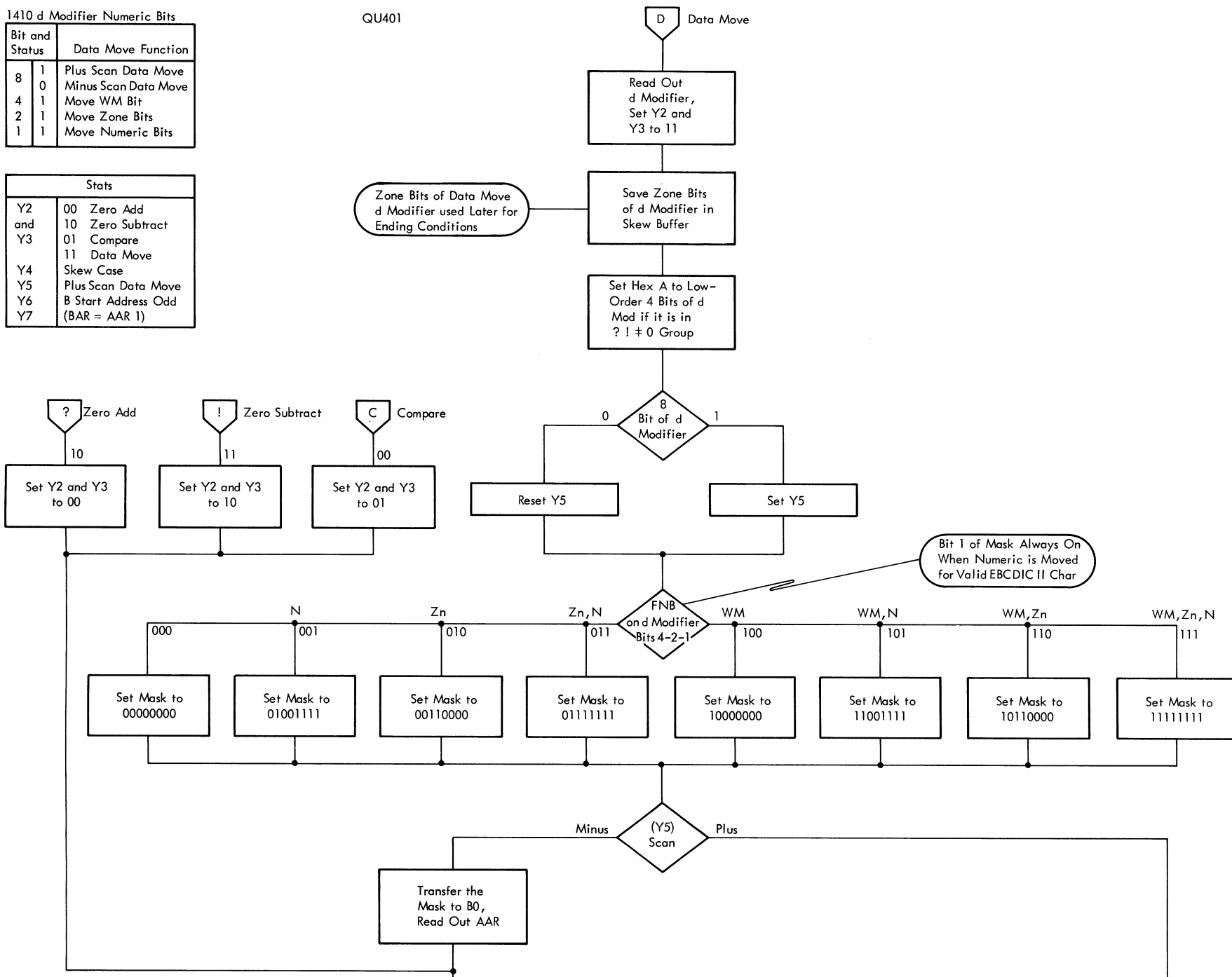
1410 d Modifier Numeric Bits

Bit and Status	Data Move Function
8 1	Plus Scan Data Move
8 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 (BAR = AAR - 1)
Y7	

QU401

D Data Move



QU402

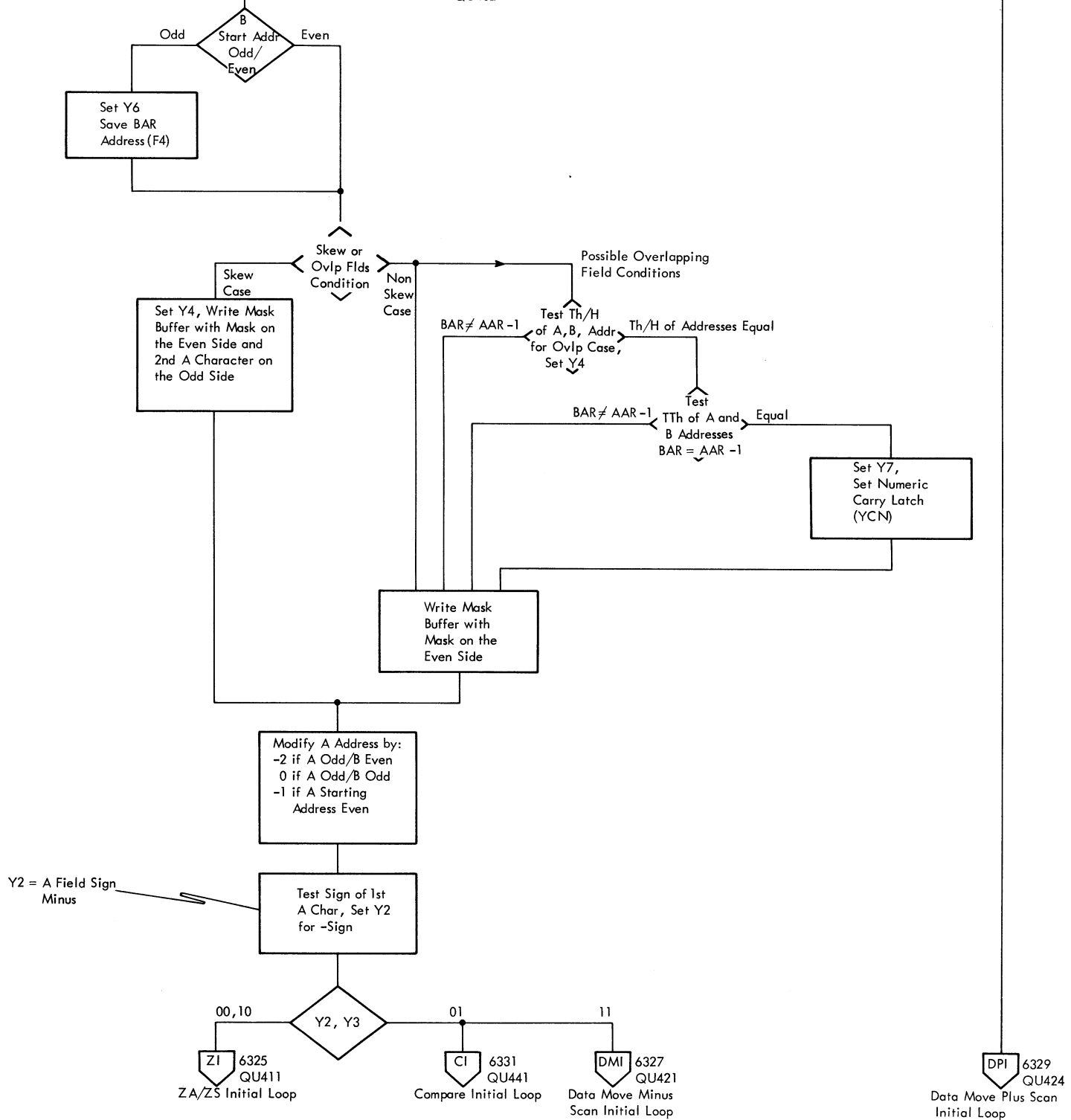


FIGURE 6324. 1410E DATA MOVE, ZERO ADD/SUBTRACT, AND COMPARE

Stats	
Y2	A Field Sign Negative
Y3	Zero Add/Subtract
Y4	Skew Case
Y5	AAR End Address, Modification Completed after WM is Found
Y6	Starting B Address Odd
Y7	Ovlp Field Case (BAR = AAR - 1)

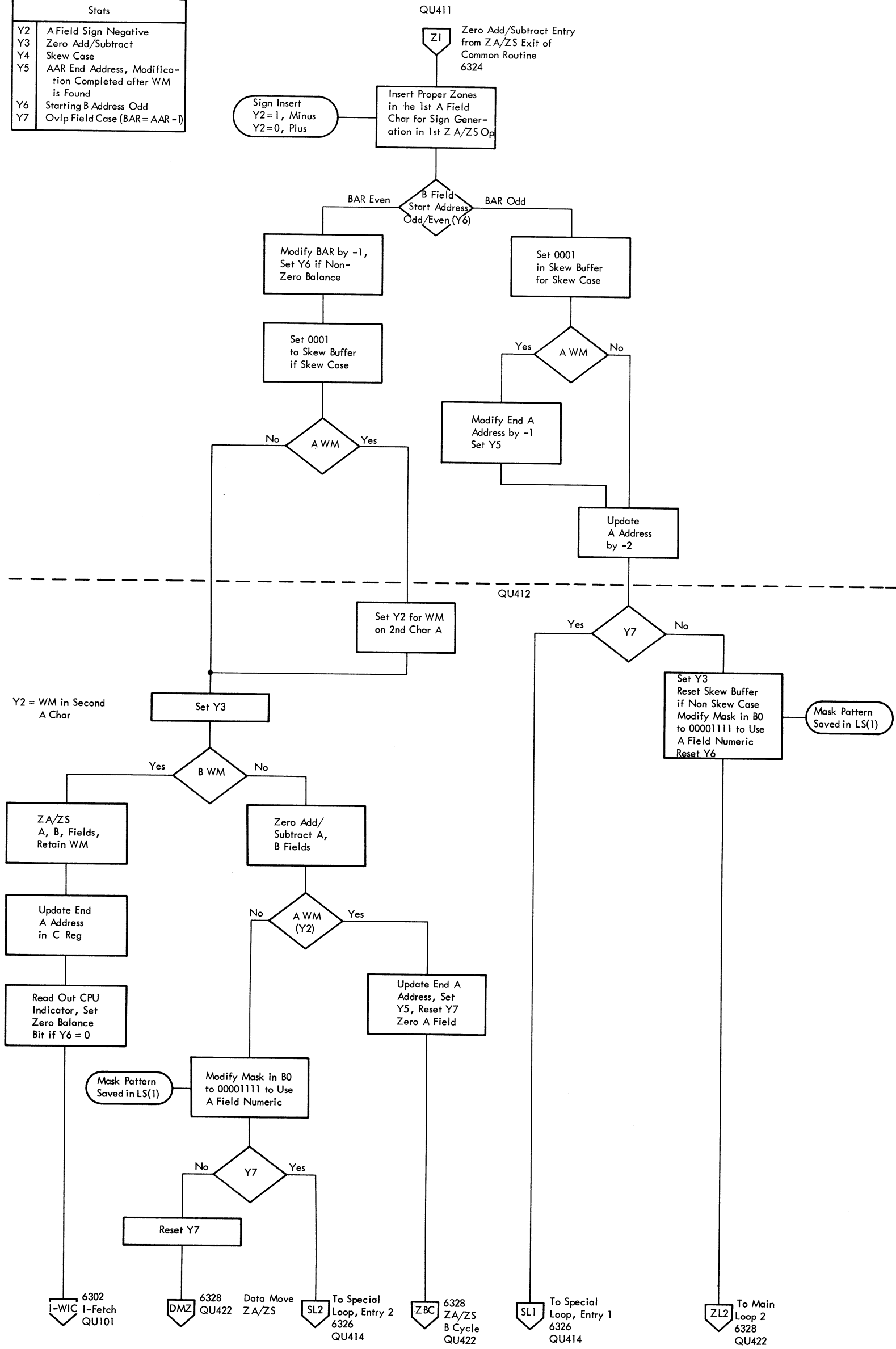


FIGURE 6325. 1410E ZERO ADD/SUBTRACT--INITIAL LOOP



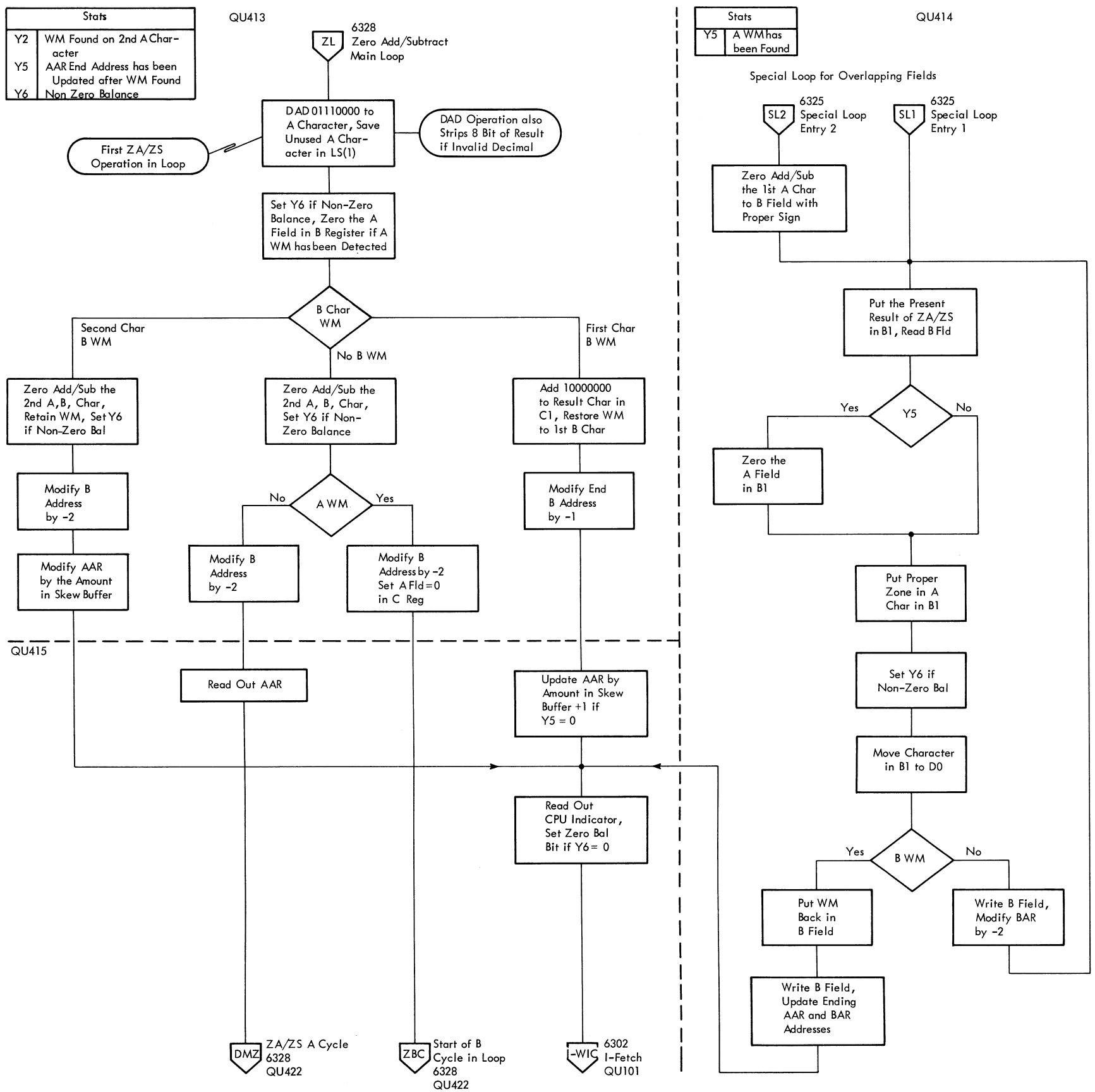


FIGURE 6326. 1410E, ZERO ADD/SUBTRACT MAIN AND SPECIAL LOOP

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

QU421

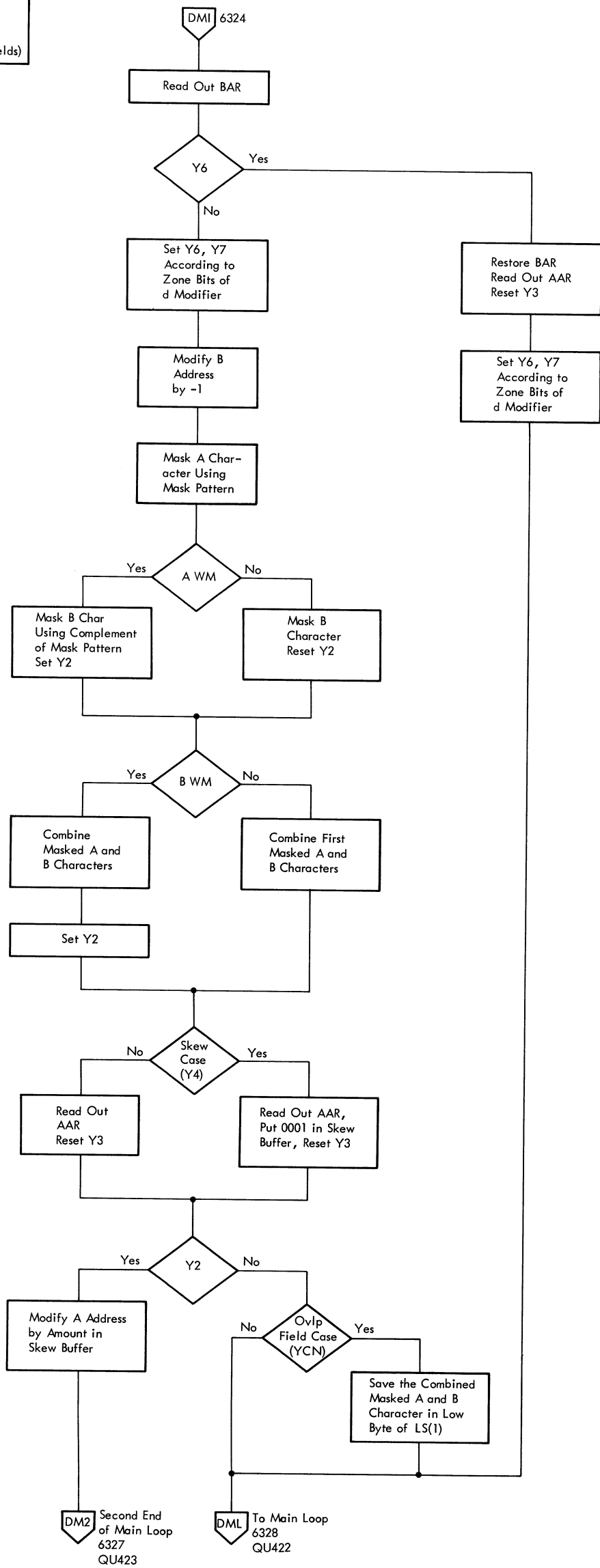


FIGURE 6327. 1410E DATA MOVE MINUS SCAN-INITIAL LOOP

Stats	
Y2	Terminate after Processing First Character
Y3	Zero Add/Sub
Y4	Skew Case
Y5	Terminate after Processing Second Character
Y6	00 - Move to A WM or B WM
Y7	01 - Move to B WM
Y7	10 - Move to A WM
Y7	11 - Move Single Position

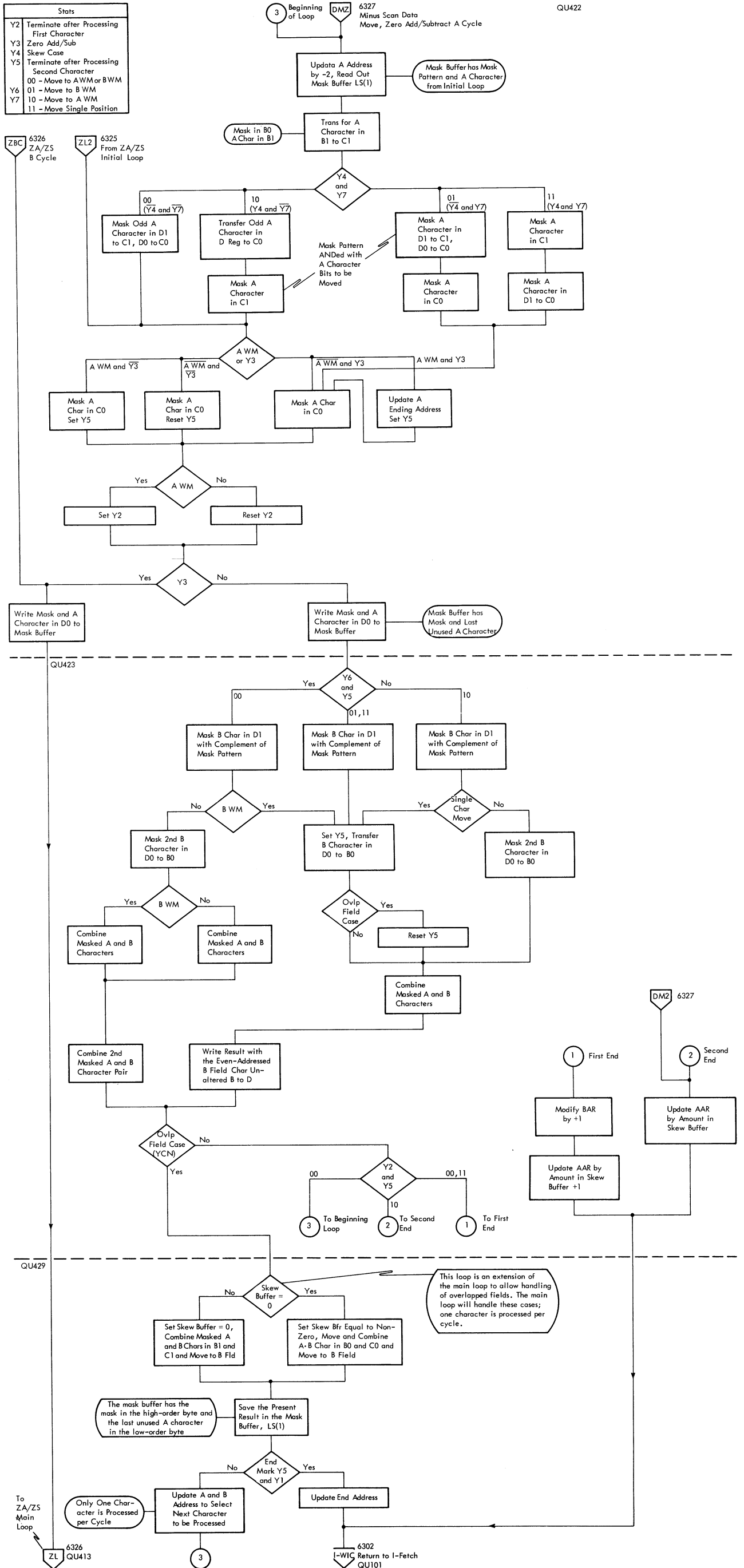


FIGURE 6328. 1410E DATA MOVE MINUS SCAN, ZERO ADD/SUBTRACT A CYCLES

Stats	
Y4	Skew Case
Y7	B Starting Address Even

QU424

DPI 6324

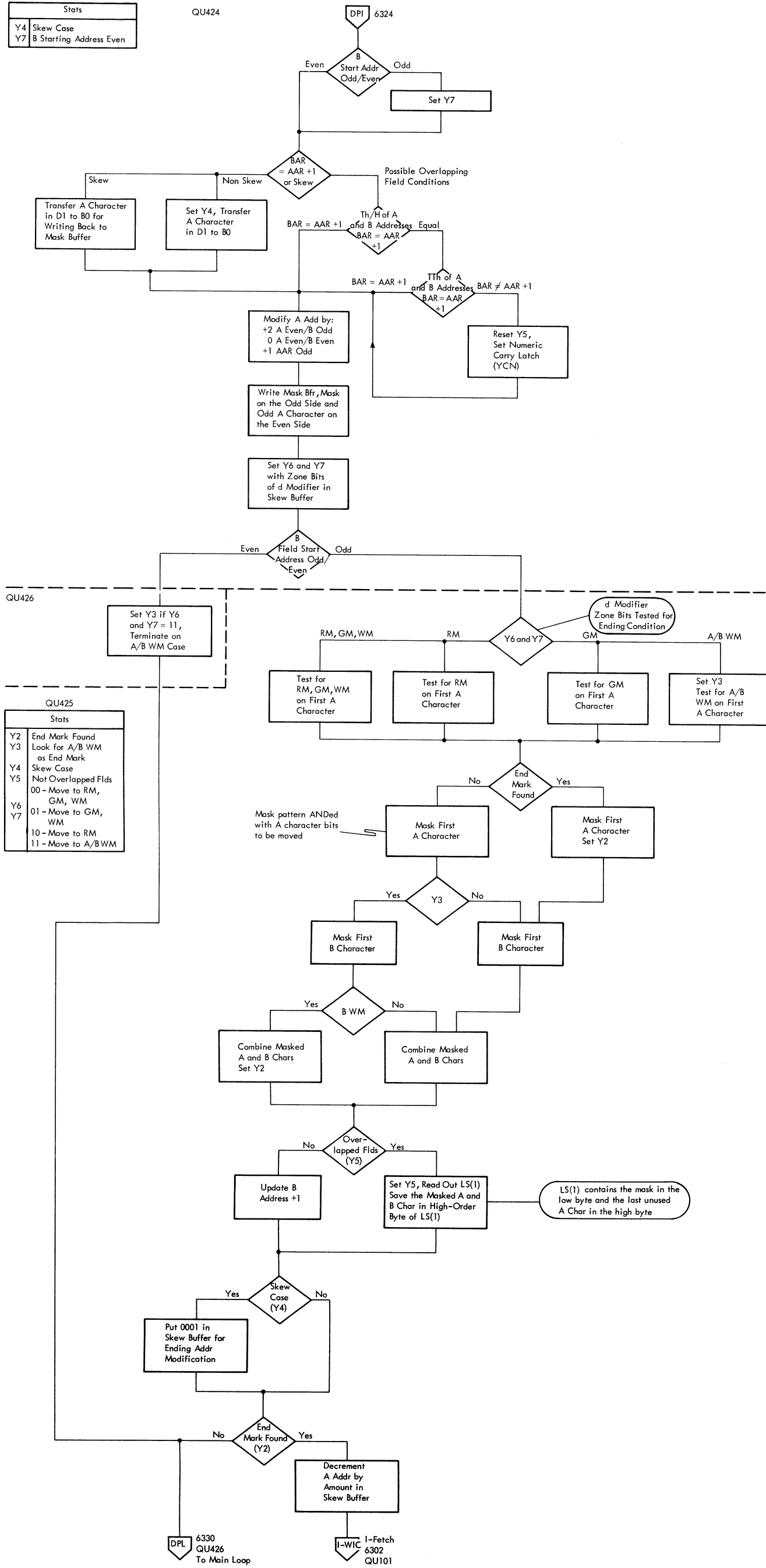


FIGURE 6329. 1410E DATA MOVE PLUS SCAN-INITIAL LOOP

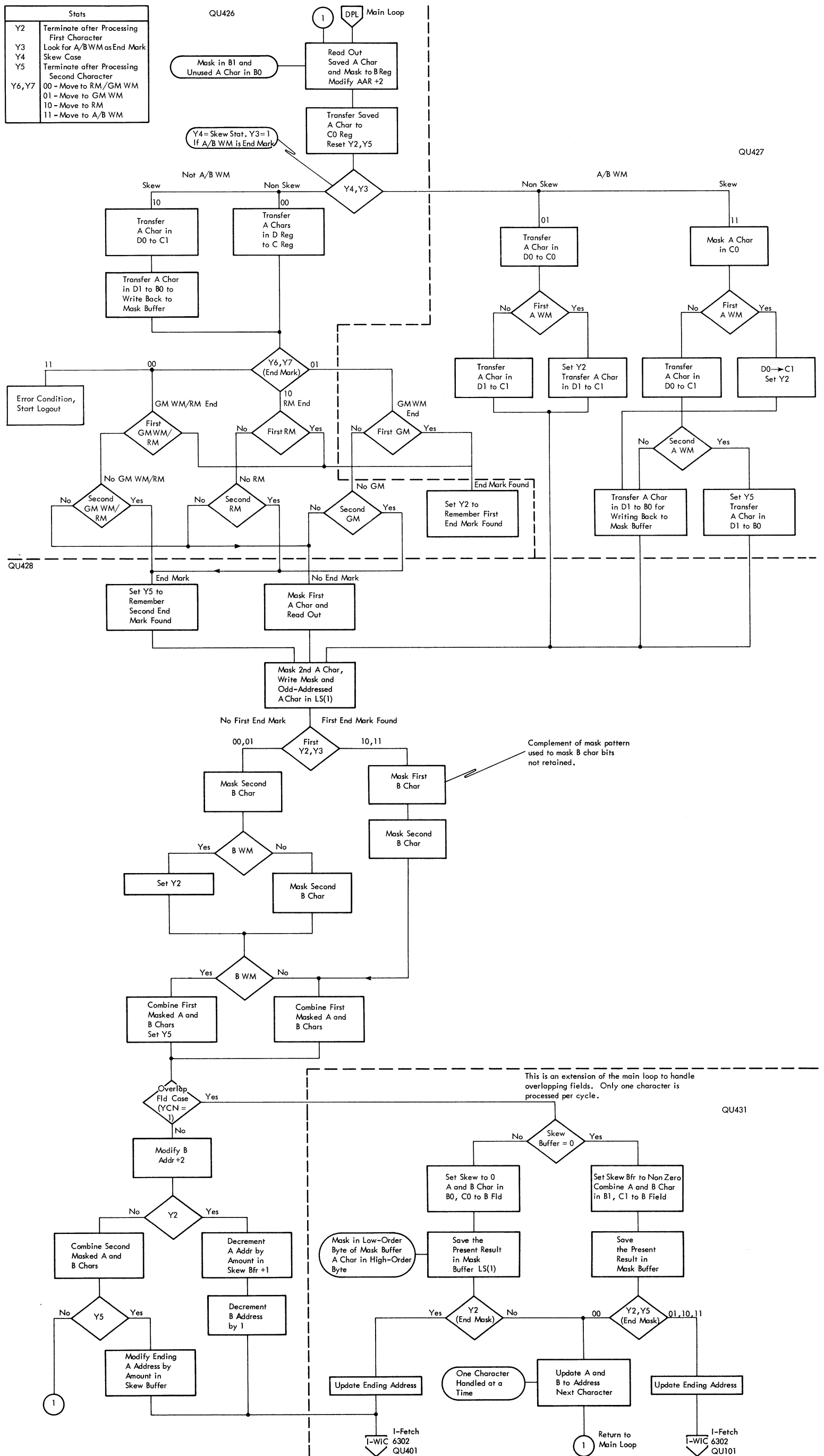


FIGURE 6330. 1410E DATA MOVE PLUS SCAN - MAIN LOOP

Stats	
Y2	A Field Sign Negative
Y4	Skew Case
Y5	0 = Compare
Y6	1 = Table Look-Up (TLU)
	B Start Address Odd

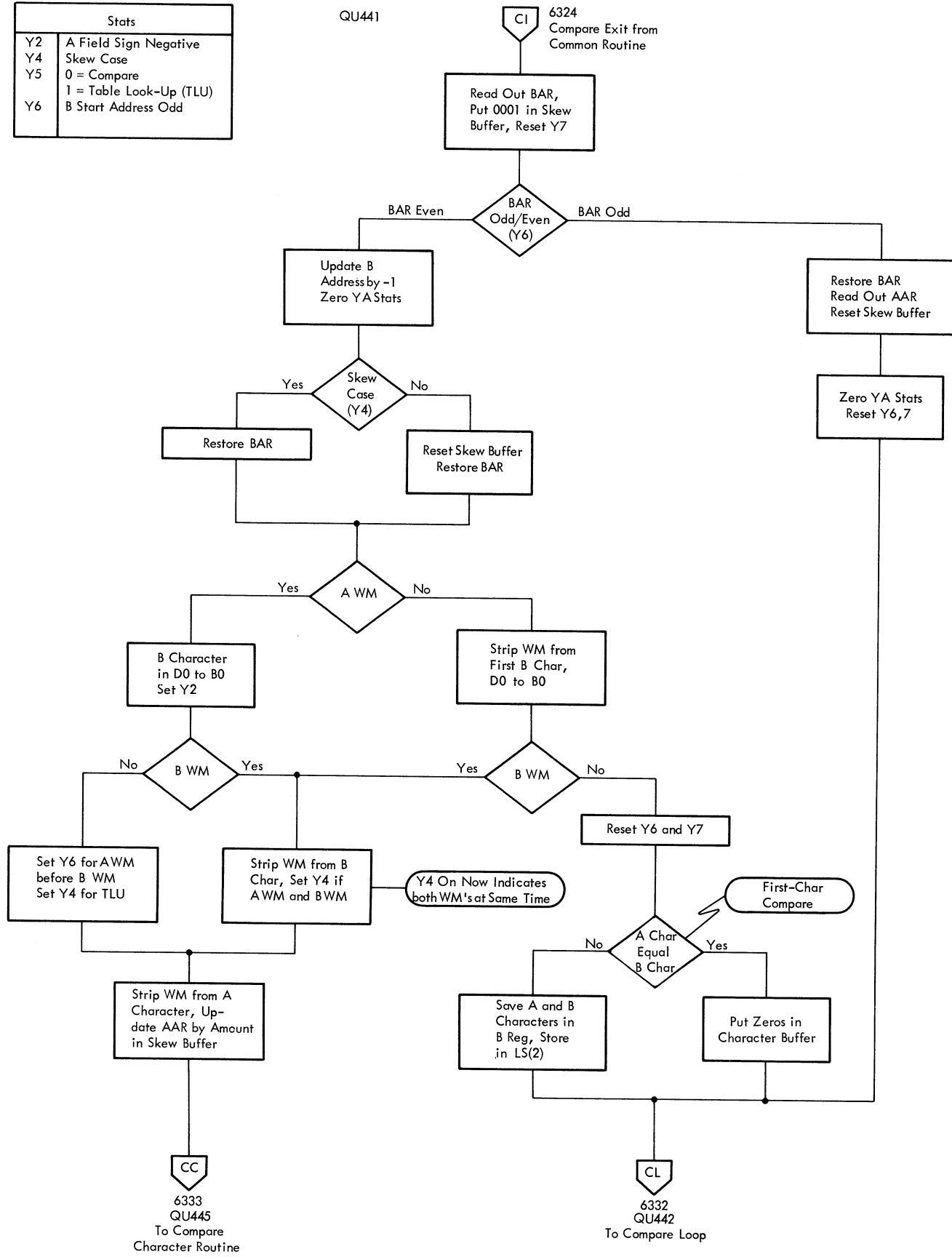


FIGURE 6331. 1410E COMPARE INITIAL LOOP

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

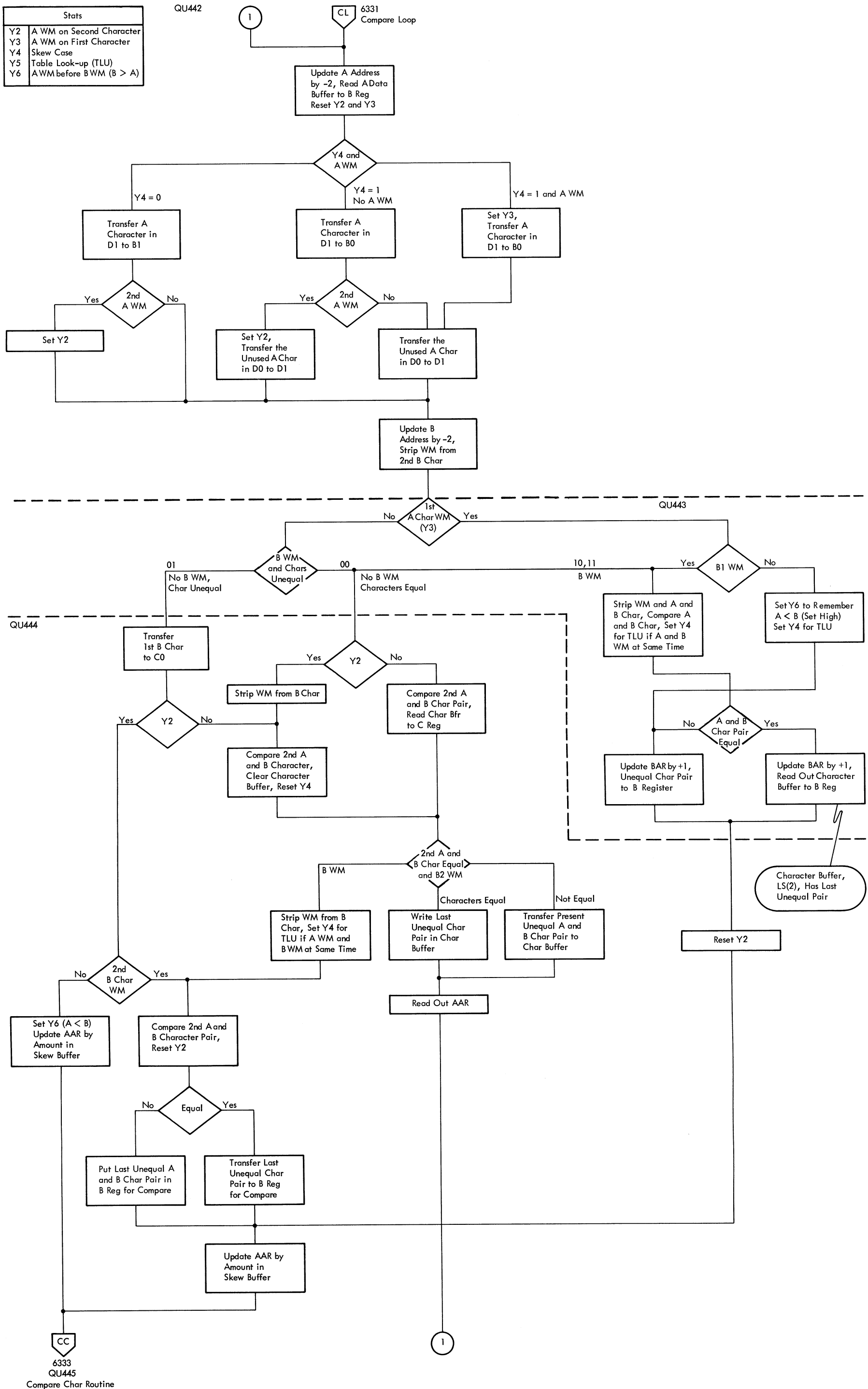


FIGURE 6332. 1410E COMPARE MAIN LOOP

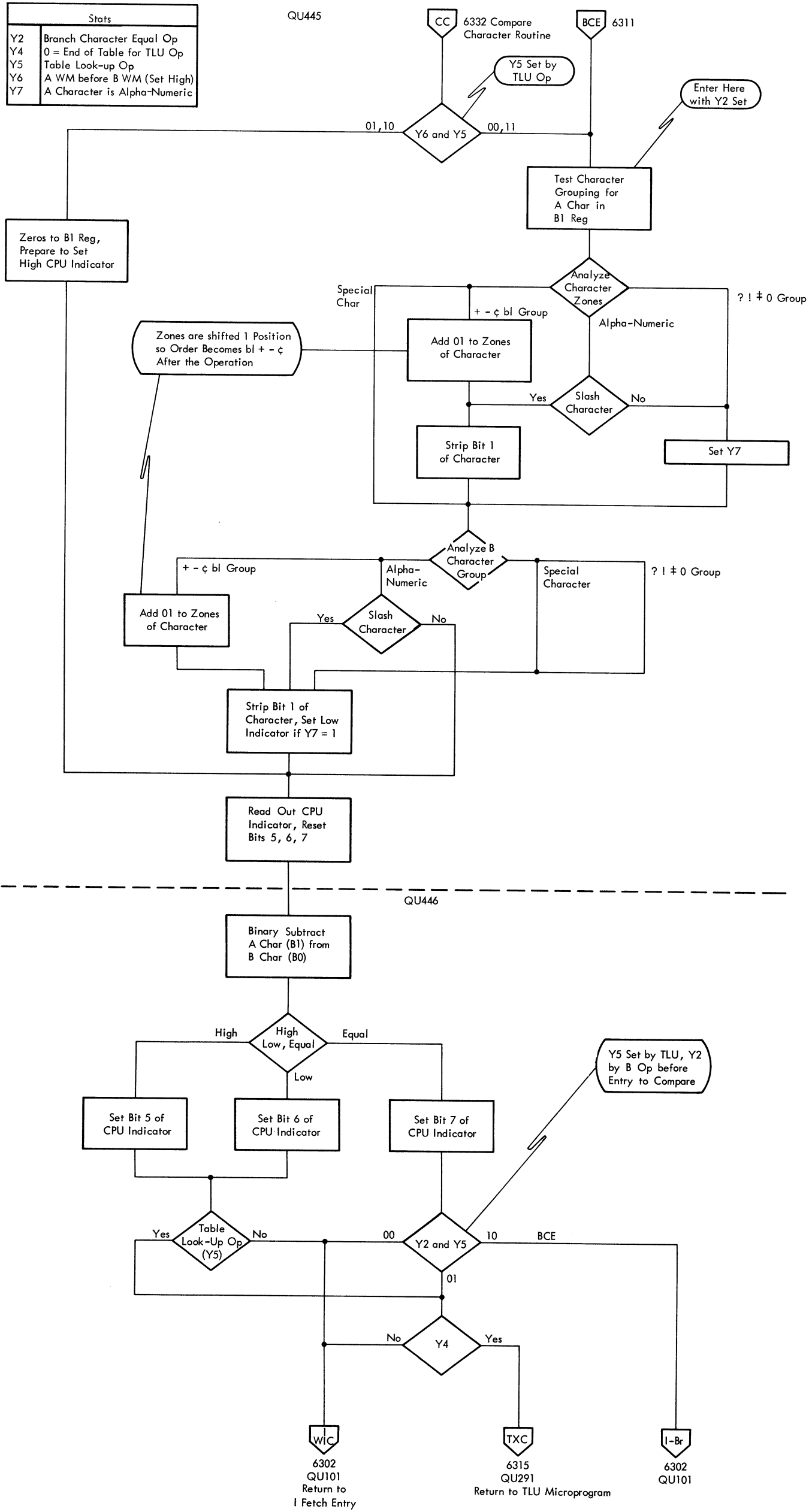


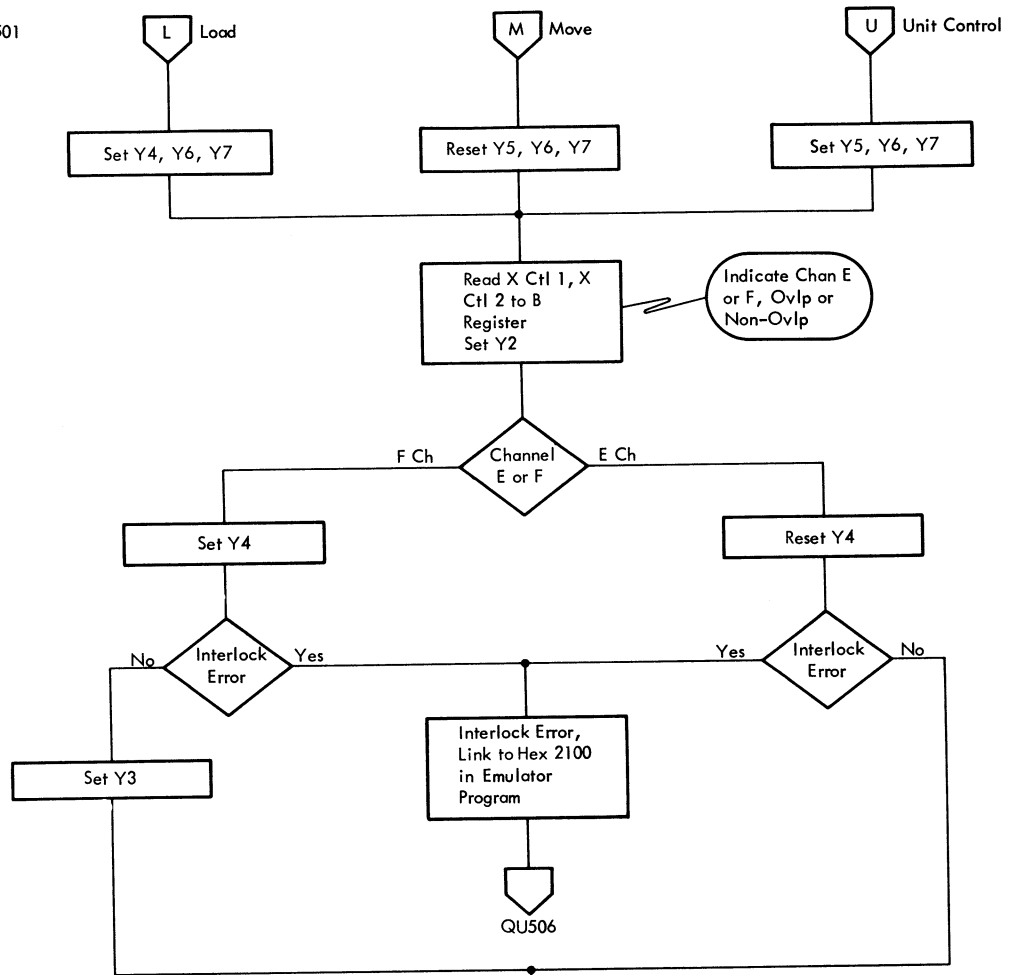
FIGURE 6333. 1410E COMPARE CHARACTER AND ENDING



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

QU501



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

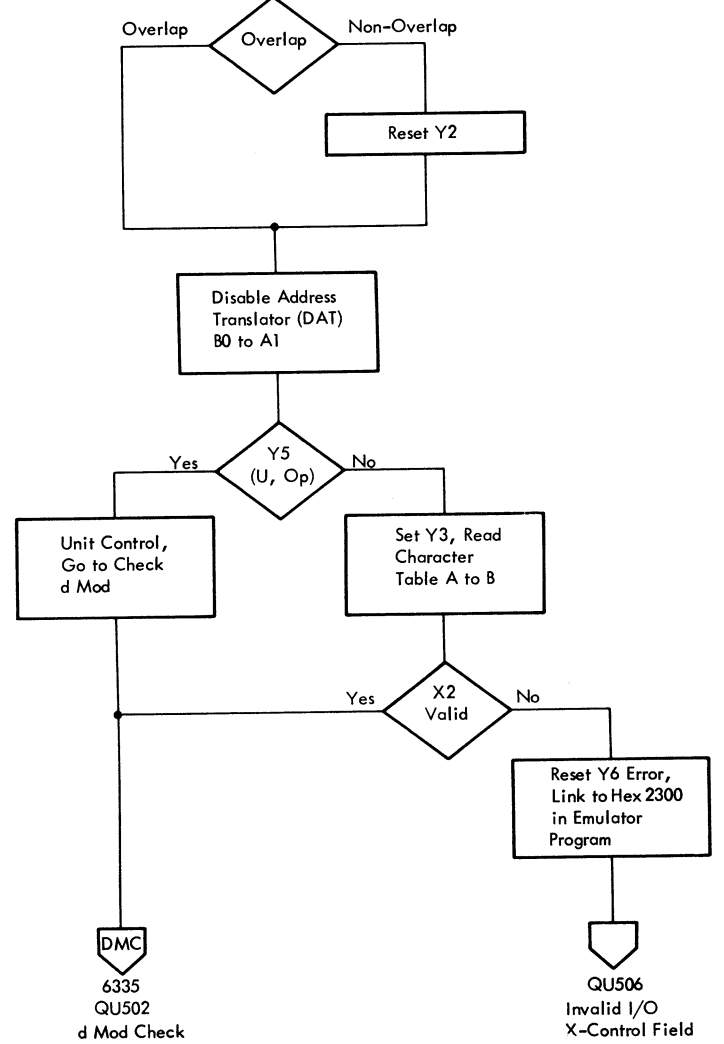


FIGURE 6334. 1410E UNIT CONTROL, I/O MOVE/LOAD, INITIAL LOOP

Stats	
Y2	Write WM as 1
Y3	End of Core Op
Y4	Load Mode
Y5	I/O No Op
Y6	Overlap
Y7	F Channel Op

QU502  
Move/Load

Register Contents	
A	20 and d Modifier
B	X1 and X2
C	System Status and d Modifier
D	Table Character for d Modifier
H	Hex F1
J	Hex F2, F3, or F4

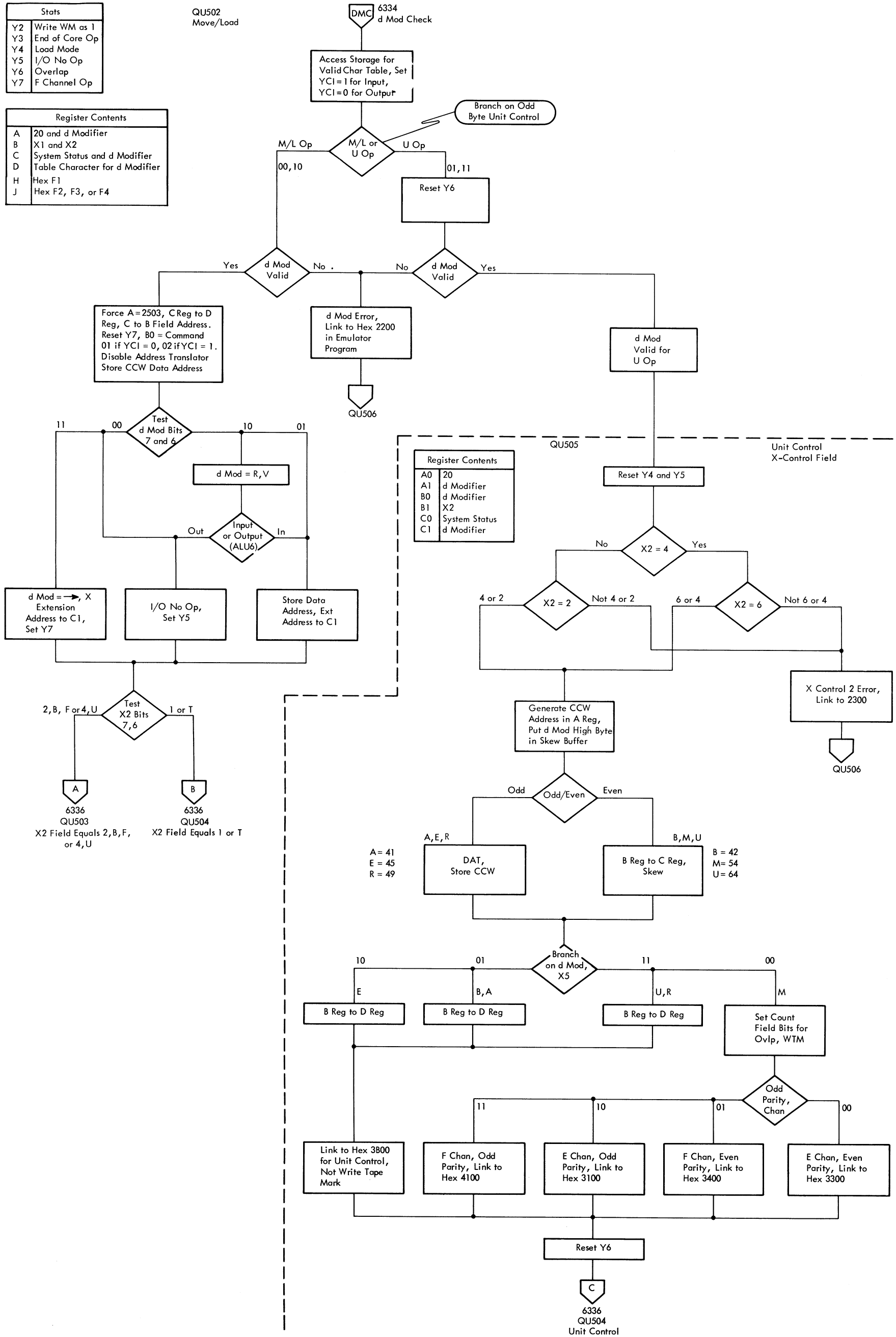
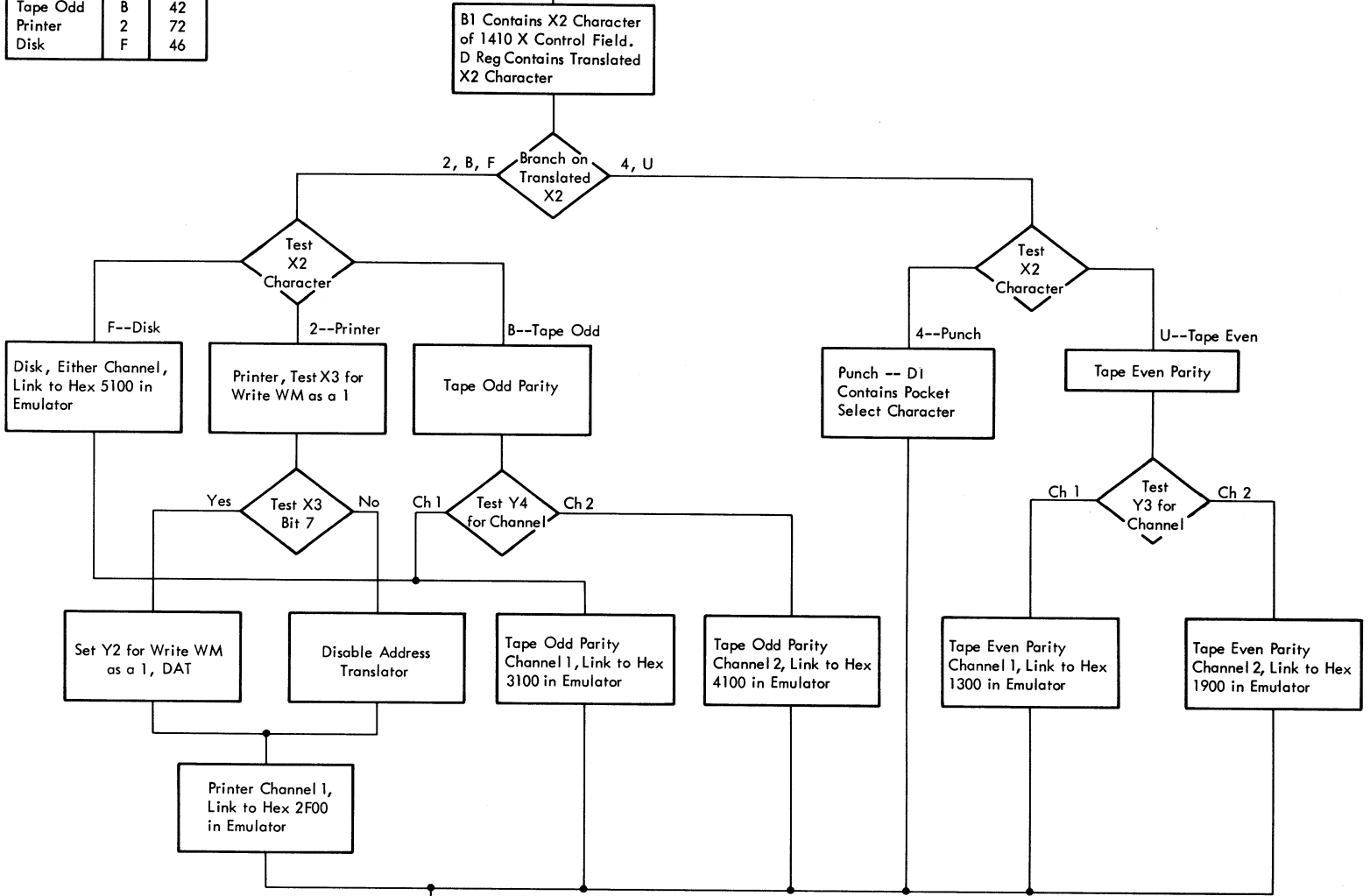


FIGURE 6335. X-CONTROL FIELD TRANSLATION

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

QU503

A 6335 X2 Field Equals 2, B, F or 4, 0



QU504

Packet	Cmd Byte
NR	02
1	42
2	82

C Reg to D Reg Store  
CCW, Store Command  
Ext Address

B 6335, X2  
Field Equals 1

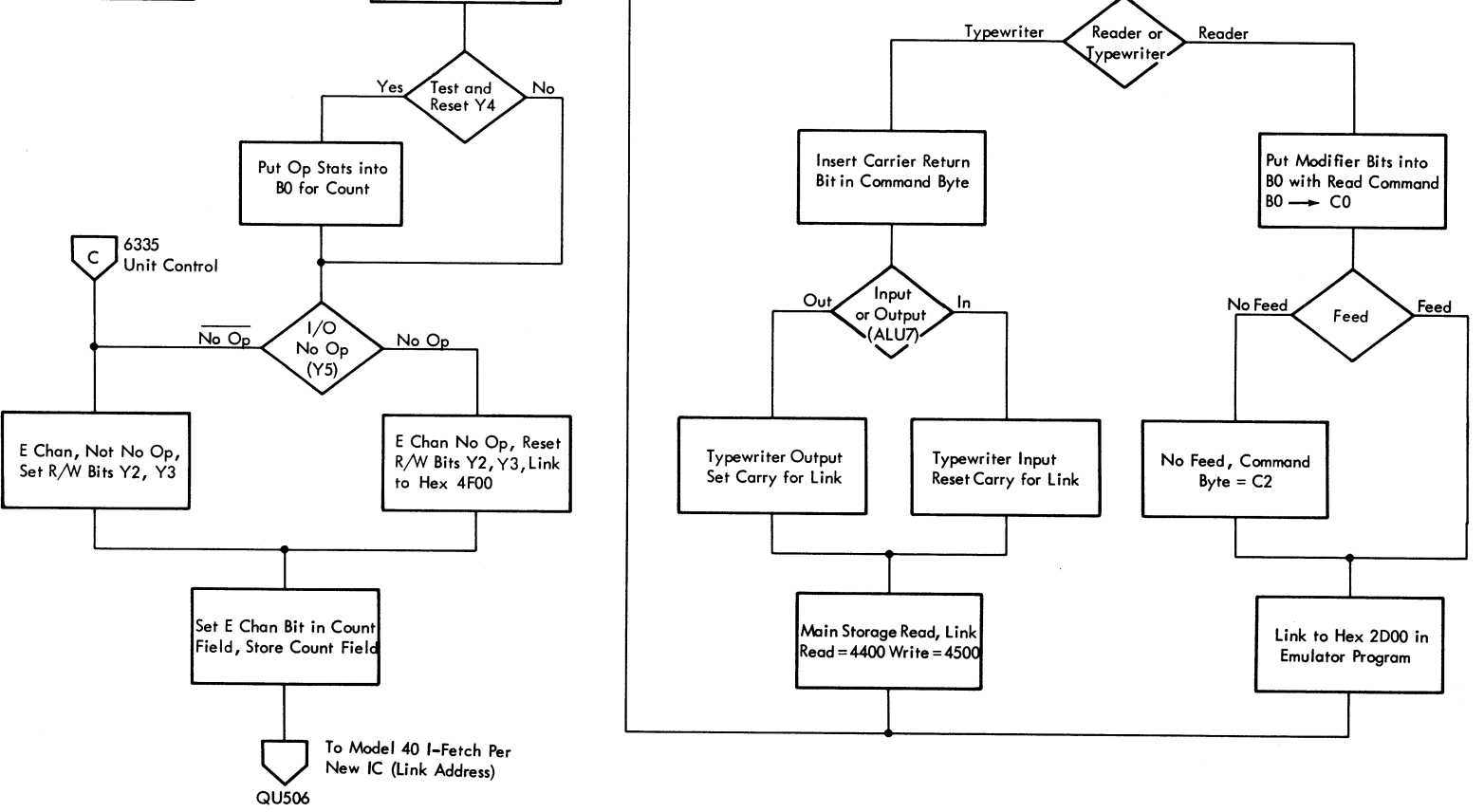
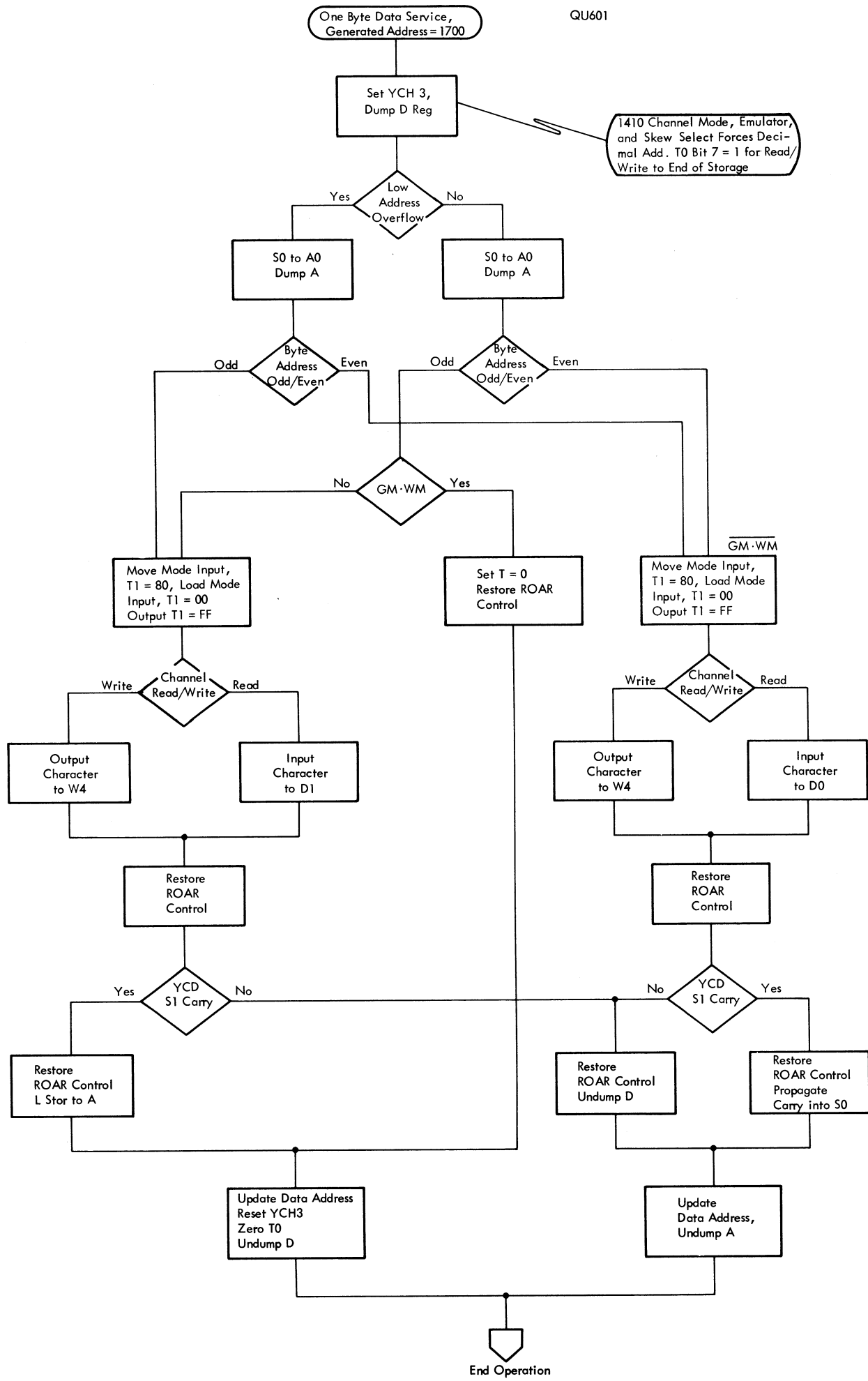


FIGURE 6336. 1410E I/O UNIT SELECTION



1410 Channel Mode, Emulator, and Skew Select Forces Decimal Add. T0 Bit 7 = 1 for Read/Write to End of Storage

FIGURE 6337. 1410E, ONE-BYTE DATA SERVICE

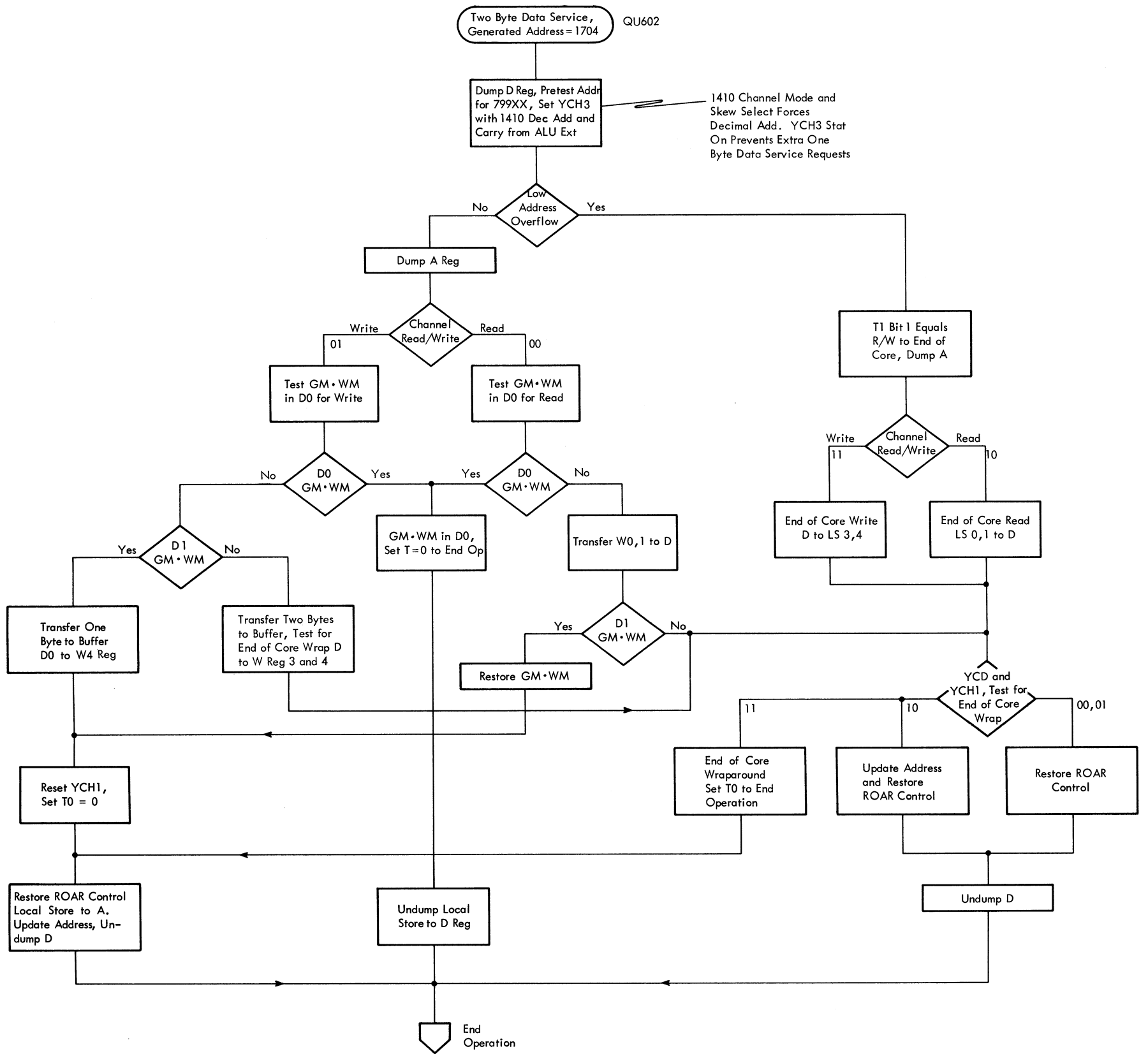


FIGURE 6338. 1410E, TWO-BYTE DATA SERVICE

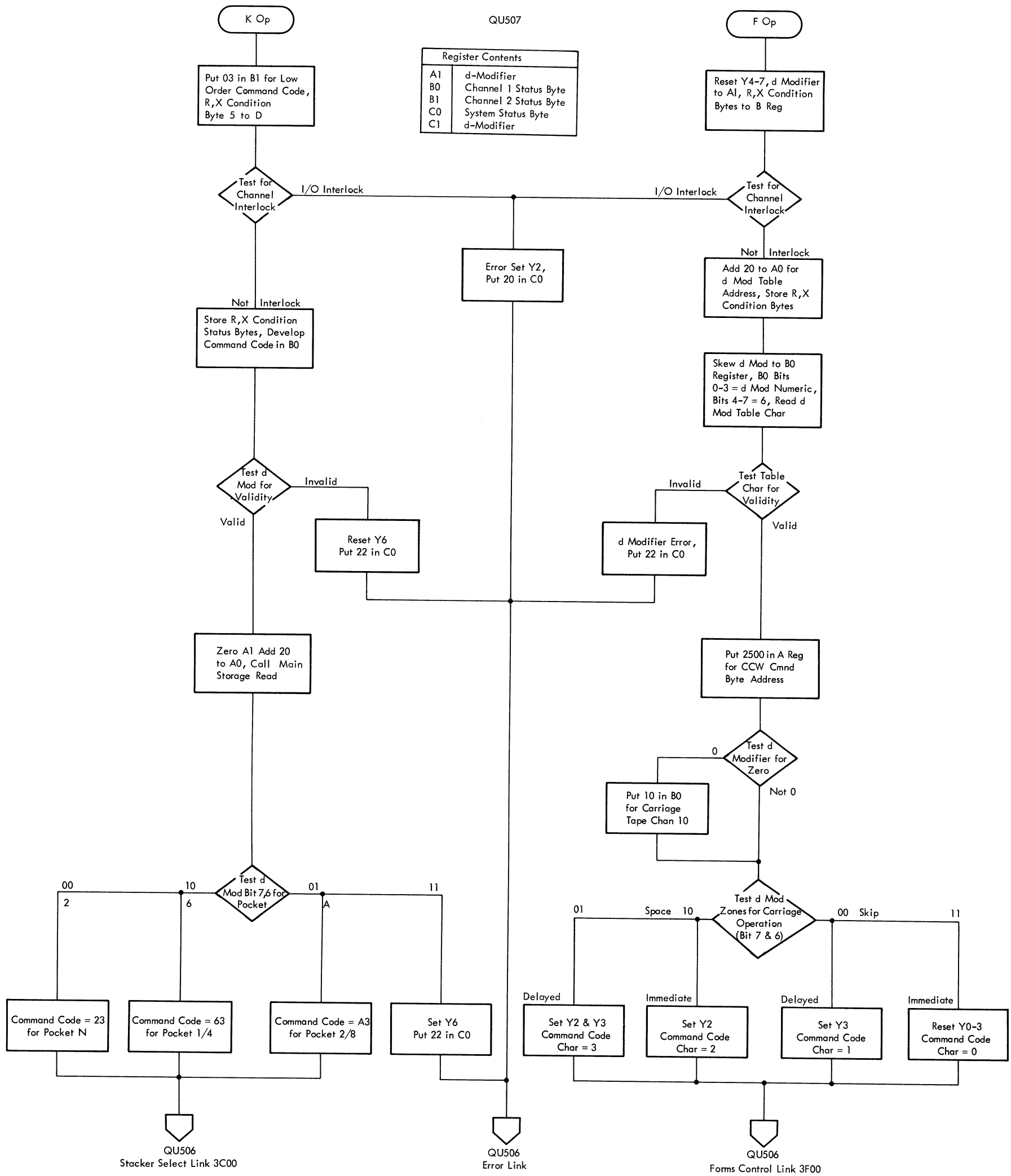


FIGURE 6339. 1410E FORMS CONTROL, STACKER SELECT

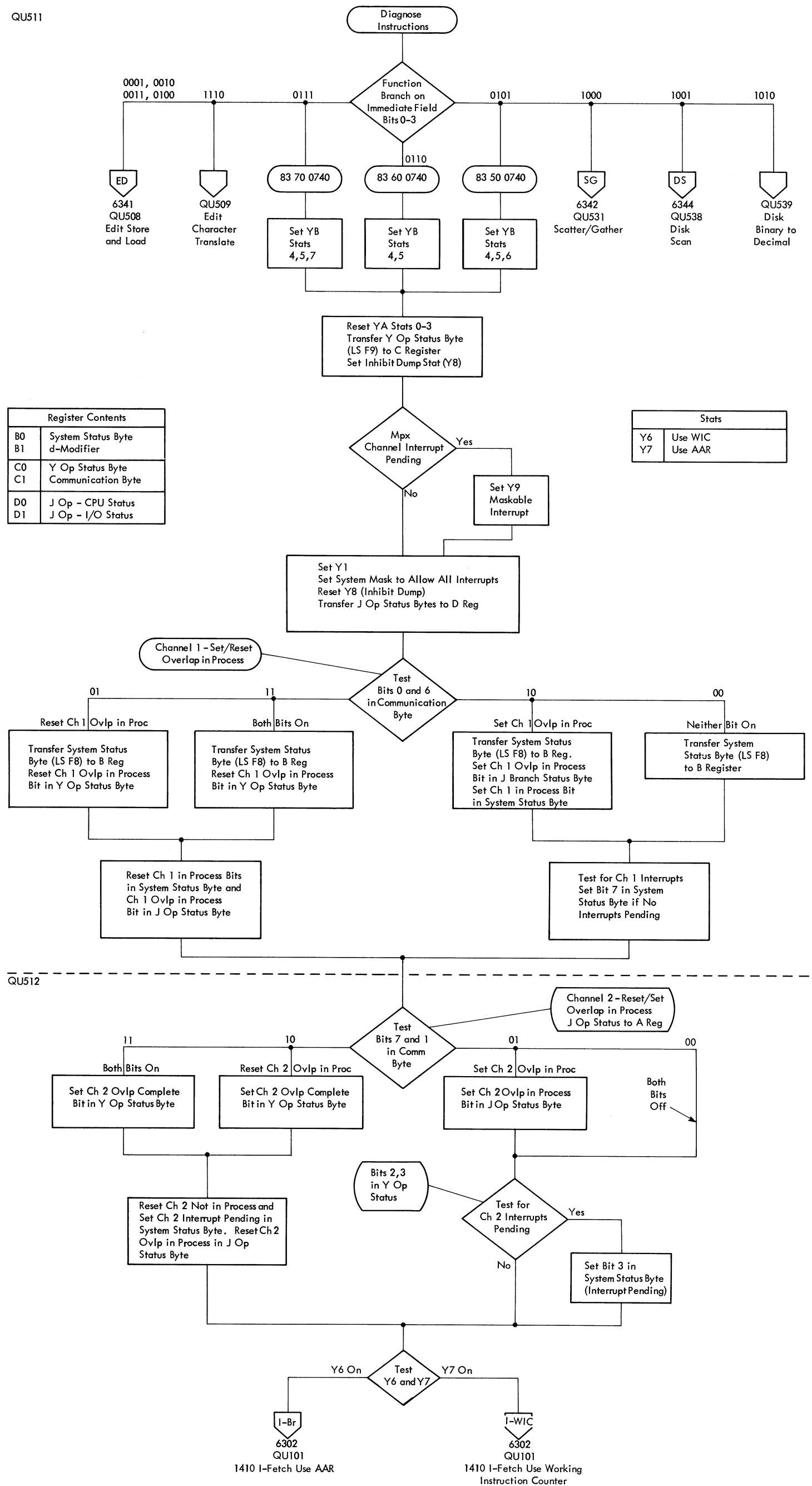


FIGURE 6340. 1410E DIAGNOSE INSTRUCTION, I-FETCH LINKAGES

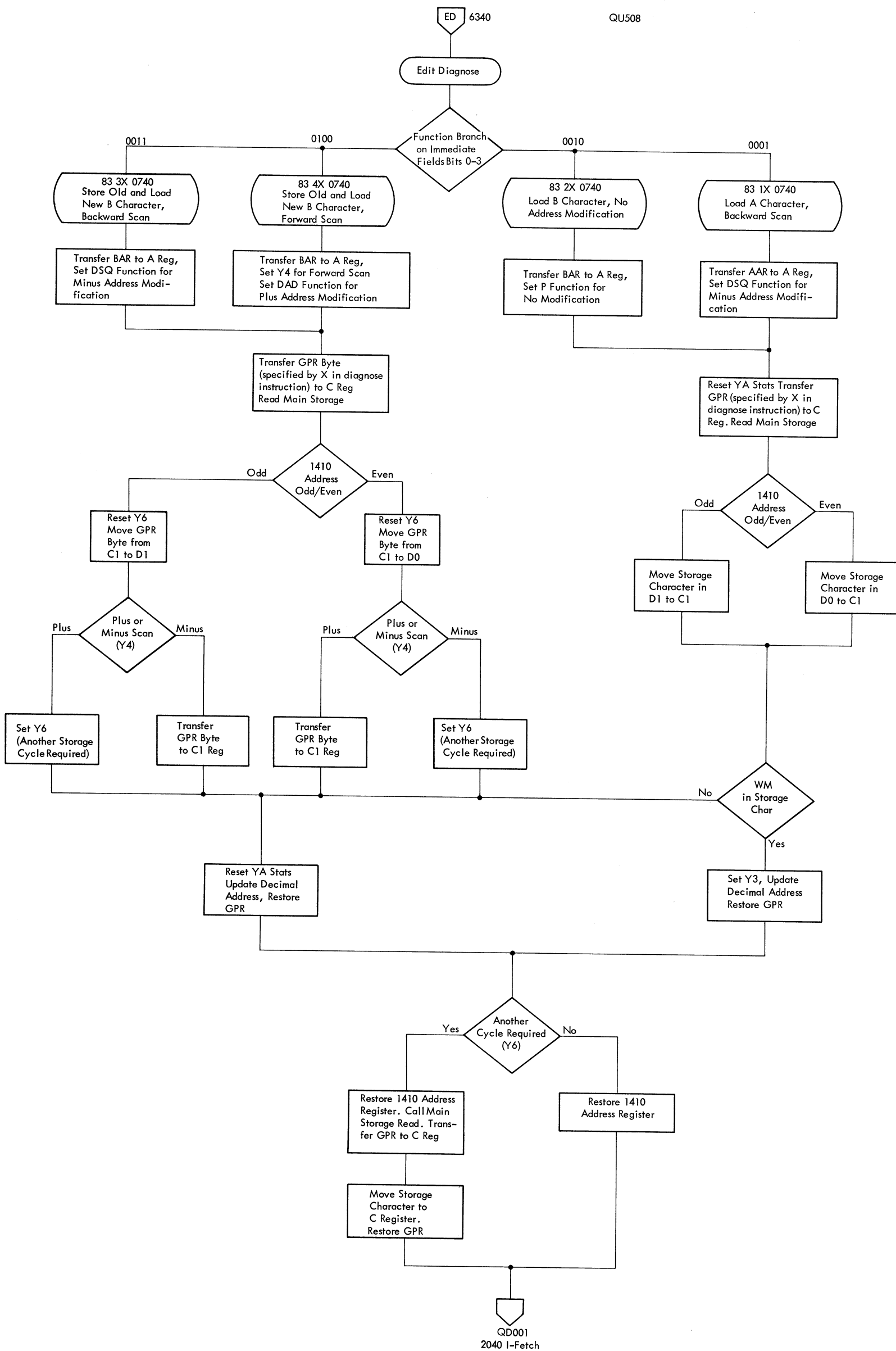


FIGURE 6341. 1410E EDIT DIAGNOSE



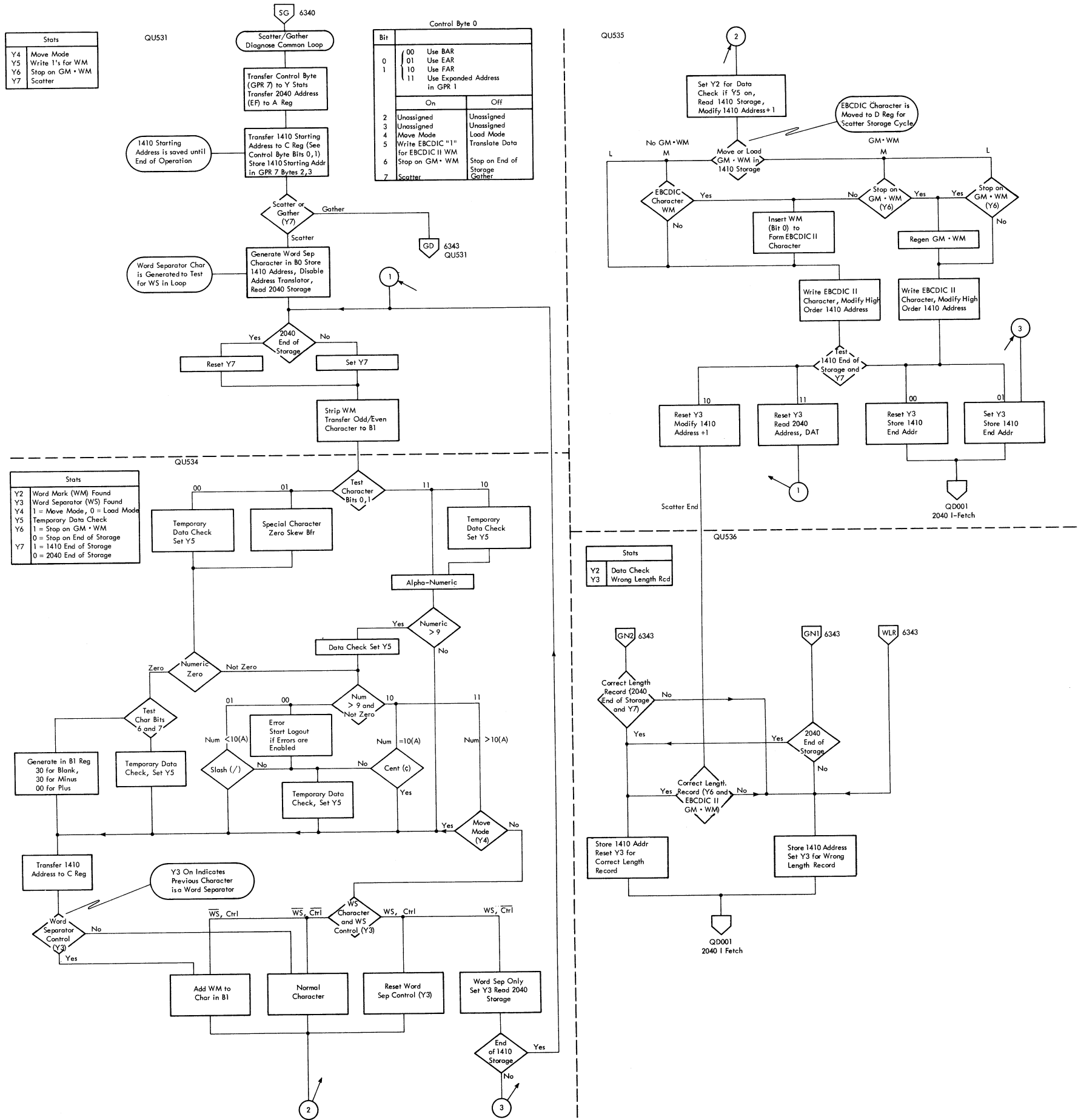


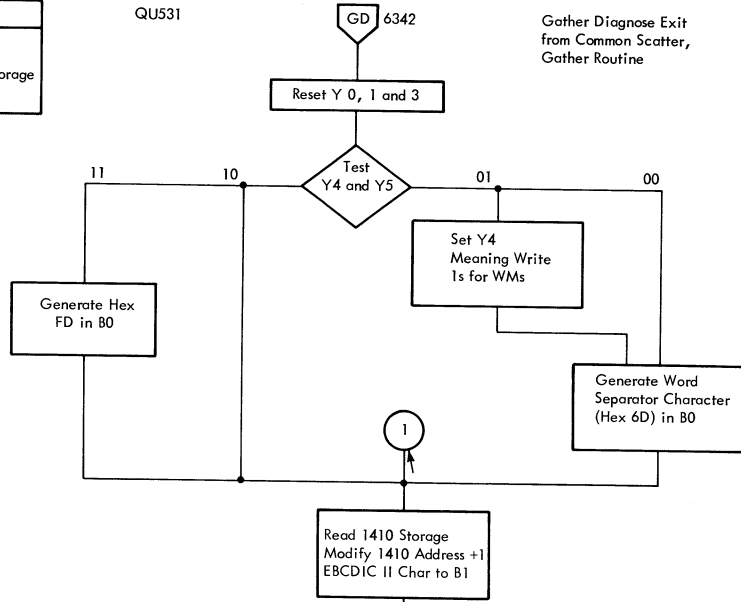
FIGURE 6342. 1410E SCATTER/GATHER DIAGNOSE

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

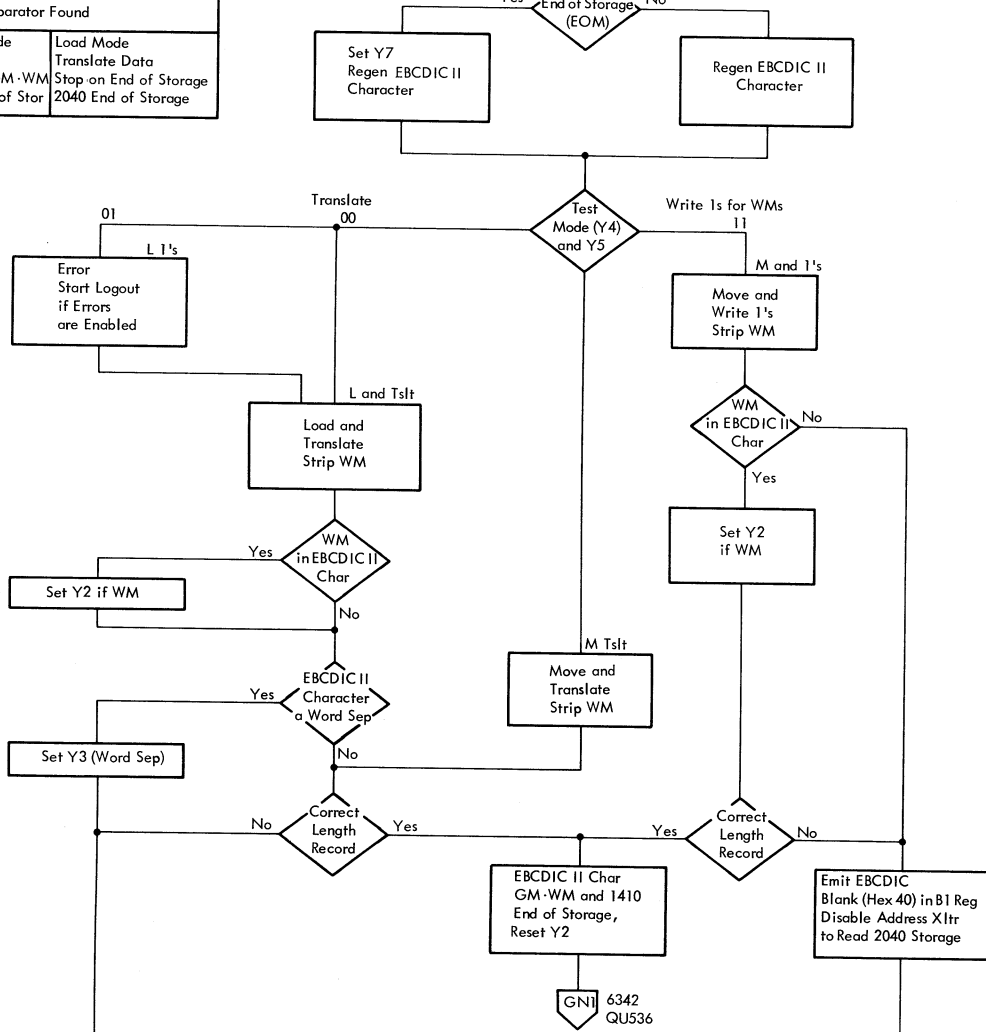
Gather Diagnose Exit  
from Common Scatter,  
Gather Routine



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

QU532

1410  
End of Storage  
(EOM)



QU533

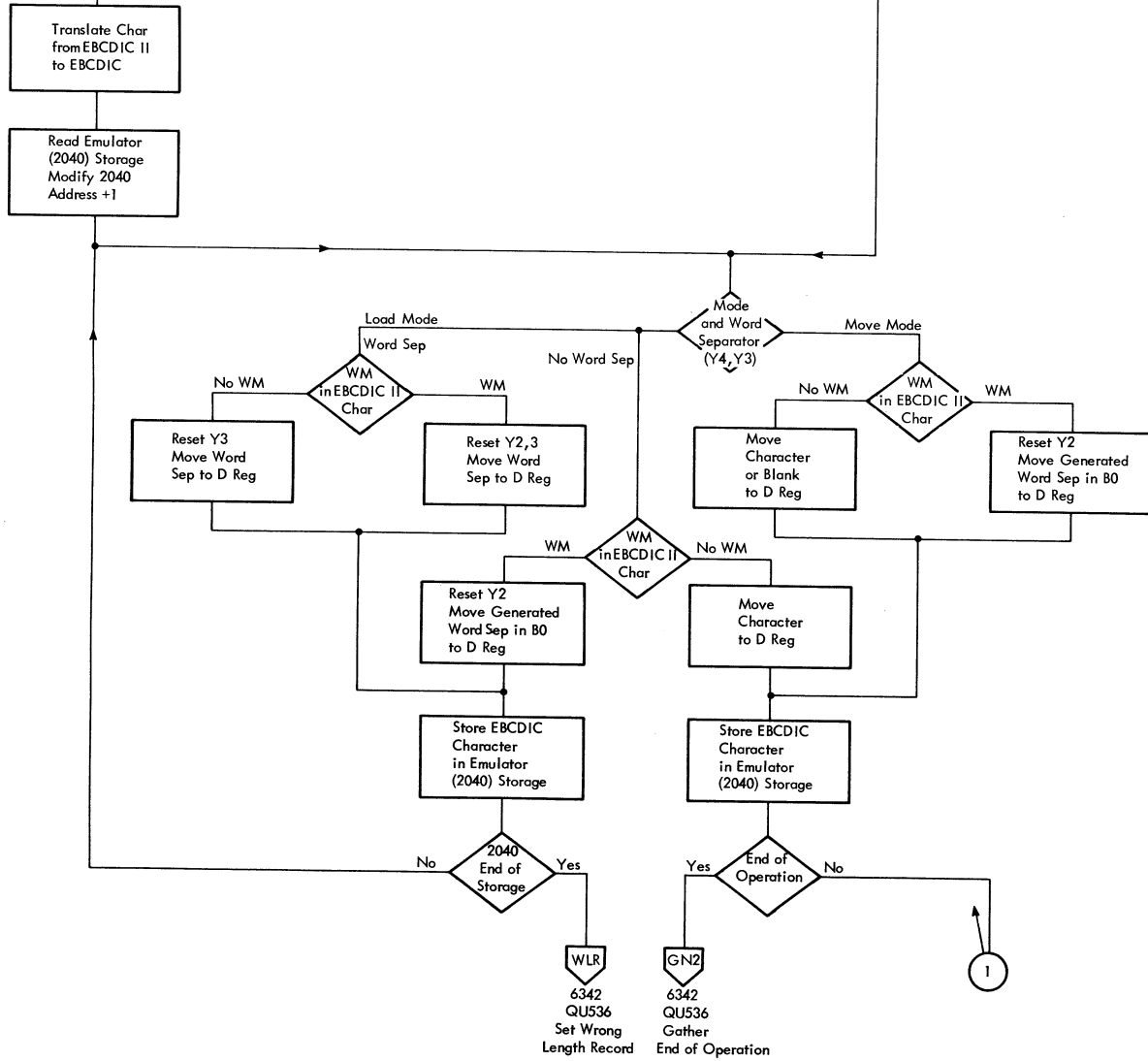


FIGURE 6343. 1410E GATHER DIAGNOSE

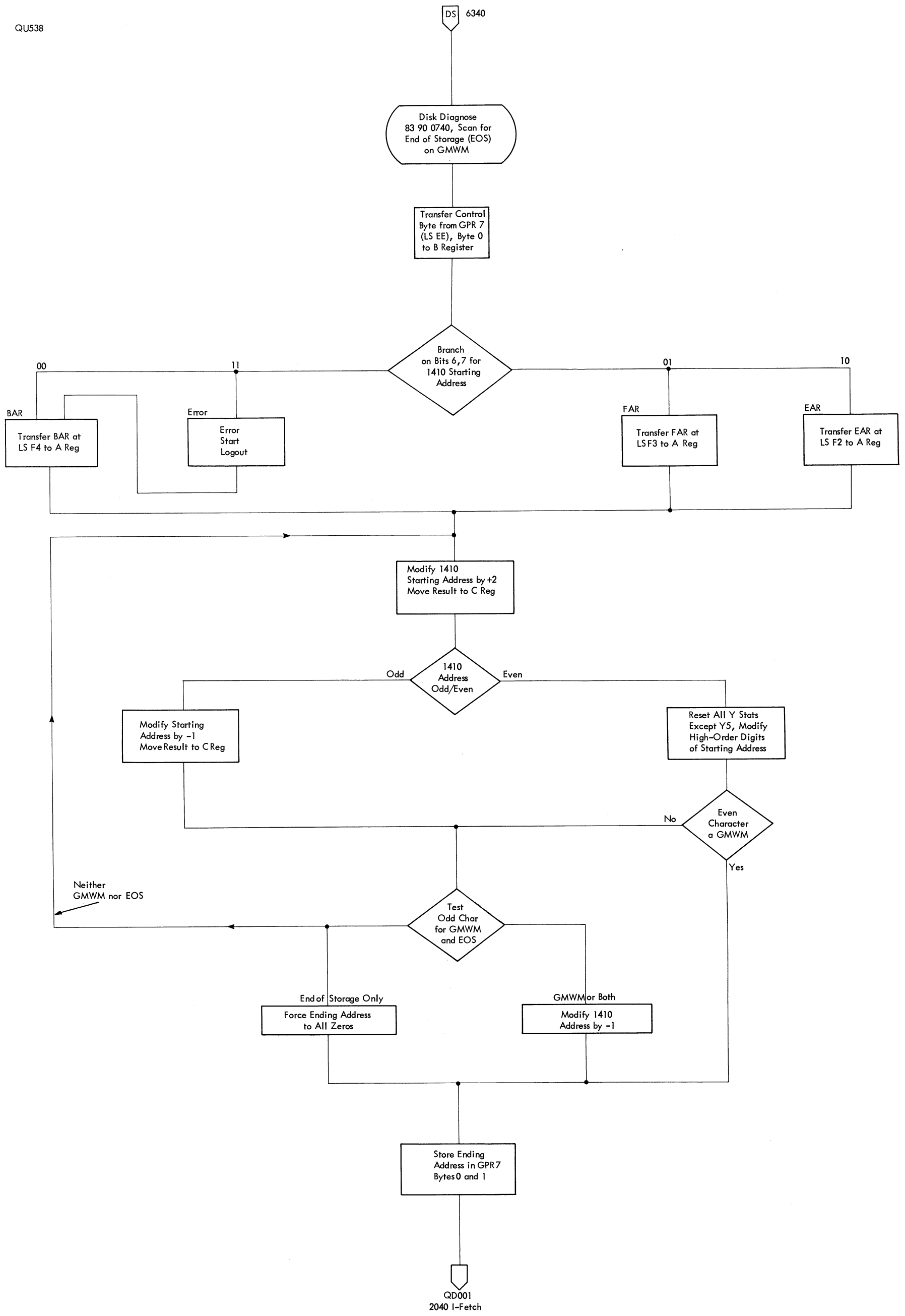


FIGURE 6344. 1410E DISK DIAGNOSE--END OF STORAGE OR GMWM SCAN

Add flow charts			
add (1401)	. . . . .	6202	
fixed-point	. . . . .	616	
general flow chart	. . . . .	657	
zero and add (1401)	. . . . .	6209	
AND flow chart	. . . . .	636	
Branch flow charts			
branch and link	. . . . .	608	
branch on bit equal	. . . . .	6219	
branch on character equal (1401)	. . . . .	6219	
branch on condition	. . . . .	618	
branch on count	. . . . .	612	
branch on index	. . . . .	632	
branch on word mark	. . . . .	6219	
branch on zone	. . . . .	6219	
multi-way (1401)	. . . . .	6205	
tests (1401)	. . . . .	6222	
1st and 2nd level function branches	. . . . .	607	
Carry latches	. . . . .	508	
Clock control	. . . . .	502	
Compare flow charts			
compare (1401)	. . . . .	6203	
RX algebraic	. . . . .	617	
SS decimal	. . . . .	662	
SS logical operation	. . . . .	649	
Condensed logic flow charts (CLF)			
branch and link	. . . . .	608	
branch on condition	. . . . .	618	
branch on count	. . . . .	612	
branch on index	. . . . .	632	
convert binary to decimal	. . . . .	613	
convert decimal to binary	. . . . .	610	
decimal divide add/subtract paths	. . . . .	652	
decimal divide example	. . . . .	651	
decimal move with offset	. . . . .	656	
decimal pack	. . . . .	654	
decimal unpack	. . . . .	655	
diagnose instruction	. . . . .	639	
edit and mark	. . . . .	643	
fixed-point divide initialization	. . . . .	623	
fixed-point divide loop	. . . . .	624	
floating-point divide loop	. . . . .	630	
floating-point load and store	. . . . .	627	
floating-point mult/divide initialization	. . . . .	628	
floating-point multiply loop	. . . . .	629	
flow chart for decimal add sub compare	. . . . .	657	
how to use flow charts	. . . . .	599	
I/O codes, common decoding	. . . . .	666	
I/O interrupts	. . . . .	671	
instruction fetch microprogram	. . . . .	601	
instruction fetch, RS and SI operations	. . . . .	604	
instruction fetch, RX fixed-point	. . . . .	602	
instruction fetch, RX floating-point	. . . . .	603	
instruction fetch, SS decimal	. . . . .	606	
instruction fetch, SS logical	. . . . .	605	
instruction matrix	. . . . .	600	
load PSW	. . . . .	638	
machine status at function branches	. . . . .	607	
multiplex channel microprogram	. . . . .	669	
multiplex channel status	. . . . .	670	
read direct and write direct	. . . . .	637	
RR and RX fixed-point arith and logic	. . . . .	615	
RR and RX fixed-point multiply	. . . . .	620	
RR and RX fixed-point multiply, loops	. . . . .	622	
RR and RX fixed-point multiply, notes	. . . . .	621	
RR and RX floating-point operation	. . . . .	626	
RR fixed-point sign operation	. . . . .	611	
RR floating-point sign operations	. . . . .	625	
RS load and store multiple	. . . . .	634	
RX compare algebraic	. . . . .	617	
RX fixed-point add and subtract	. . . . .	616	
sel channel I/O instructions microprogram	. . . . .	675	
selector channel status	. . . . .	674	
set and insert storage key	. . . . .	609	
set program mask	. . . . .	614	
set system mask	. . . . .	633	
shifts	. . . . .	635	
SI operations - AND, OR, XOR, move	. . . . .	636	
SS decimal compare	. . . . .	662	
SS decimal divide	. . . . .	650	
SS decimal load and process operand 1	. . . . .	658	
SS decimal load operand 2 and process	. . . . .	660	
SS decimal load zero and add entry	. . . . .	659	
SS decimal multiply	. . . . .	653	
SS decimal terminate	. . . . .	661	
SS edit, refill	. . . . .	644	
SS logical operation, compare	. . . . .	649	
SS logical operations, move complete	. . . . .	648	
SS operations - move, zone, and numeric	. . . . .	647	
SS translate	. . . . .	641	
SS translate and test	. . . . .	642	
start I/O instruction (multiplex channel)	. . . . .	665	
start I/O microprogram Mpx channel	. . . . .	667	
store PSW-general flow	. . . . .	686	
test channel and Mpx halt I/O	. . . . .	666	
test I/O multiplex channel microprogram	. . . . .	668	
test under mask	. . . . .	631	
update timer microprogram	. . . . .	671	
CPU data flow	. . . . .	012	
Data flow charts			
CPU data flow	. . . . .	012	
CPU microprogram flow chart	. . . . .	014	
microprogram data flow	. . . . .	013	
multiplex and MS unit data and control	. . . . .	101	
ROS control word	. . . . .	015	
selector channel data flow	. . . . .	011	
Decimal operations flow charts			
add	. . . . .	657	
compare, decimal add sub	. . . . .	657	
compare	. . . . .	662	
convert binary to decimal	. . . . .	613	
convert decimal to binary	. . . . .	610	
divide	. . . . .	650	
divide, add/subtract paths	. . . . .	652	
divide example	. . . . .	651	
instruction fetch	. . . . .	606	
load and process operand 1	. . . . .	658	
load operand 2 and process	. . . . .	660	
load zero and add entry	. . . . .	659	
move with offset	. . . . .	656	
multiply	. . . . .	653	
pack	. . . . .	654	
subtract	. . . . .	657	
terminate	. . . . .	661	
unpack	. . . . .	655	
Divide flow charts			
decimal divide example	. . . . .	651	
decimal paths	. . . . .	652	
fixed-point initialization	. . . . .	623	
fixed-point loop	. . . . .	624	
floating-point initialization	. . . . .	628	
floating-point loop	. . . . .	630	
SS decimal	. . . . .	650	
Dump	. . . . .	669	
Edit flow charts			
edit	. . . . .	643	
edit and mark	. . . . .	643	
SS edit, refill	. . . . .	644	
Error check MAP's			
control	. . . . .	906	
early	. . . . .	907	
late	. . . . .	908	
Fetch CAW	. . . . .	667	
I/O flow charts			
carriage control (1401)	. . . . .	6213	
common decoding	. . . . .	666	
halt	. . . . .	666	
interrupts	. . . . .	671	

M, L, U operation (1401)	6210	RR and RX fixed point, notes	621
read and punch column binary (1401)	6225	SS decimal	653
selector channel instructions	675	Objectives, channel microprogram	666, 669, 670
stacker select (1401)	6213	OR flow chart	636
start I/O instruction	665	Power supplies	
start microprogram Mpx channel	667	divide (1401)	6208
test channel	666	Mid-Pac MAP	917
test I/O microprogram	668	Mid-Pac wiring diagram	510
Instruction fetch		2.5 kc HF MAP	918
instruction fetch	601	2.5 kc HF wiring diagram	511
instruction fetch (1401)	6200	Read only storage	911
RS operations (2nd level)	604	Relocate feature (1401)	6221
RX fixed-point (2nd level)	602	Relocate latches (1401)	015
RX floating-point (2nd level)	603	ROS control word	015
SI operations (2nd level)	604	RR instruction flow charts	
SS decimal	606	fixed-point arithmetic and logic	615
SS logical (2nd level)	605	fixed-point multiply	620
Load fixed-point flow charts		fixed-point multiply, detail of loops	622
complement	611	fixed-point multiply, notes	621
negative	611	fixed-point sign operation	611
positive	611	floating-point operations	626
test	611	floating-point sign operations	625
Load floating-point flow charts		RS instruction flow charts	
complement	625	instruction fetch	604
halve	625	load and store multiple	634
load	625	RX instruction flow charts	
negative	625	compare algebraic	617
positive	625	fixed-point add and subtract	616
test	625	fixed-point arithmetic and logic	615
Load PSW	638	fixed-point multiply	620
Local storage	912	fixed-point multiply, detail of loops	622
Logical operations flow charts		fixed-point multiply, notes	621
compare	649	fixed-point, instruction fetch	602
move complete	648	floating-point operation	626
move zone and numeric	647	floating-point, instruction fetch	603
SS instruction fetch	605	Selector channel	
Main storage	914	channel data flow	011
Main storage X-dimension drive	513	channel MAP	916
Malfunction analysis procedure (MAP)		channel control	509
control check	906	channel status	674
early check	907	I/O instructions microprogram	675
interpret errors	901	set channel to 1401 mode (1401)	6217
late check	908	SI instruction flow charts	
local storage	912	AND	636
main storage	914	instruction fetch	604
Mid-Pac power supply	917	move	636
multiplex channel	915	OR	636
read only storage	911	XOR	636
selector channel	916	Simplified logic diagrams (SLD)	
storage protect	913	carry latches	508
2.5 kc HF power supply	918	clock control (SP)	502
Microprograms		decimal correction	507
CPU flow chart	014	decimal filler	506
data flow	013	function and control registers	505
I/O instructions (selector channel)	675	LSAR parity generation	501
I/O interrupts	671	main storage control and timing circuits	504
instruction fetch	601	main storage X-dimension drive	513
multiplex channel	669	Mid-Pac power supply wiring diagram	510
start I/O Mpx channel	667	multiplex channel controls	512
test I/O Mpx channel	668	selector channel controls	509
update timer	671	2.5 kc HF power supply wiring diagram	511
Move flow chart	636	SS instruction flow charts	
Mpx channel		compare	649
channel control	912	decimal compare	662
channel MAP	915	decimal divide	650
channel status	670	decimal load and process operand 1	658
halt I/O	666	decimal load operand 2 and process	660
multiplex and MS unit data and control	101	decimal load zero and add entry	659
multiplex channel microprogram	669	decimal multiply	653
start I/O instruction (Mpx chan)	665	decimal terminate	661
start I/O microprogram	667	edit	644
test I/O microprogram	668	instruction fetch, decimal	606
Multiply flow charts		instruction fetch, logical	605
floating-point initialization	628	move complete	648
floating-point loop	629	move numeric	647
multiply (1401)	6208	move zone	647
RR and RX fixed point	620	refill	644
RR and RX fixed point, detail of loops	622	translate	641

translate and test . . . . .	642	1410/7010 compatibility feature flow charts	
Storage MAP's		add . . . . .	6316, 6317
local storage . . . . .	912	address readout . . . . .	6304
main storage . . . . .	914	arithmetic operations . . . . .	6316
read only storage . . . . .	911	branch if bit equal . . . . .	6311
storage protect . . . . .	913	branch if character equal . . . . .	6311
Store CSW . . . . .	668	branch if word mark and/or zone equal . . . . .	6311
Store PSW . . . . .	686	branch on channel status . . . . .	6309
Subtract flow charts		branch on internal indicator . . . . .	6310
general flow chart . . . . .	657	carriage control . . . . .	6339
RX fixed-point . . . . .	616	chaining . . . . .	6307
subtract (1401) . . . . .	6202	clear storage . . . . .	6314
zero and subtract (1401) . . . . .	6209	clear word mark . . . . .	6314
Undump . . . . .	669	compare . . . . .	6331
XOR flow chart . . . . .	636	control field readout . . . . .	6303
X-dimension drive, main storage . . . . .	913	control field translation . . . . .	6335
1401/1460 compatibility feature flow charts		data move . . . . .	6324
add and subtract . . . . .	6202	data move minus scan . . . . .	6337
address modify . . . . .	6213	data move plus scan . . . . .	6329
branch if bit equal . . . . .	6219	data service . . . . .	6337
branch if character equal . . . . .	6219	diagnose instructions . . . . .	6340
branch if word mark or zone . . . . .	6219	disk . . . . .	6344
branch tests . . . . .	6222	edit . . . . .	6341
carriage control . . . . .	6212	gather . . . . .	6342, 6343
clear storage and special clears . . . . .	6214	scatter . . . . .	6342
clear word mark . . . . .	6214	divide . . . . .	6316, 6322
compare . . . . .	6203	edit diagnose . . . . .	6341
diagnose . . . . .	6221	end of storage scan . . . . .	6344
DOS compatibility feature . . . . .	6207B, 6207C	forms control . . . . .	6339
emulator program entry . . . . .	6220	gather diagnose . . . . .	6342, 6343
gather (DOS compatibility feature) . . . . .	6207C	group mark word mark scan . . . . .	6344
I/O M, L, U operations . . . . .	6210	indexing . . . . .	6308
increment-decrement . . . . .	6206	input/output . . . . .	6334
index add . . . . .	6227	instruction fetch . . . . .	6301-6308
index factor fetch . . . . .	6226	load . . . . .	6334
instruction fetch . . . . .	6200	move . . . . .	6334
load . . . . .	6209	move characters and edit . . . . .	6341
move . . . . .	6209	move data . . . . .	6324
move and binary code . . . . .	6224	multiply . . . . .	6316, 6320
move and binary decode . . . . .	6223	no operation . . . . .	6306
move characters . . . . .	6215	non-interruptible operation . . . . .	6306
multiply and divide . . . . .	6208	one-byte data service . . . . .	6337
multi-way branch . . . . .	6205	operation codes (matrix) . . . . .	6300
no operation . . . . .	6201	priority test and branch . . . . .	6310
read and punch column binary . . . . .	6225	relocate . . . . .	6221
scatter--gather . . . . .	6207	scatter diagnose . . . . .	6342
scatter (DOS compatibility feature) . . . . .	6207B	set word mark . . . . .	6314
set selector channel to 1401 mode . . . . .	6217	store address register . . . . .	6312
set word mark . . . . .	6214	store and restore status . . . . .	6313
stacker select . . . . .	6212	subtract . . . . .	6316, 6317
store AAR or BAR . . . . .	6204	table lookup . . . . .	6315
suppress zeros . . . . .	6215	two-byte data service . . . . .	6338
tape operations . . . . .	6218	unit control operation . . . . .	6334
unit record operations . . . . .	6211	unit selection . . . . .	6336
zero and add . . . . .	6209	zero operations . . . . .	6324
zero and subtract . . . . .	6209	zero and add . . . . .	6325
1 and 2 byte data service . . . . .	6216	zero and subtract . . . . .	6325

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

© IBM Corp. 1966, 1970

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.

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  | Fair  | Poor  | Very  |
|                        | Good  |       |       |       | Poor  |
| As a text:             | ..... | ..... | ..... | ..... | ..... |
|                        | Very  | Good  | Fair  | Poor  | Very  |
|                        | Good  |       |       |       | 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.

REAL IS CL AMENT RM

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  | Fair  | Poor  | Very  |
|                        | Good  |       |       |       | Poor  |
| As a text:             | ..... | ..... | ..... | ..... | ..... |
|                        | Very  | Good  | Fair  | Poor  | Very  |
|                        | Good  |       |       |       | 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.

**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.

cut along this line

fold

fold

fold

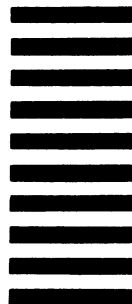
fold

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

fold

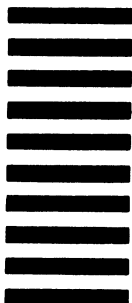
fold

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

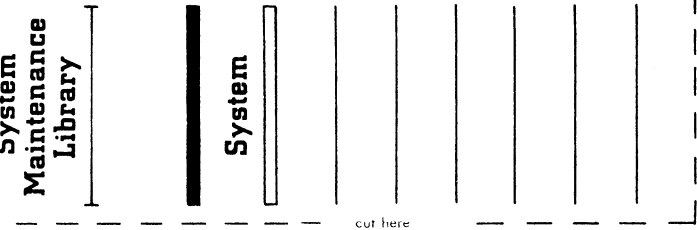


International Business Machines Corporation  
Field Engineering Division  
112 East Post Road, White Plains, N.Y. 10601

cut along this line



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



SY22-2842-3

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