IBM Field Engineering Education
Supplementary Course Material

SYSTEM/370

Model 165

Phase I

10-3

PREFACE

This publication is primarily intended for use by FE customer engineers enrolled in course 50028.

Second Edition (May 1971)
This is a major revision of, and makes SR23-3475-0 obsolete.

Address any comments concerning the contents of this publication to:
IBM, Field Engineering Education Media Development Center, Dept 927,
Rochester, Minnesota 55901

FIS Topics

- Introduction and Overview
- I-Unit
- E-Unit
- SCU
- Microdiagnostics
- Maintenance Console Maintenance

Lecture Topics

- Logout
- System Diagnostics
- System Maintenance
- Power
- Cooling
- PM
- M9A Storage Unit
- Console File
- Remote Analysis
- ECs (One Hour Lab to Support Prerequisite),
- NS Instructions
- Machine Language Program Applications
- Diagnostic Utilities

Figure 0-1 - Course Breakdown

Introduction and Overview

- Introduction
- System Components
- Maintenance Tools
- System Data Flow

I-Unit

- Functional Units
- Data Flow
- Error Analysis

E-Unit

- Functional Units
- Data Flow
- Error Analysis

SCU

- Functional Units
- Data Flow
- Display Capabilities
- Error Analysis

Maintenance Console Maintenance

Microdiagnostics

- Introduction
- Load Procedures
- Running and Error Analysis Procedures

Figure 0-2 - FIS Topics

The Model 165 is a large-scale, high-speed, general-purpose computing system with an internal performance of approximately two and a half times the Model 65, depending on the job stream. It offers high performance in both commercial and scientific applications. Fixed-point, floating-point, decimal, logical and I/O operations are performed as defined in System/360 Principles of Operation with the addition of extended precision floating-point, floating-point explicit rounding and the removal of operand boundary alignment restrictions for unprivileged operations. However, specifying off-boundary operands will result in severely reducing the speed of the system.

The Model 165 operates under OS/360, and can serve as a growth system for presently installed System/360 models. Upward compatibility (as defined in System/360 Principles of Operation) is maintained with all models of System/360.

The scientific and commerical throughput of the system is dependent on the size of main storage and high-speed buffer storage as well as the number, type and speed of the attached I/O devices.

Main storage for the Model 165 utilizes the M9A Basic Storage Module (BSM). A maximum of four M9A BSM can be housed in a 165 M9A storage frame. The M9A has a 2.0 microsecond cycle time.

Each 165 M9A storage frame can contain up to a maximum of 512 kilobytes. Frame capacity is determined by storage configuration. The available storage configurations for the 165 M9A storage are 500 kilobytes, 1.0 megabyte, 1.5 megabytes, 2.0 megabytes, and 3.0 megabytes.

Figure 0-3 - Introduction, page 1 of 2

All configurations are normally operated four-way interleaved but can be made two-way or serial under configuration control. To increase system throughput, all main storage references in the Model 165 are a doubleword, eight bytes wide, and instruction preparation is overlapped with instruction execution.

The Model 165 makes optimum use of the large main storage by means of a high-speed (C40) buffer storage in the storage control unit (SCU). The basic eight-kilobyte buffer storage retains main-storage data with which the CPU is currently working. The processor achieves high performance by working mostly with the high-speed buffer storage rather than with main storage. Buffer storage is transparent to the program.

The Model 165 also makes optimum use of the interleaved characteristic of its main storage in behalf of I/O operations by means of an I/O buffer in the SCU. The I/O buffer provides dedicated buffering for each attached channel, allowing pending channel requests to access main storage concurrently with other channel or CPU requests when vying for different logical storages.

I/O operations are carried out through the 2860 selector channel, the 2880channel and the 2870 multiplexer channel via a single channel interface. The Model 165 channel interface allows attachment of two 2870s and any mix of 2860 and 2880 channels up to a total not exceeding seven addressable channels or five channel frames.

The high performance of the Model 165 is enhanced by the implementation of the logic in monolithic system technology (MST-4) circuitry. The system also uses a read only system storage (ROS), as well as a writable control storage (WCS) that provides residence for emulators and diagnostics.

Additional optional features include a high-speed multiply unit that allows faster fixed-point and floating-point multiply operations, 7074/7090 class emulators, an eight-kilobyte expansion of the high-speed buffer storage for a total of 16 kilobytes, and an additional channel feature allowing attachment of up to five additional addressable channels.

Figure 0-4 - Introduction, page 2 of 2

Fixed-length fields, such as halfwords and doublewords, must be located in main storage on an integral boundary for that unit of information. A boundary is called integral for a unit of information when its storage address is a multiple of the length of the unit in bytes. For example, words (four bytes) must be located in storage so that their address is a multiple of the number 4. A halfword (two bytes) must have an address that is a multiple of the number 2, and doublewords (eight bytes) must have an address that is a multiple of the number 8.

Storage addresses are expressed in binary form. In binary, integral boundaries for halfwords, words, and doublewords can be specified only by the binary addresses in which one, two, or three of the low-order bits, respectively, are zero (Figure 3). For example, the integral boundary for a word is a binary address in which the two low-order positions are zero.

Variable-length fields are not limited to integral boundaries, and may start on any byte location.

Figure 0-5 - Information Positioning

- Fix the CPU failures that are located by any type of card calling CE aid (microdiagnostics, FLTs, support documentation).
- Fix all power and cooling problems.
- Install all ECs.
- Fix all 2880 channel failures.
- Configurate storage.
- Fix all M9A storage unit failures.
- Fix maintenance console mechanical failures, including both microfiche viewers and the minnow file.

Figure 1-1 - Job Responsibilities

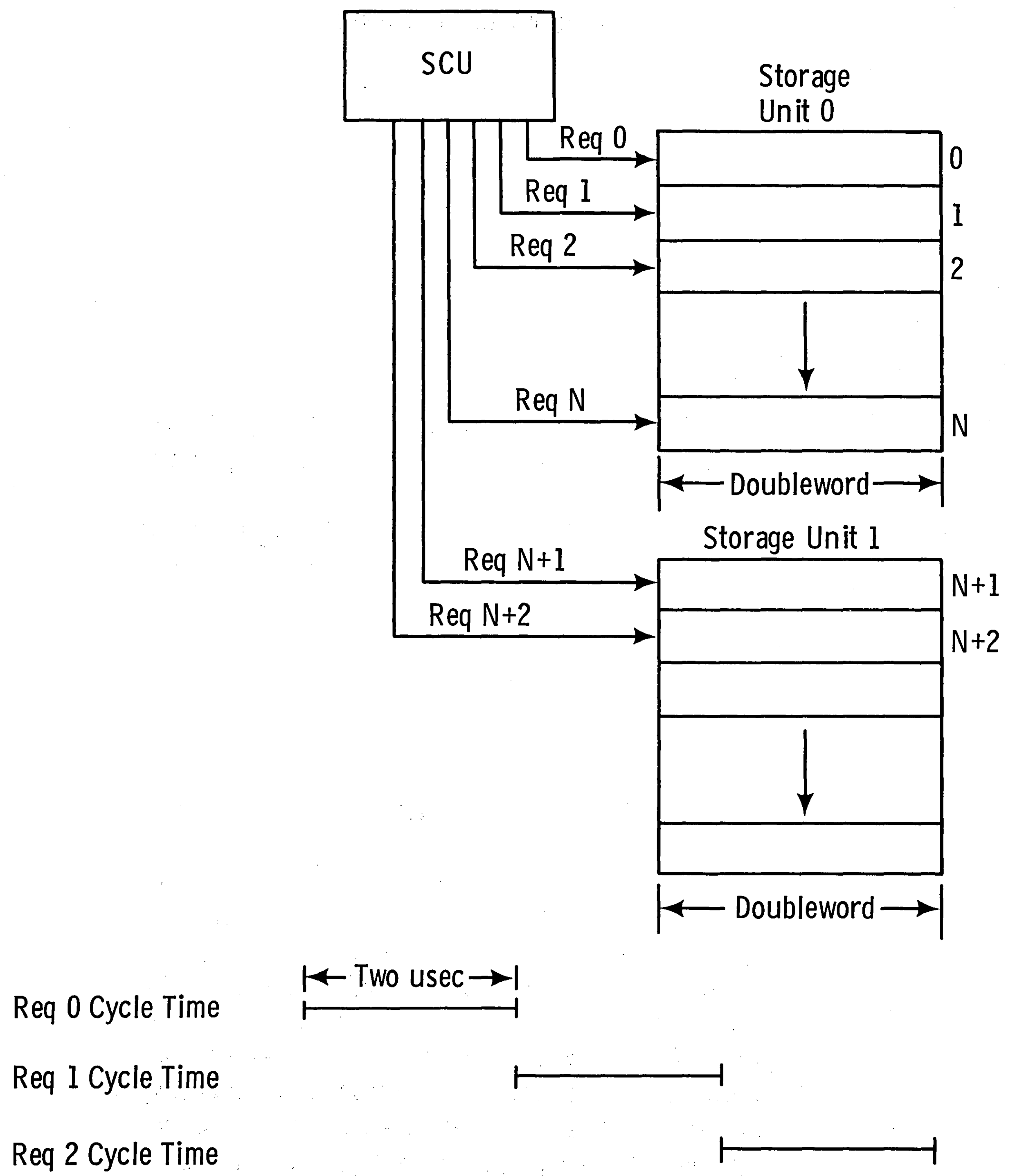


Figure 1-2 - Storage Addressing Number 1

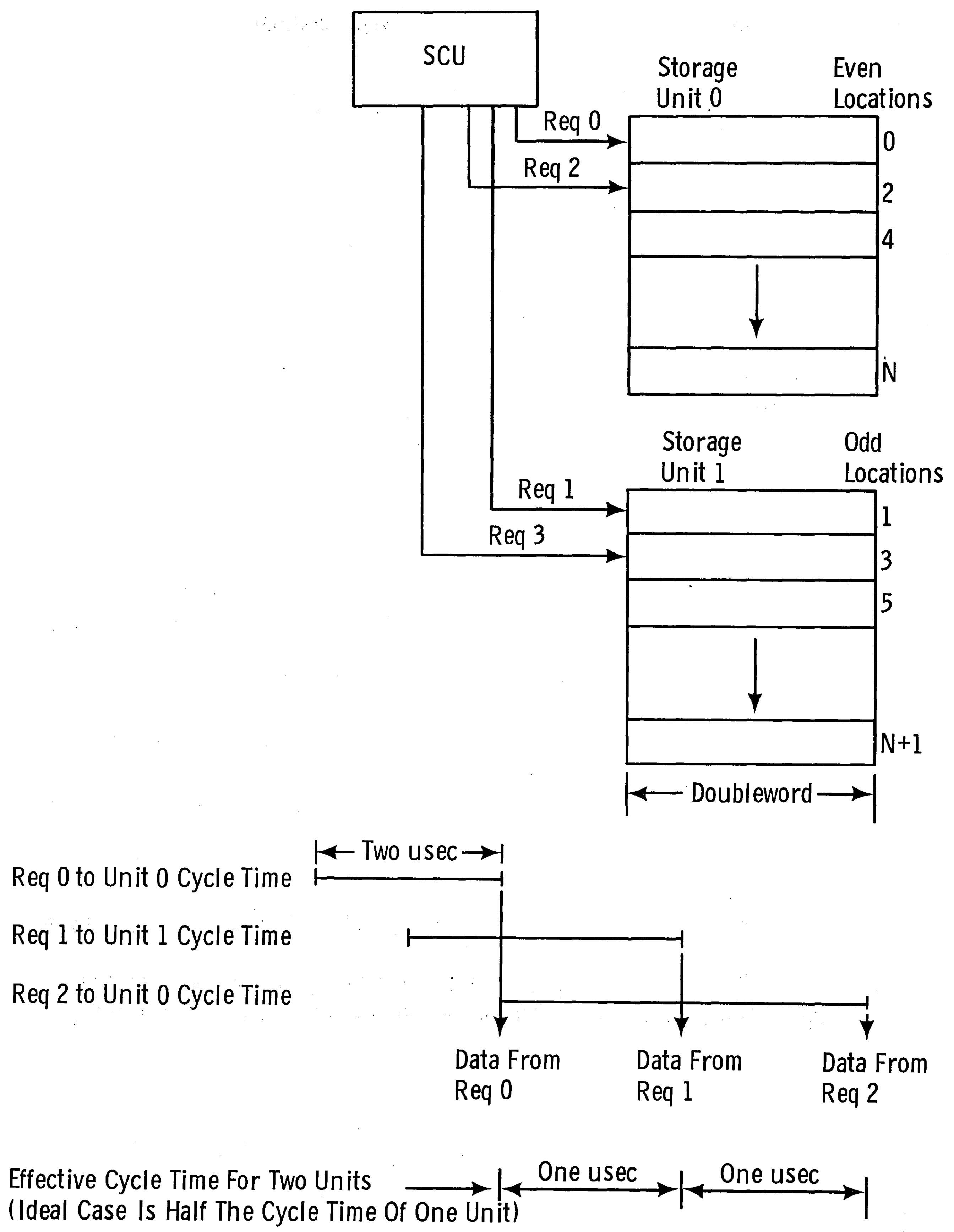


Figure 1-3 - Storage Addressing Number 2

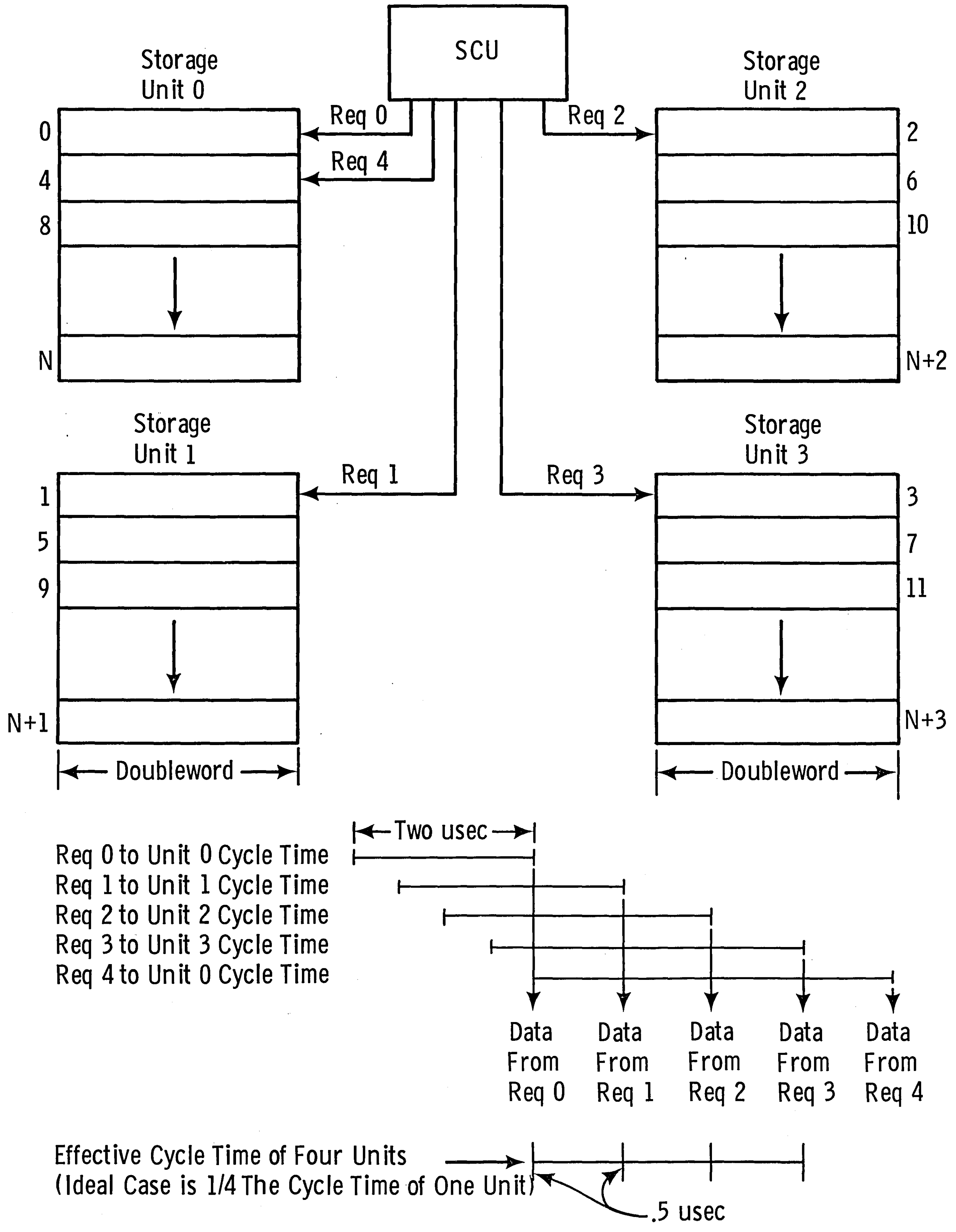


Figure 1-4 - Storage Addressing Number 3

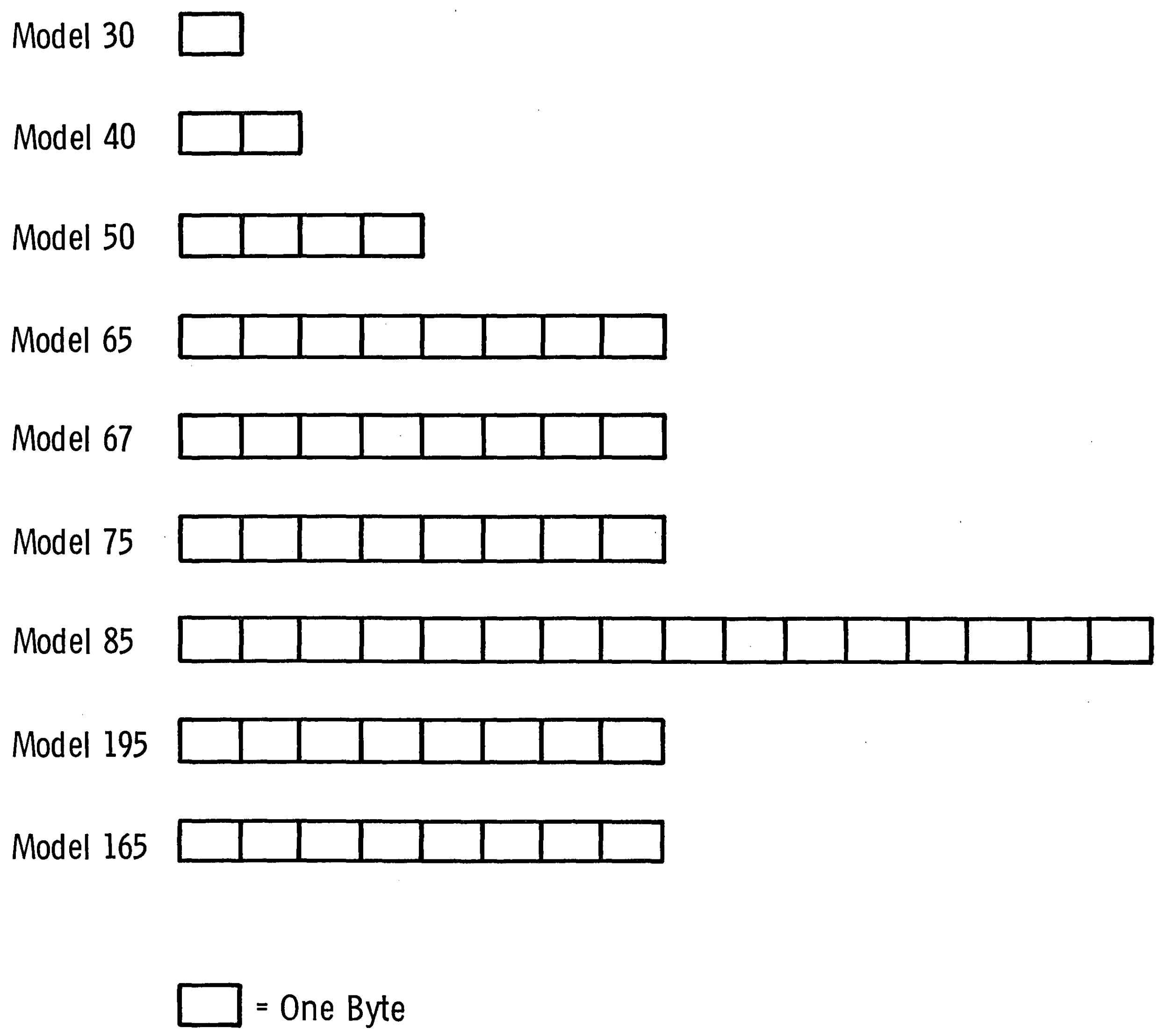


Figure 1-5 - Storage Access Size

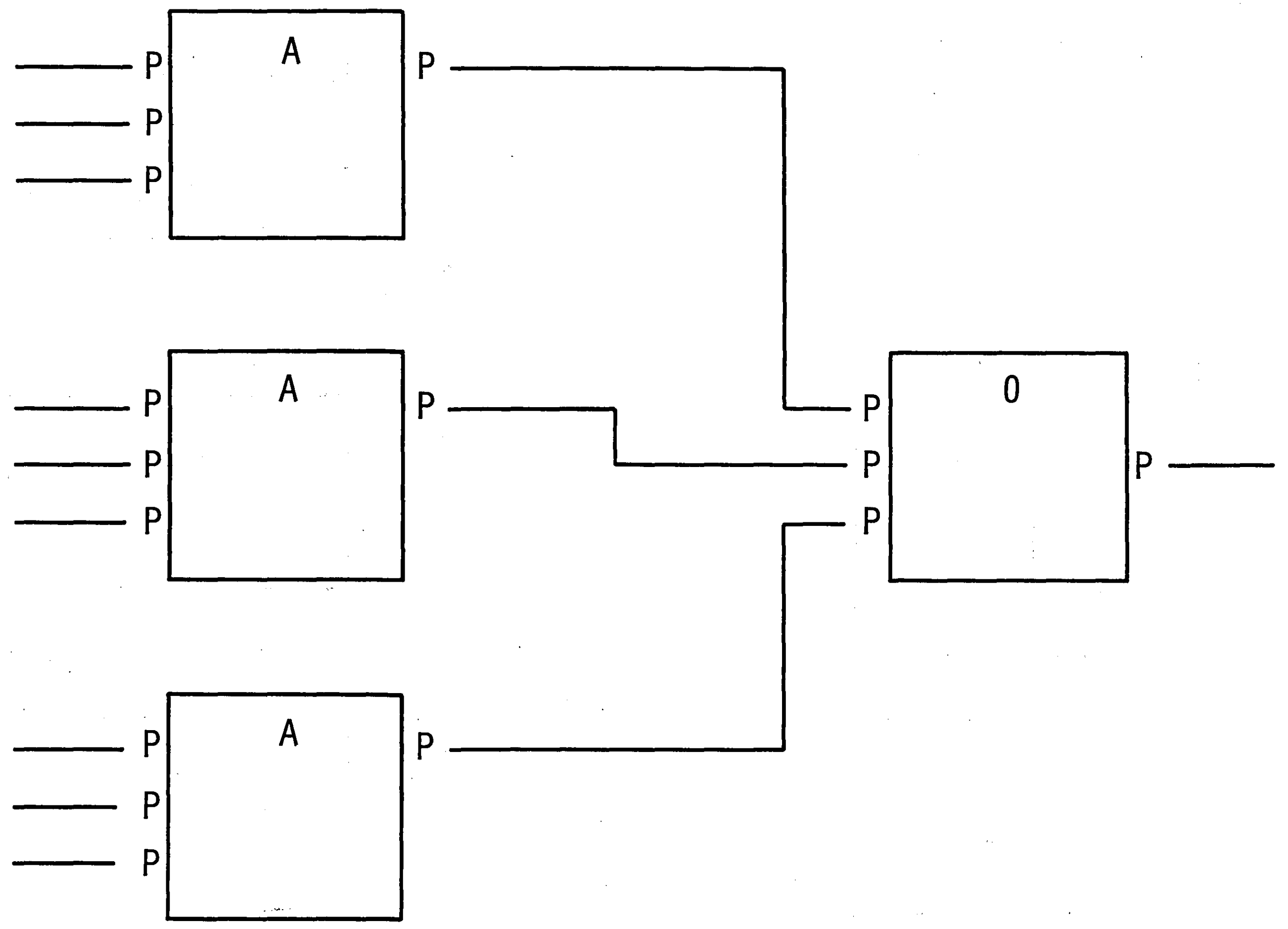


Figure 1-6 - SMS

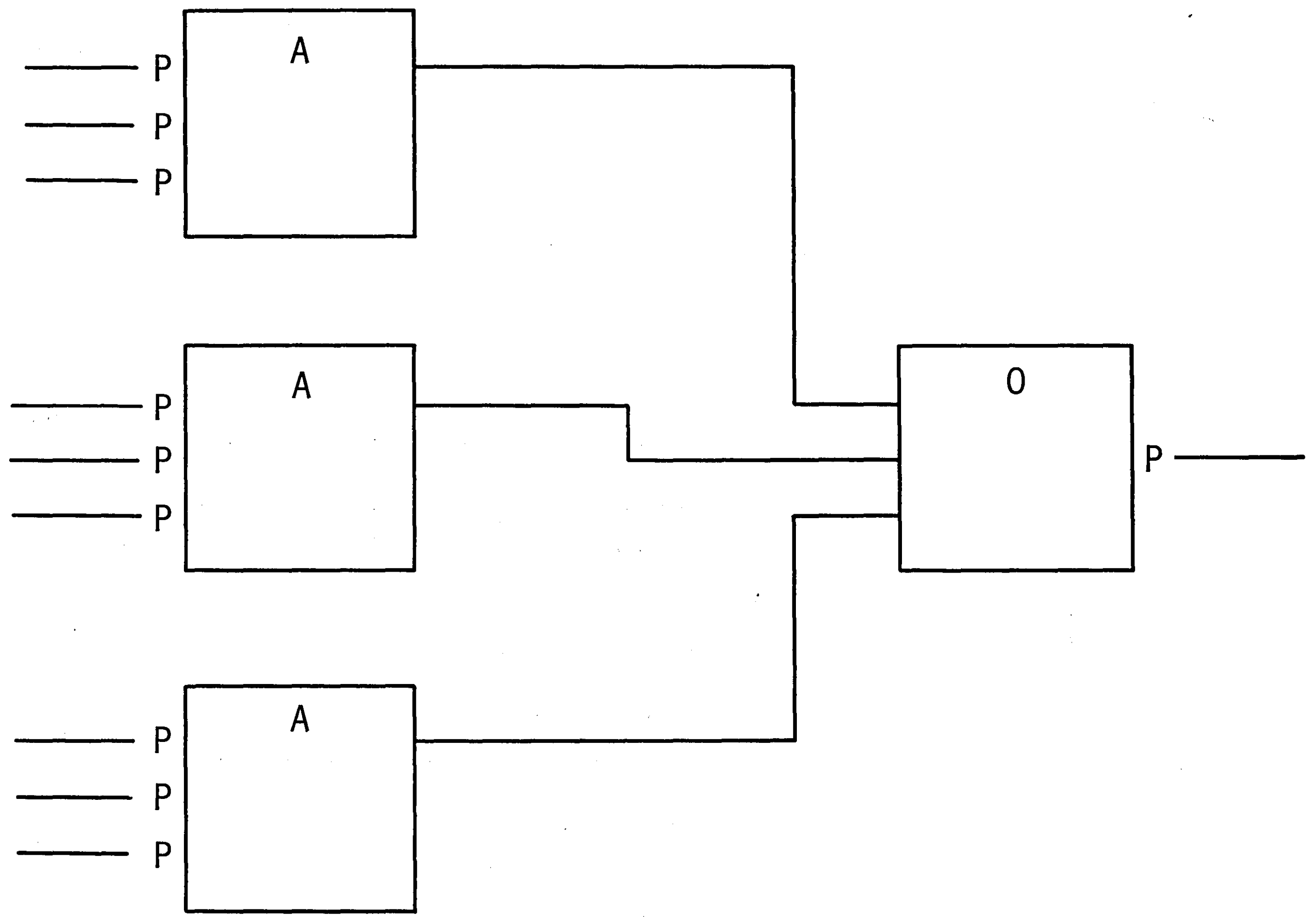


Figure 1-7 - SLT

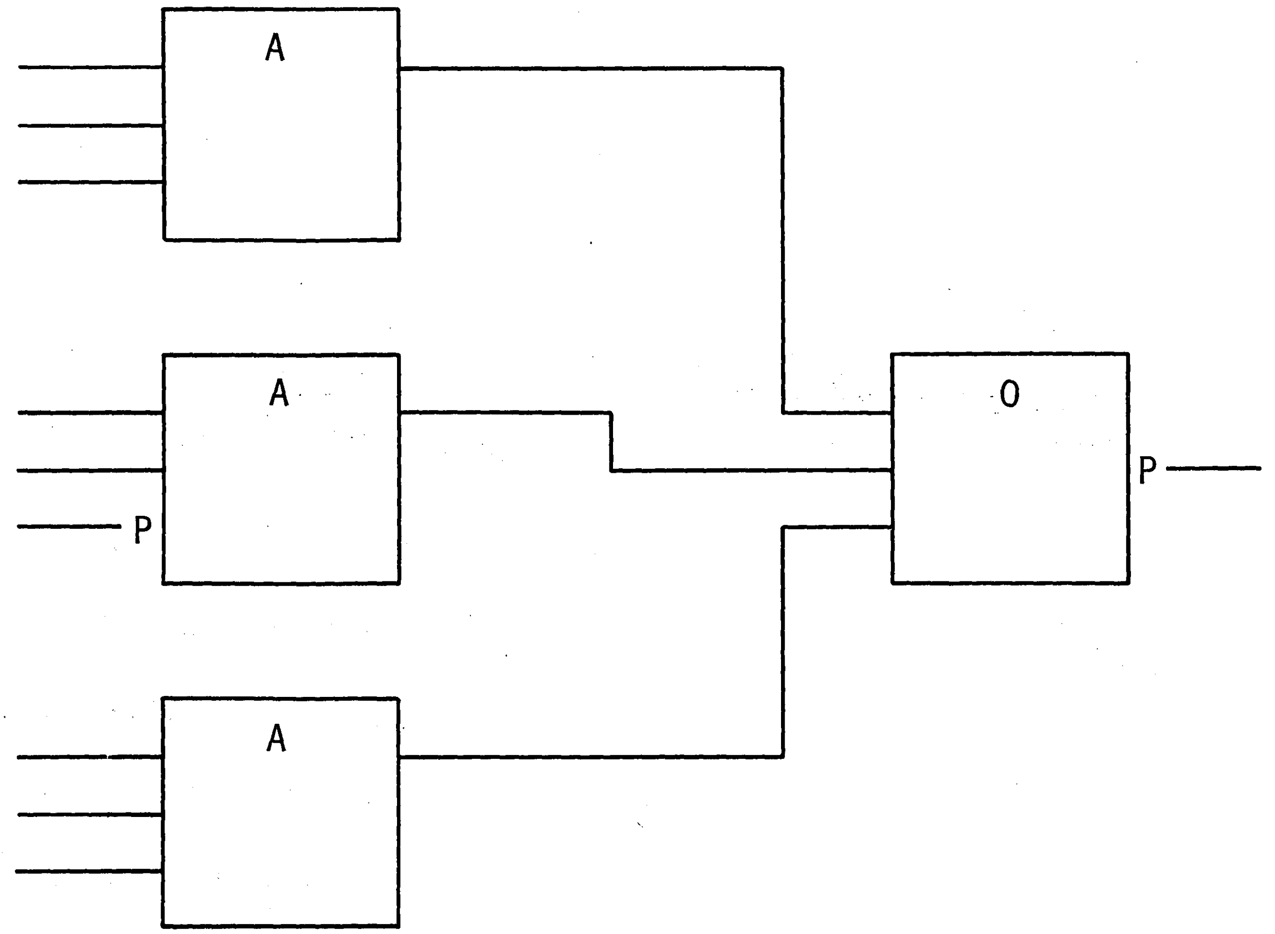


Figure 1-8 - MST

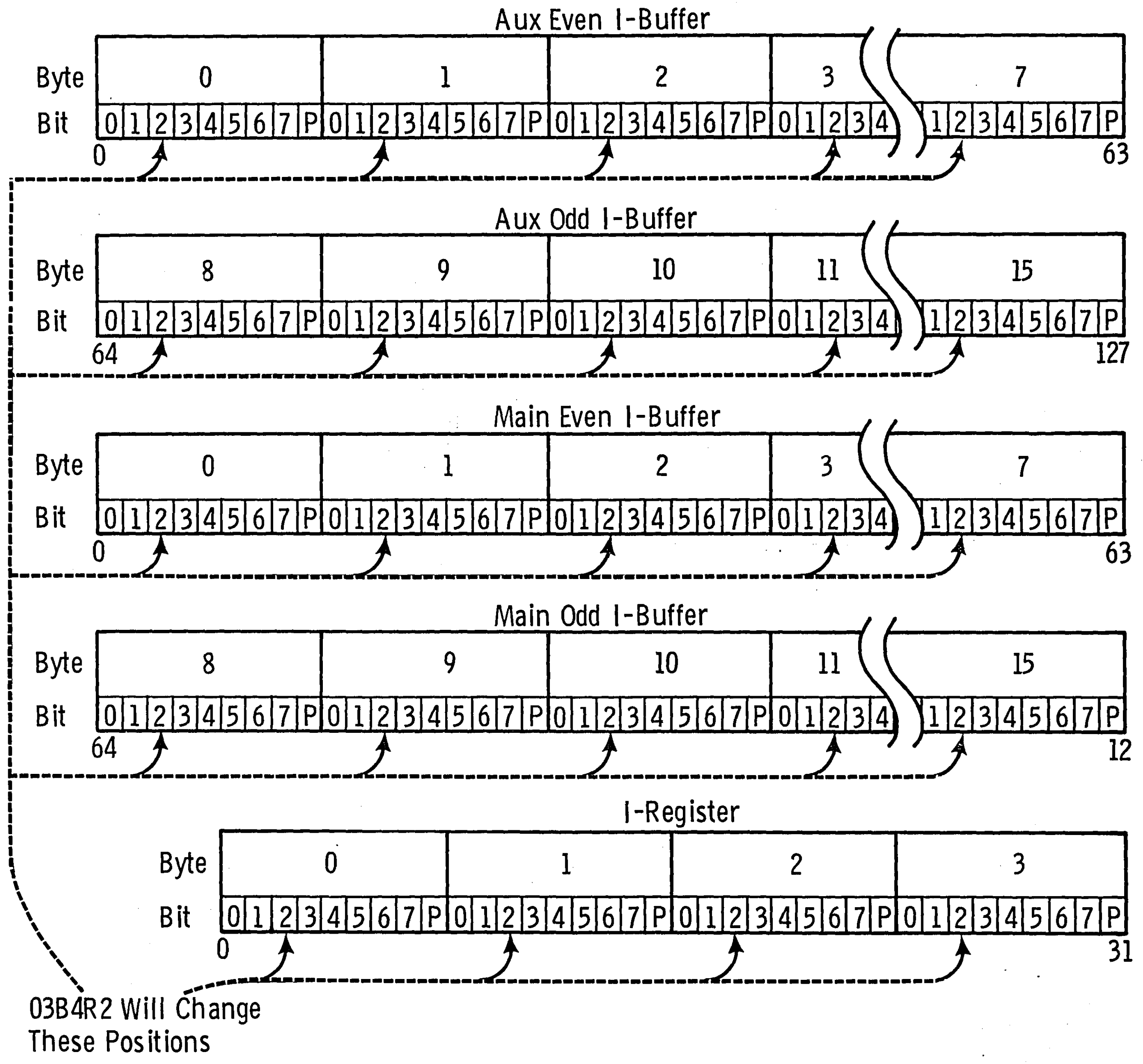


Figure 1-9 - Functional Packaging Example

1. Fix CPU failures that are located by any type of card calling CE aid (microdiagnostics, FLTs, support documentation).
2. Fix all power problems.
3. Fix all E-Unit failures.
4. Fix all cooling problems.
5. Install all ECs.
6. Fix all 2880 channel failures.
7. Configurate main storage.
8. Fix all maintenance console failures.
9. Fix all M9A storage unit failures.
10. Fix all SCU failures.
11. Fix maintenance console mechanical failures, including both microfiche viewers and the console file.
12. Fix all operators console-feature failures.

Figure 1-10 - Session 1 Question 3

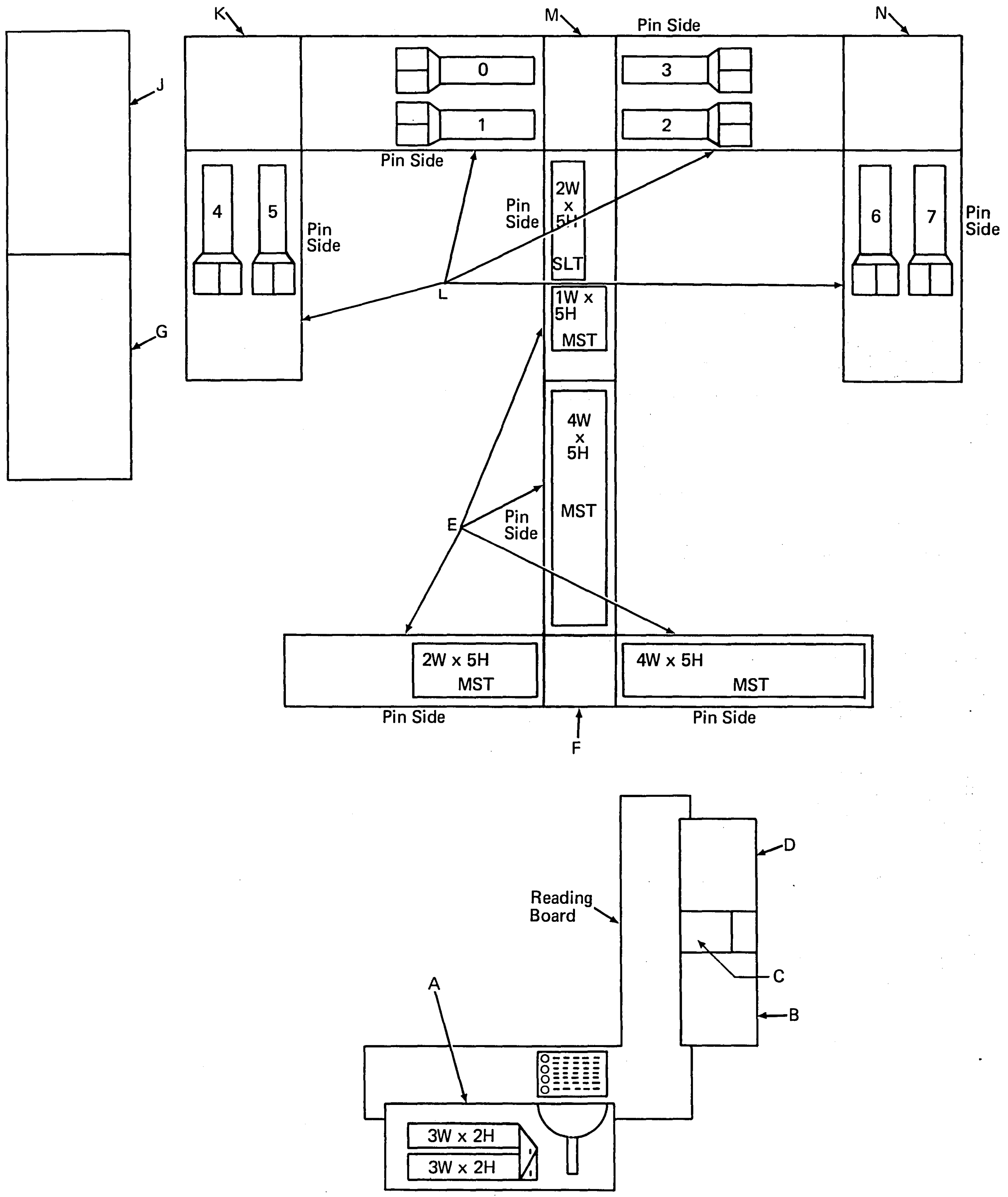


Figure 2-1 - Session 2 Questions 1-4

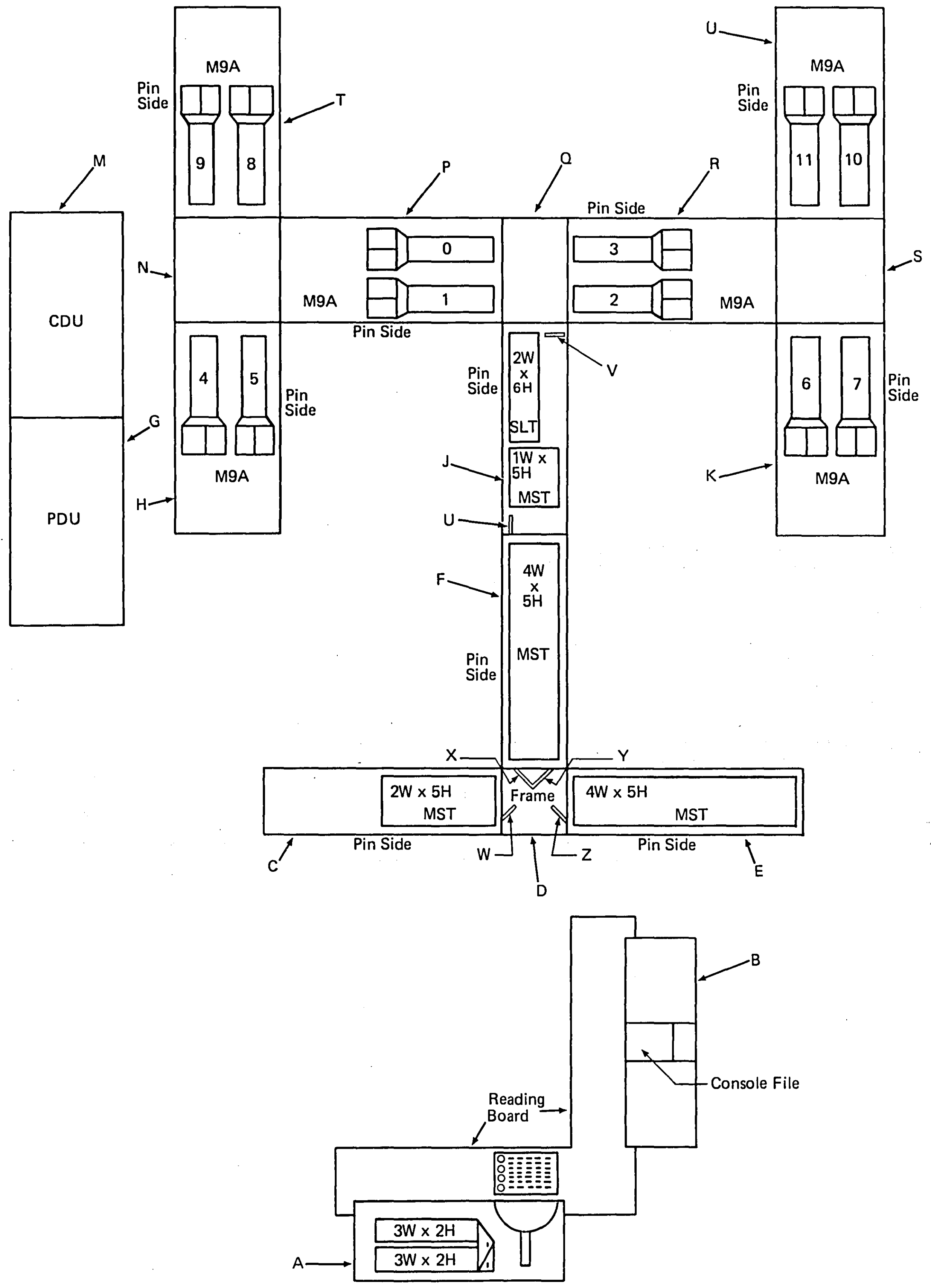


Figure 2-2 - Session 2 Questions 5-8

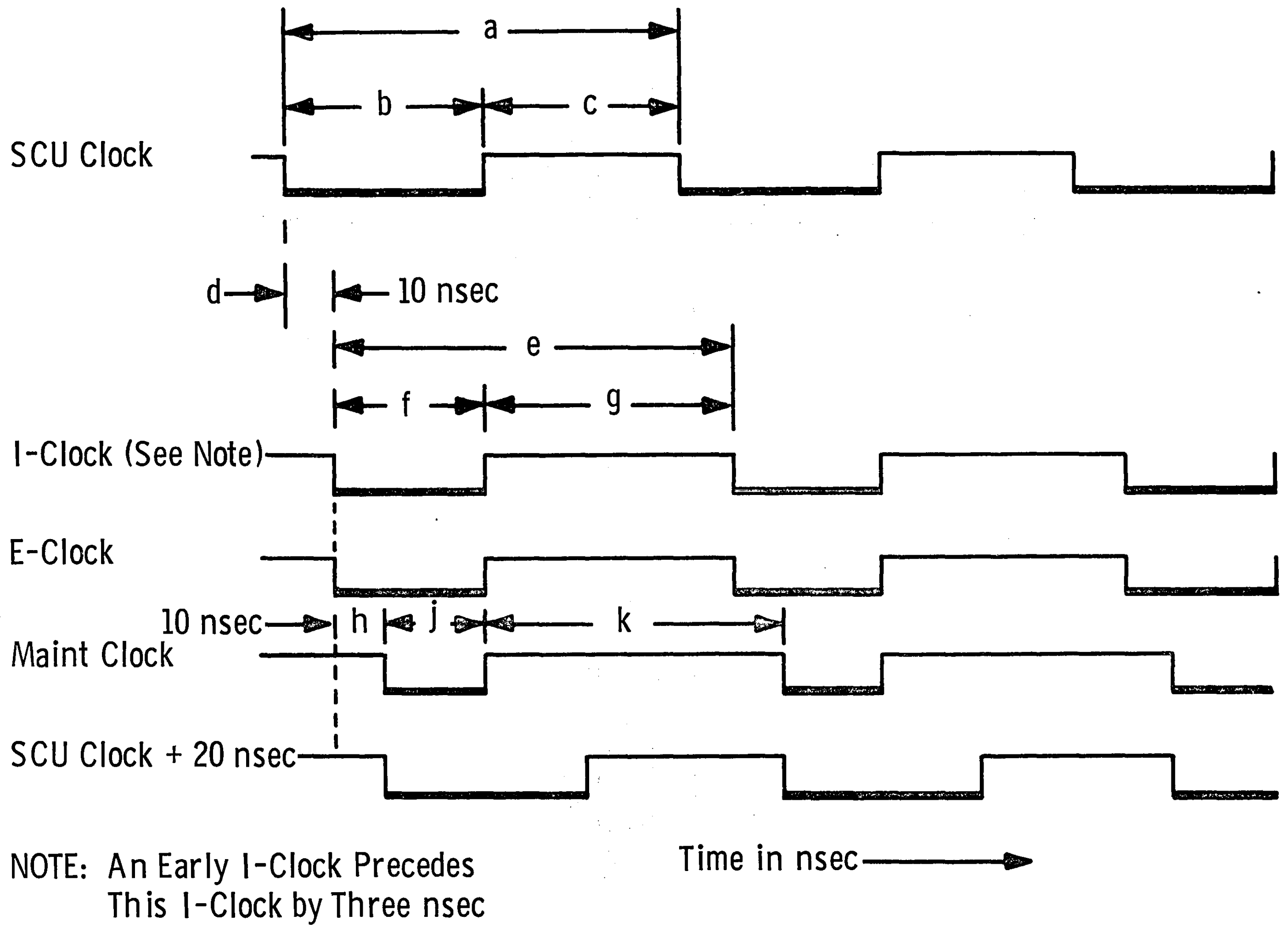


Figure 2-3 - Session 2 Questions 14-17

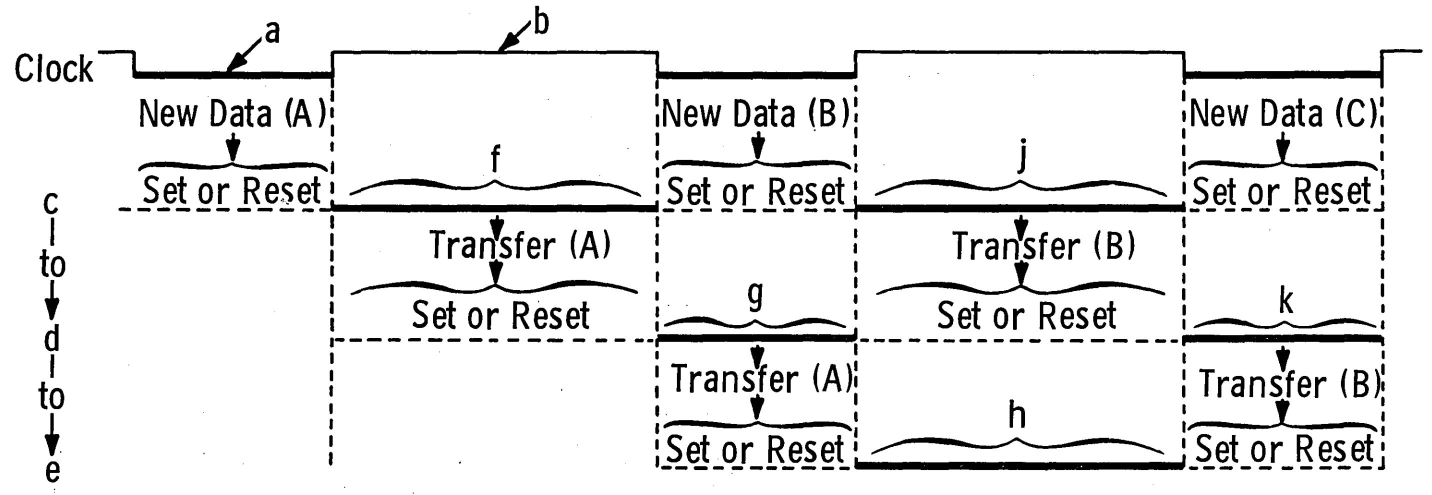


Figure 2-4 - Session 2 Questions 23-25

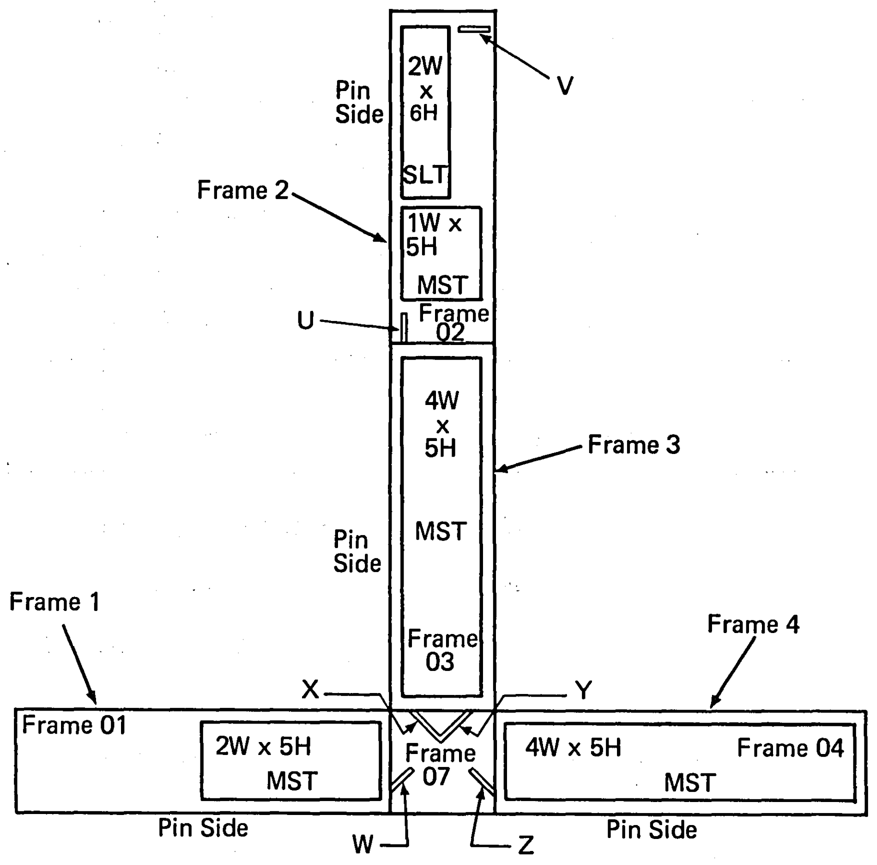


Figure 2-5 - CPU

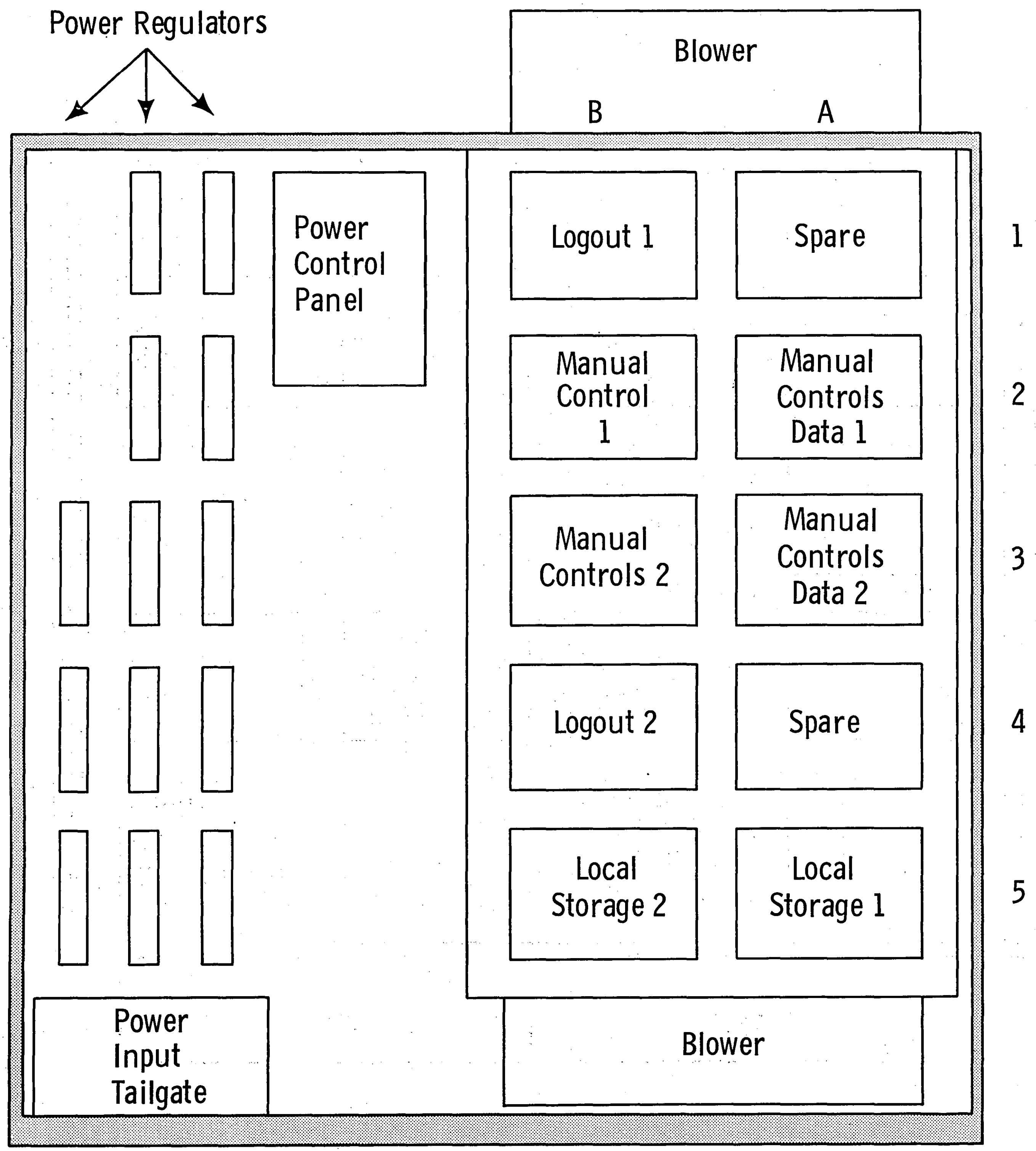


Figure 2-6 - Frame 01 Pin Side

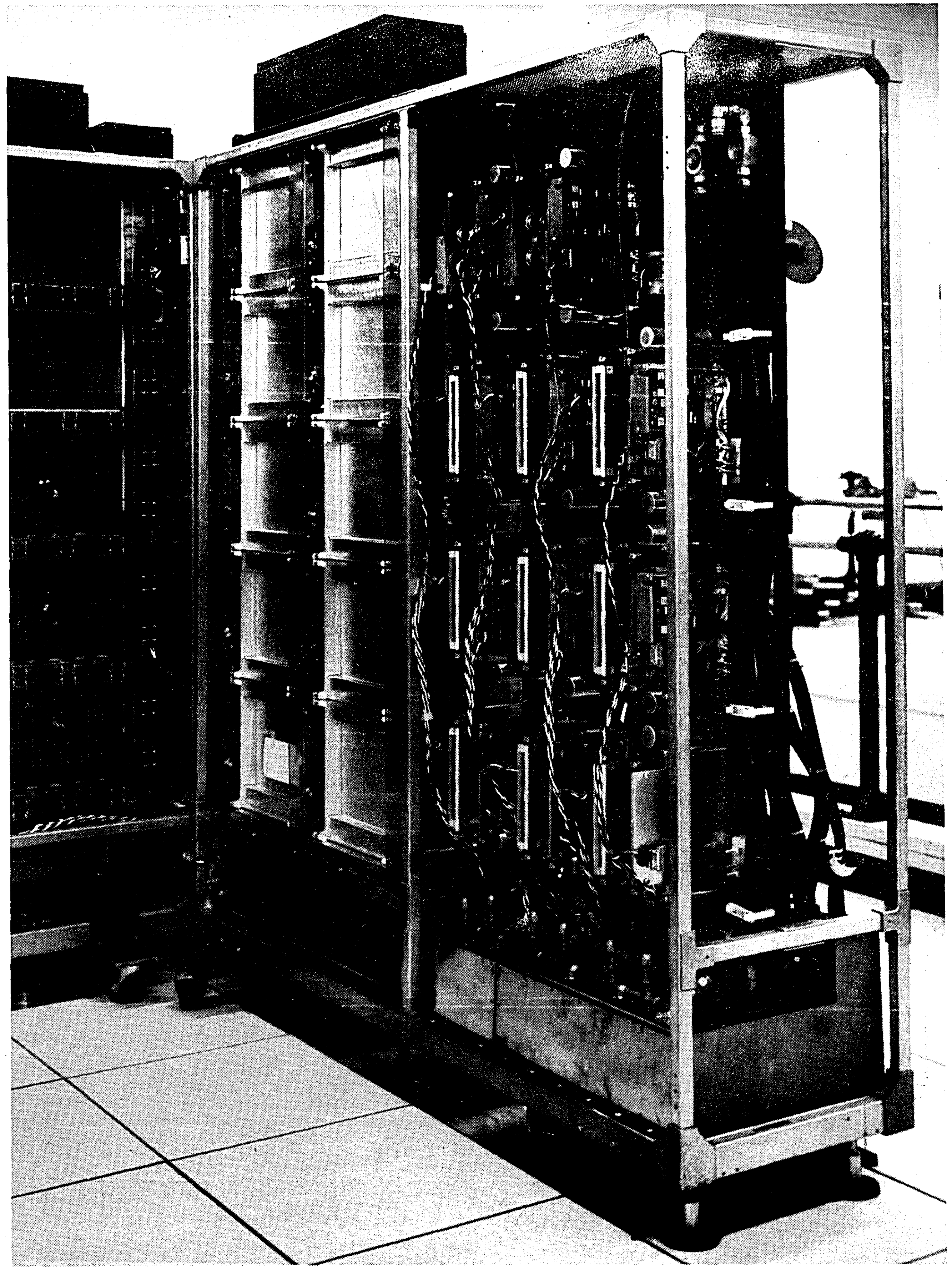


Figure 2-7 - Frame 1

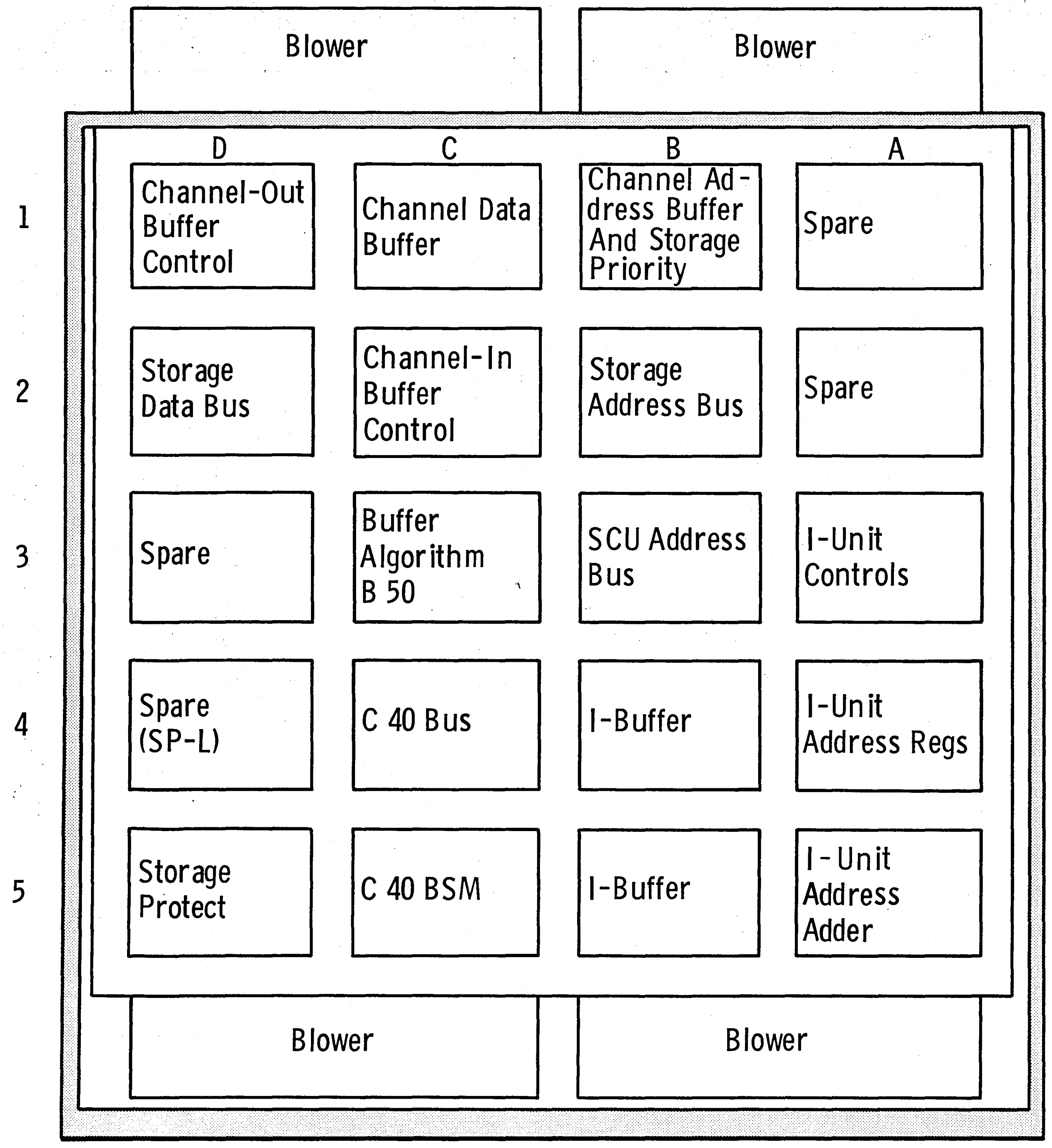
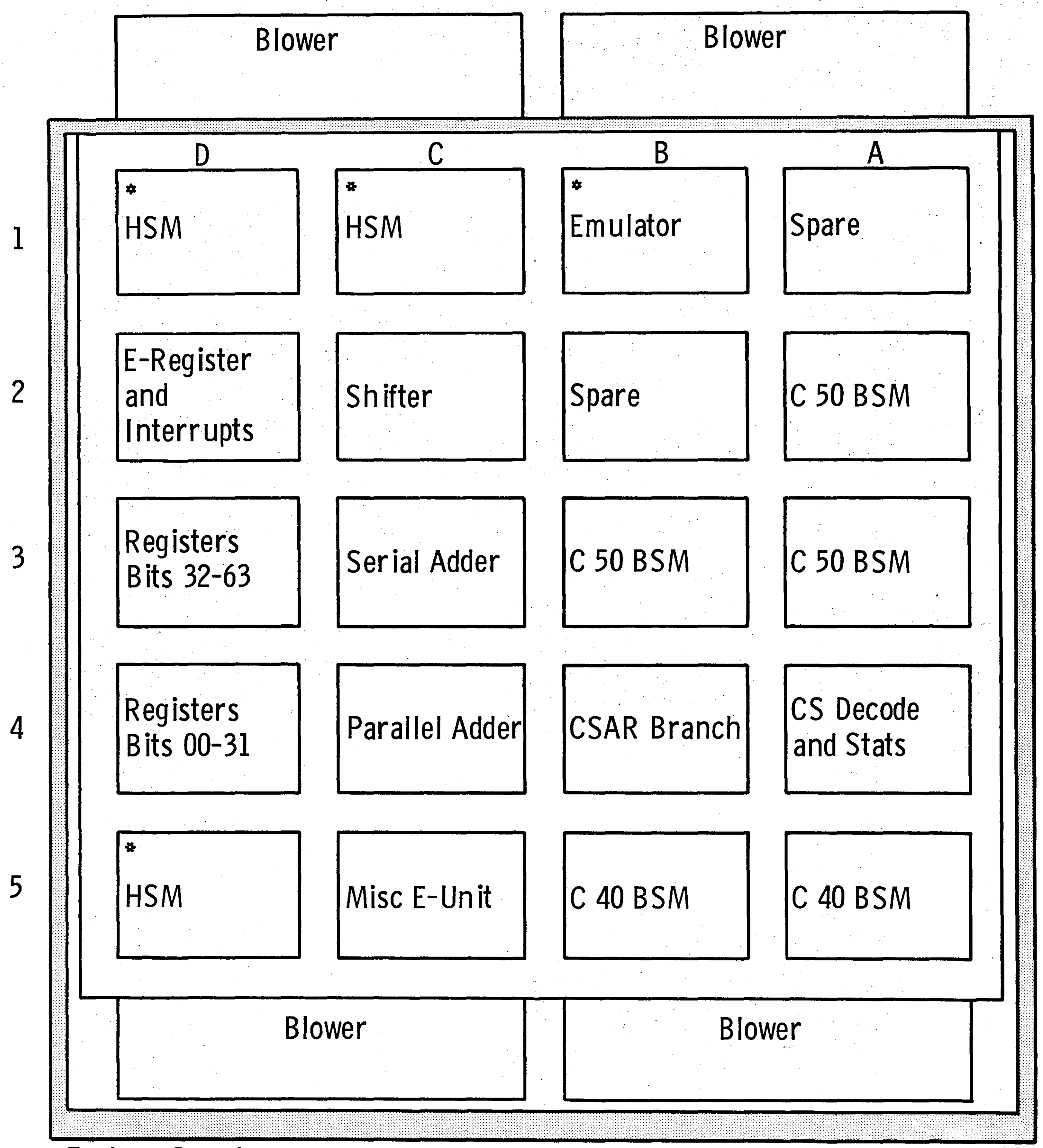


Figure 2-8 - Frame 03 Pin Side



*Feature Board

Figure 2-9 - Frame 04 Pin Side

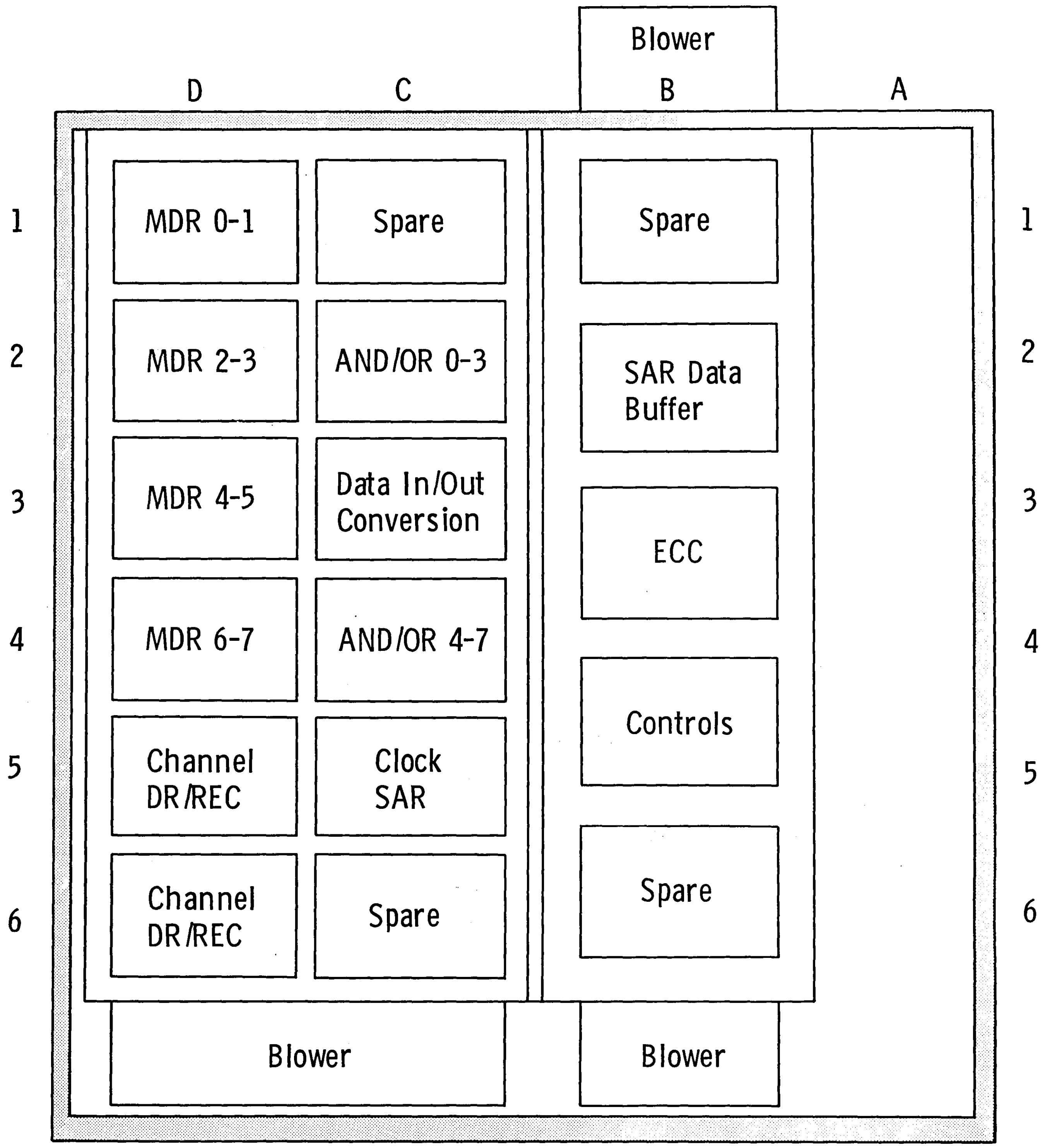


Figure 2-10 - Frame 02 Pin Side

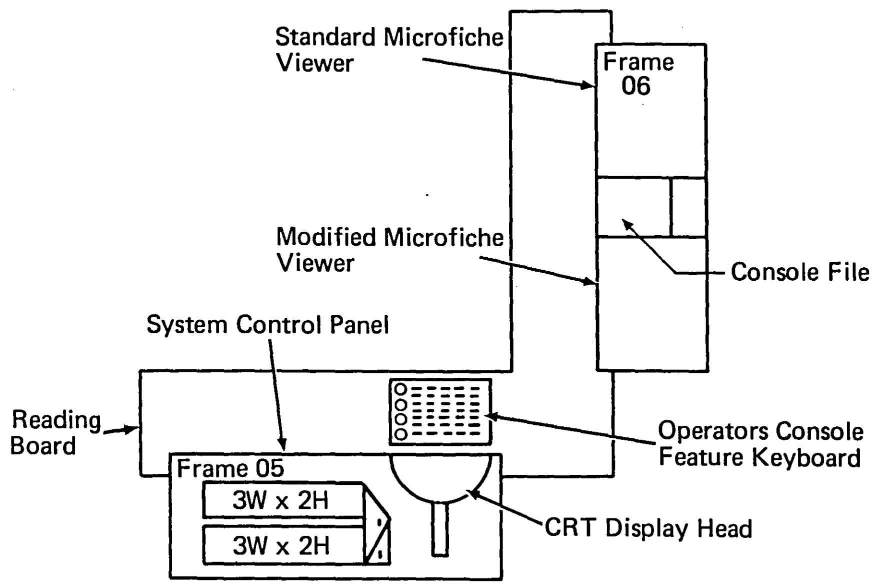
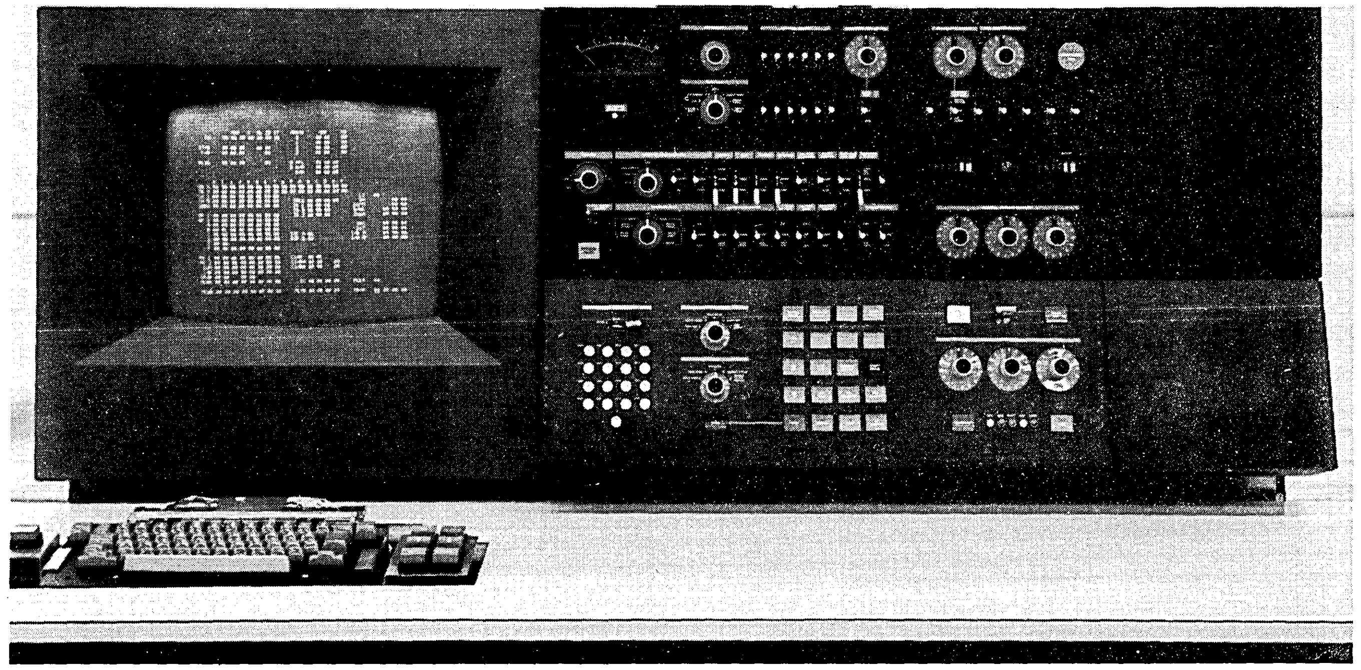


Figure 2-11 - Frames 5 and 6



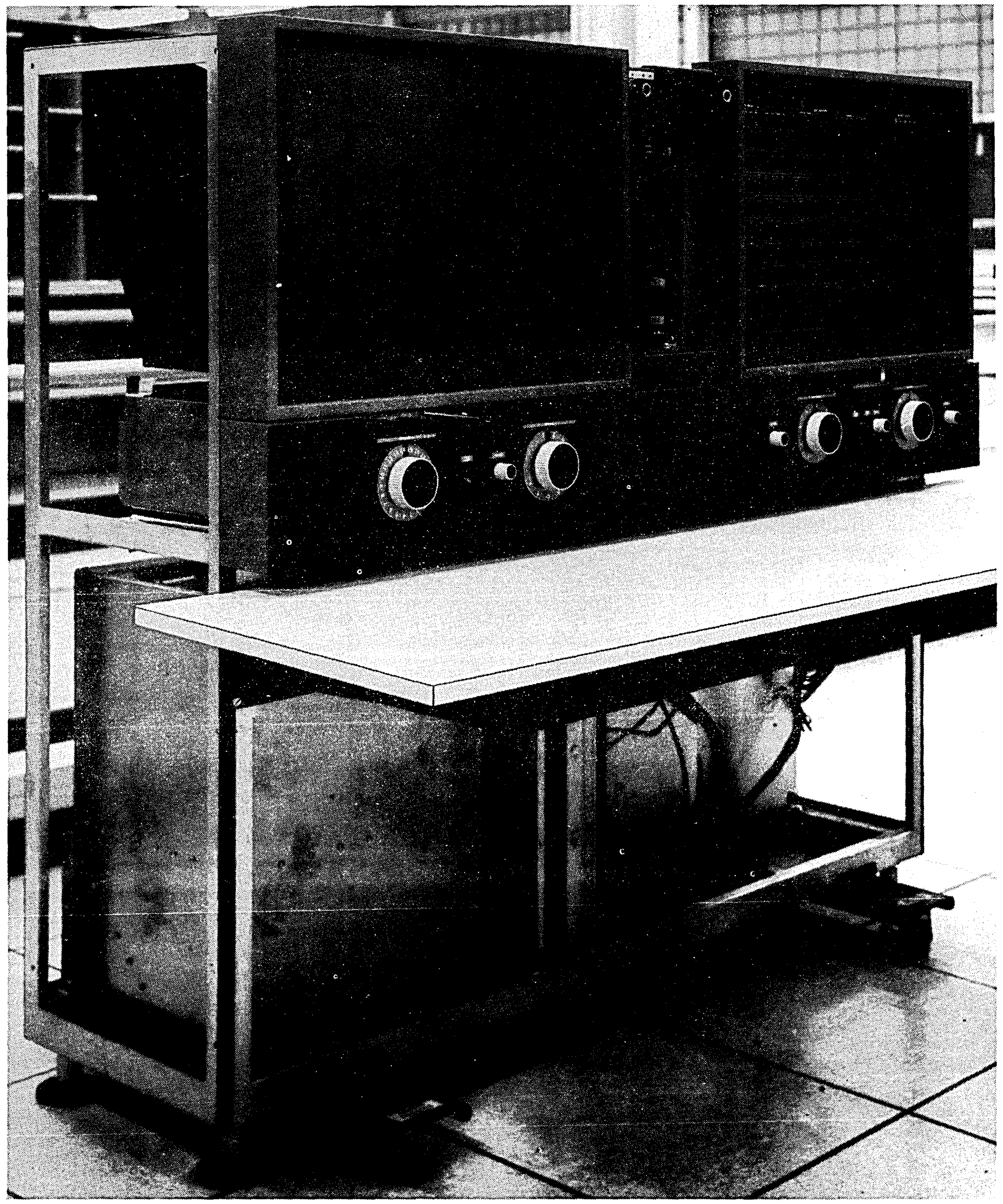


Figure 2-13 - Microfiche Viewers

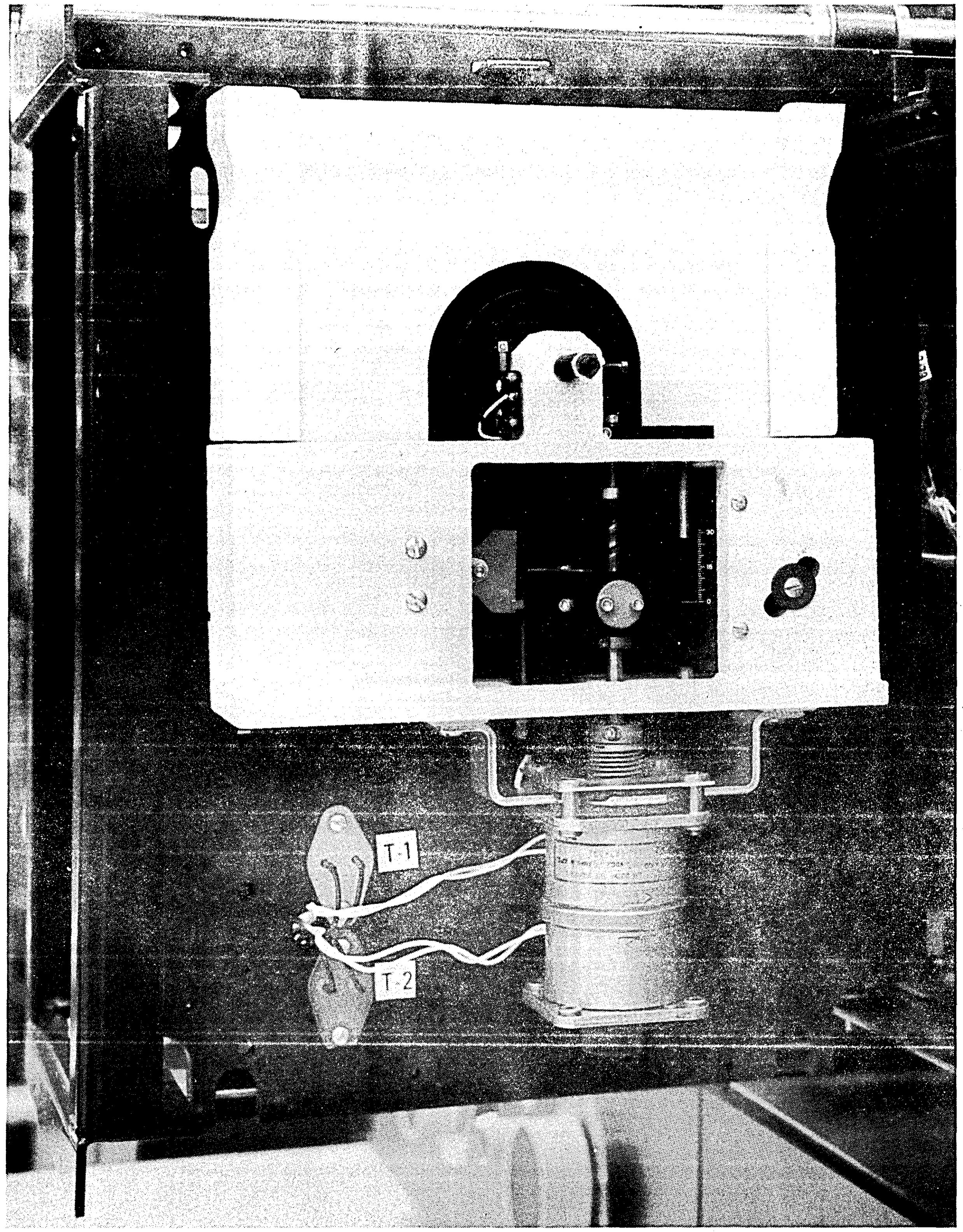


Figure 2-14 - Console File

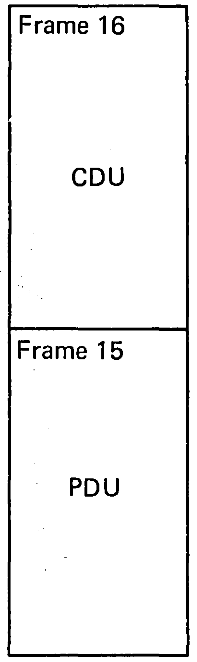


Figure 2-15 - Power and Cooling Frames

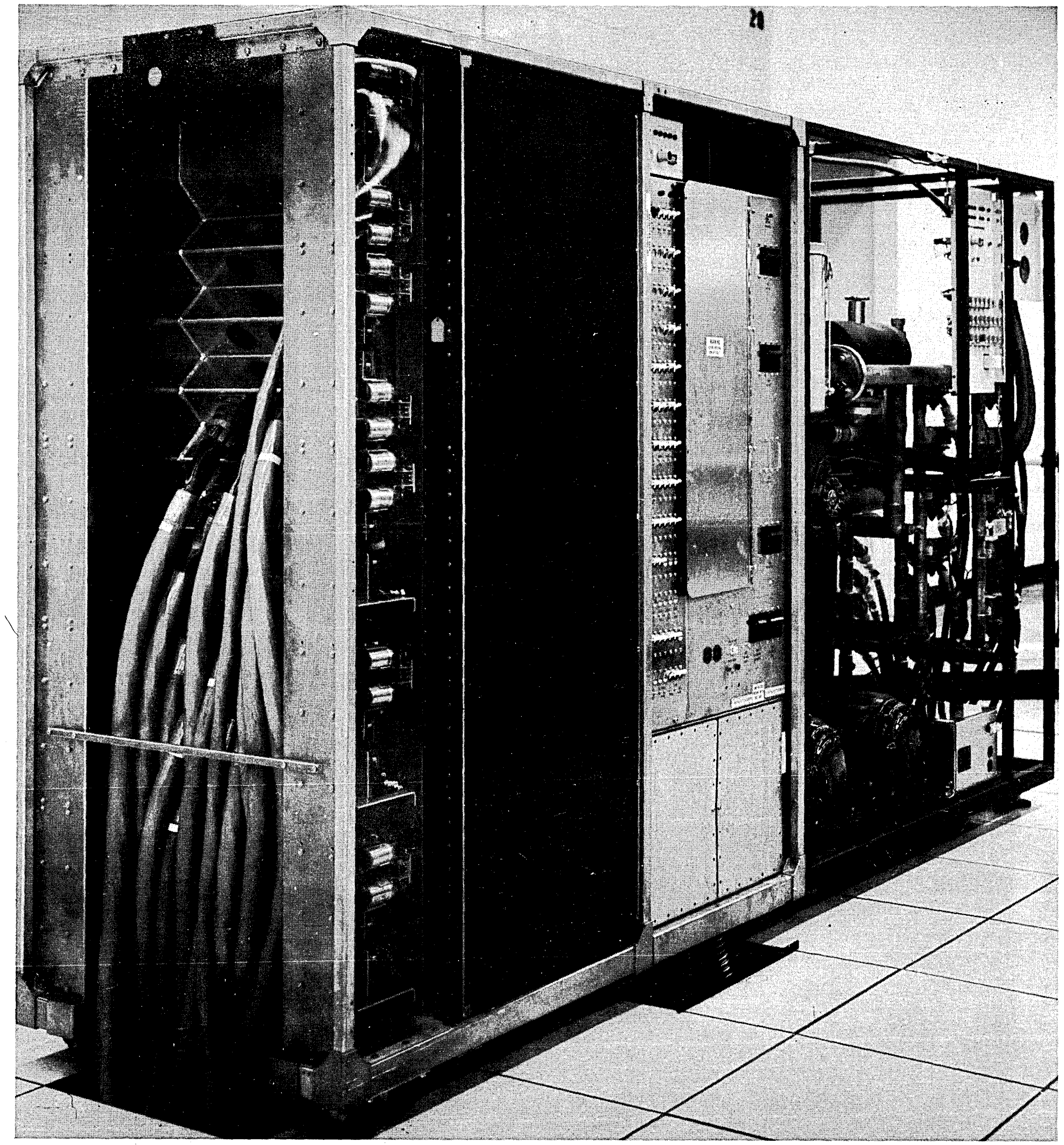


Figure 2-16 - CDU and PDU

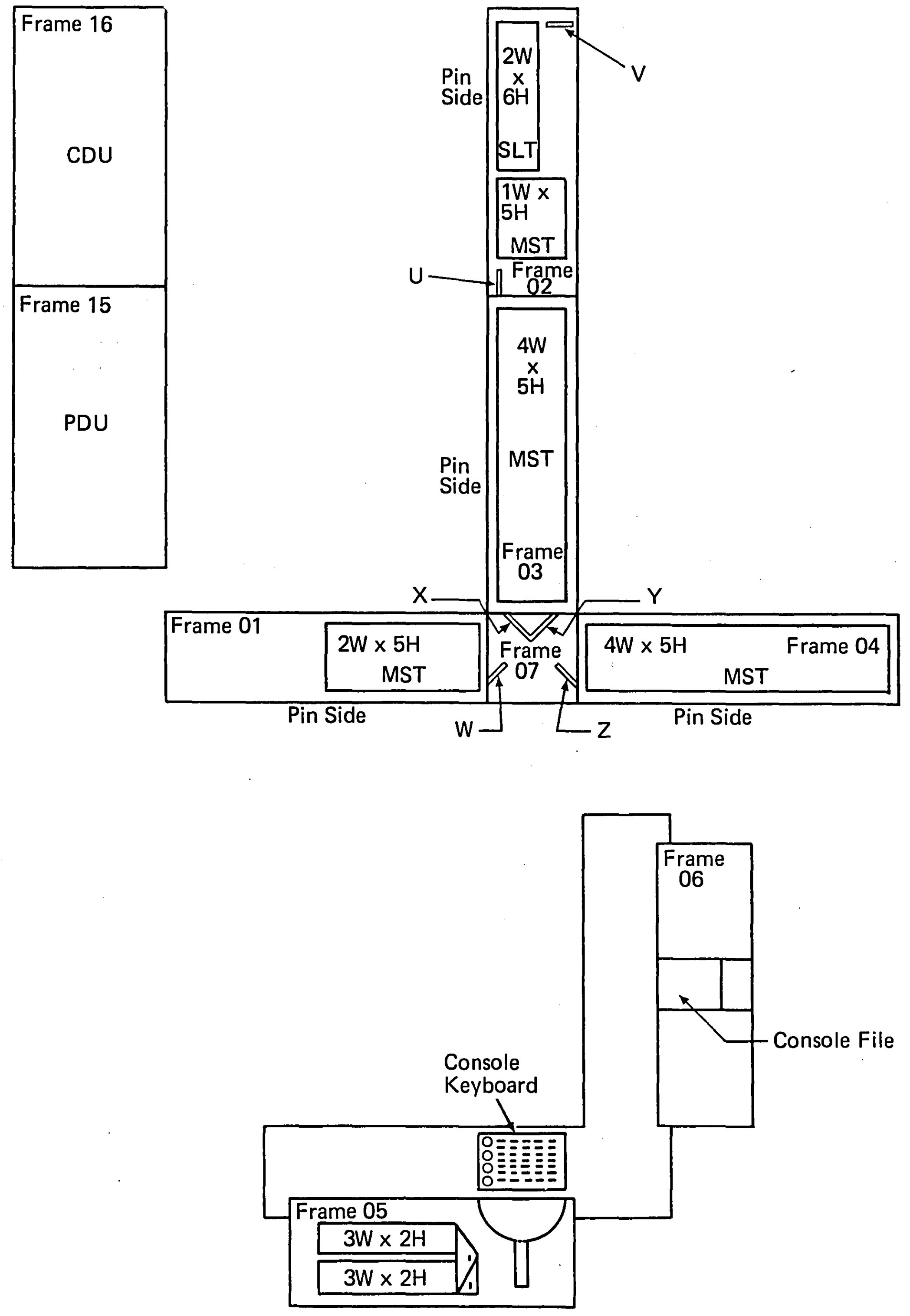
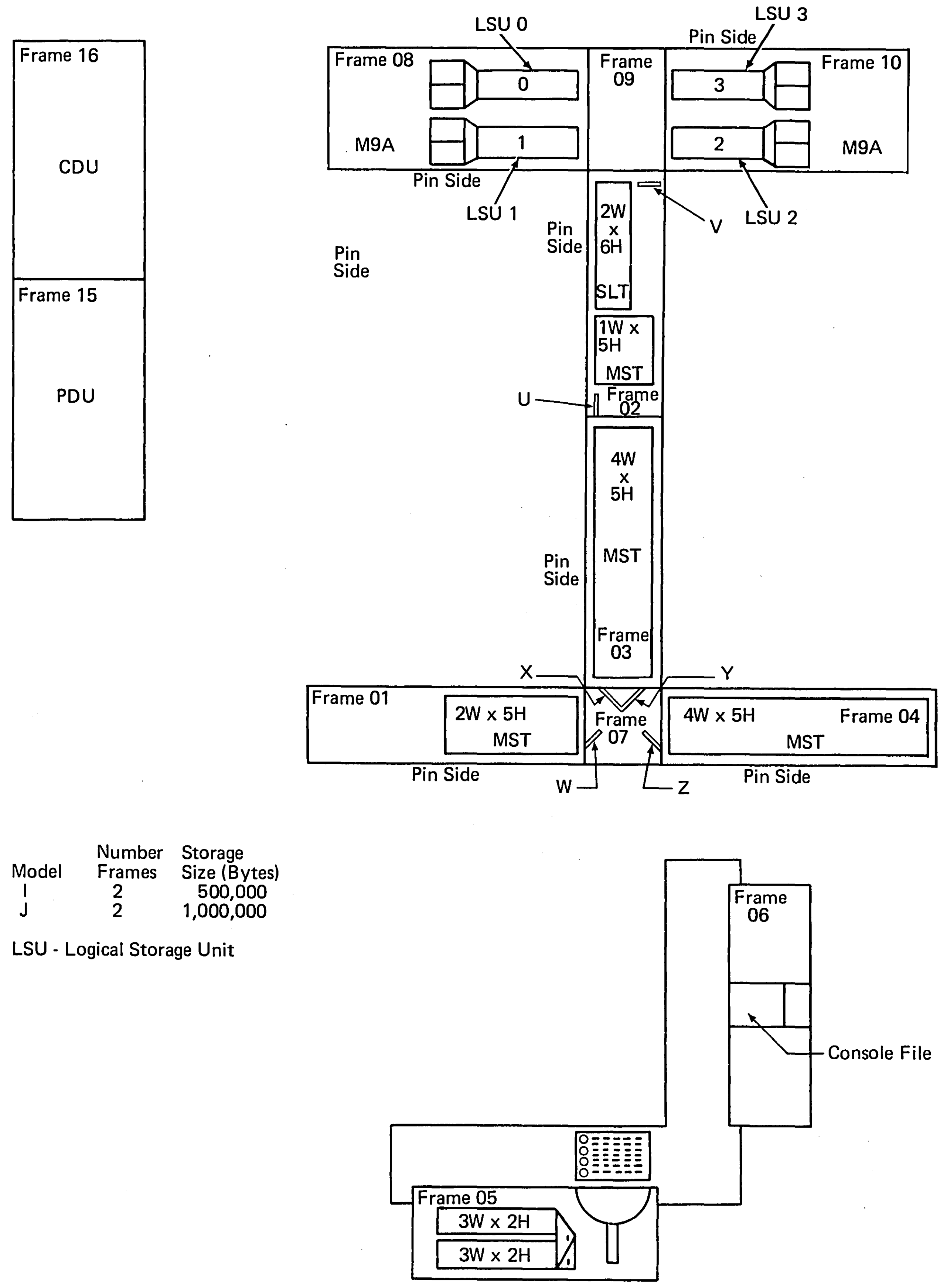


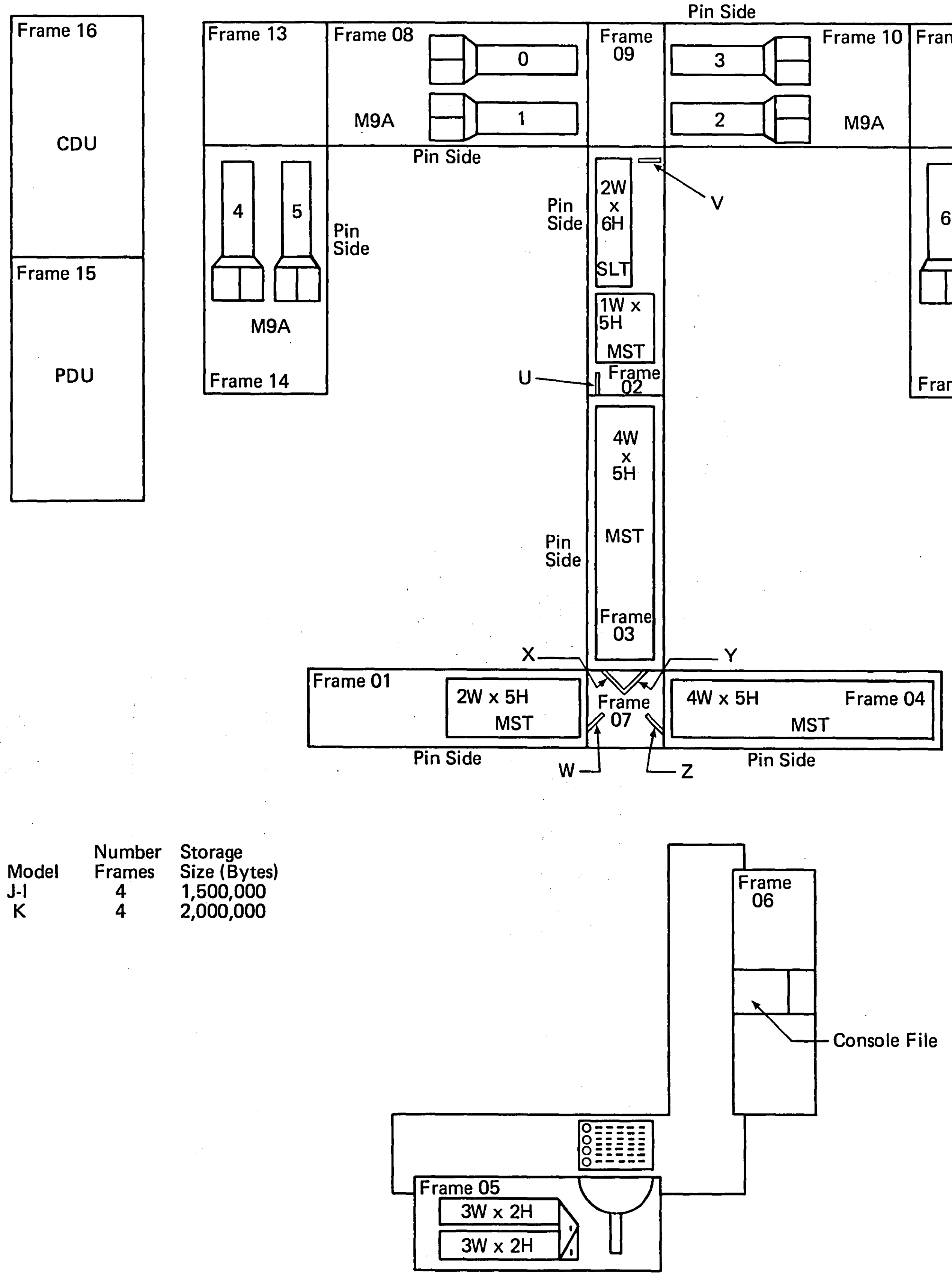
Figure 2-17 - System We Have Built



Model	Number Frames	Storage Size (Bytes)
I	2	500,000
J	2	1,000,000

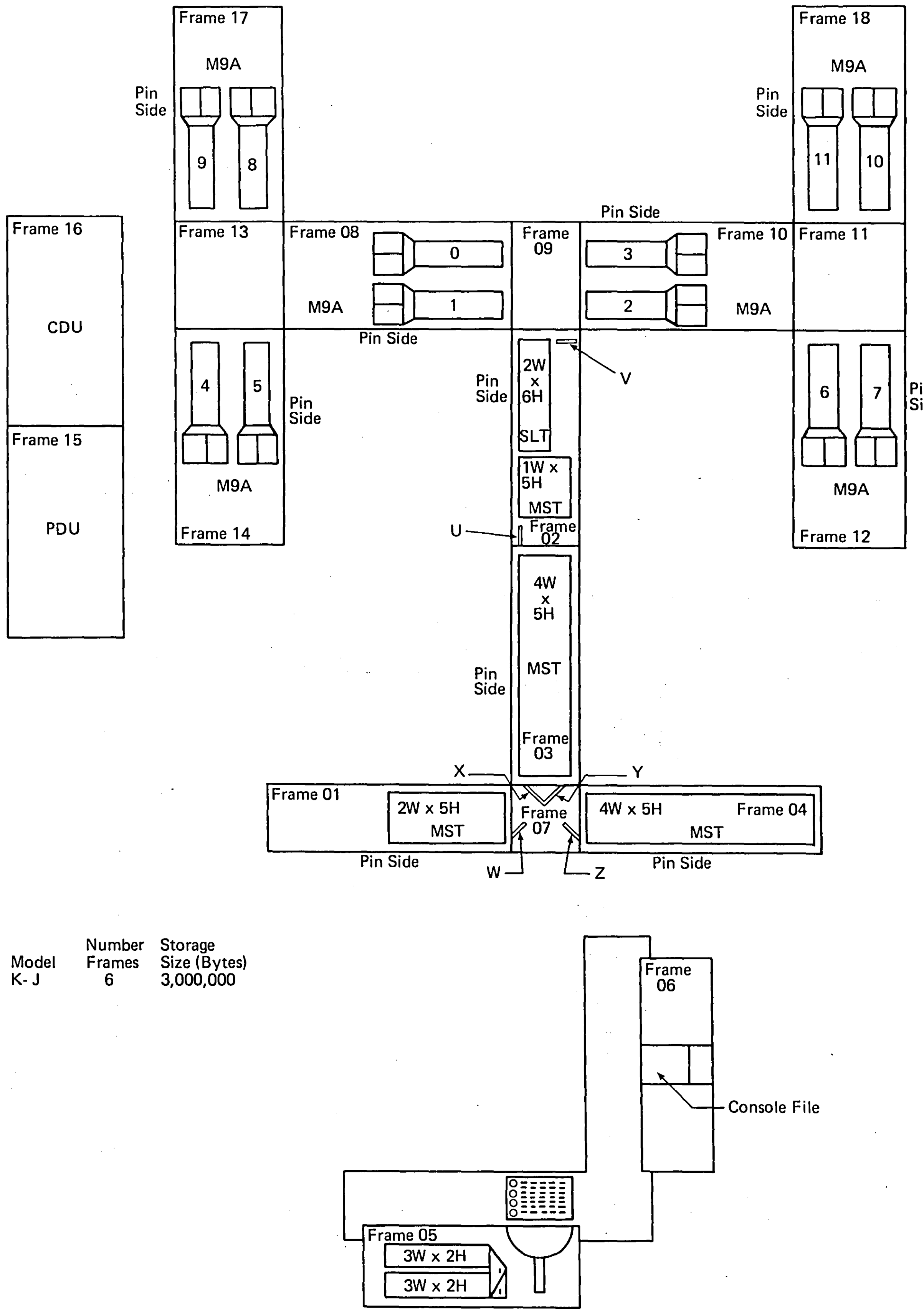
LSU - Logical Storage Unit

Figure 2-18 - I and J Storage



Model	Number Frames	Storage Size (Bytes)
J-I	4	1,500,000
K	4	2,000,000

Figure 2-19 - J-I and K Storage



Model	Number Frames	Storage Size (Bytes)
K- J	6	3,000,000

Figure 2-20 - K-J Storage

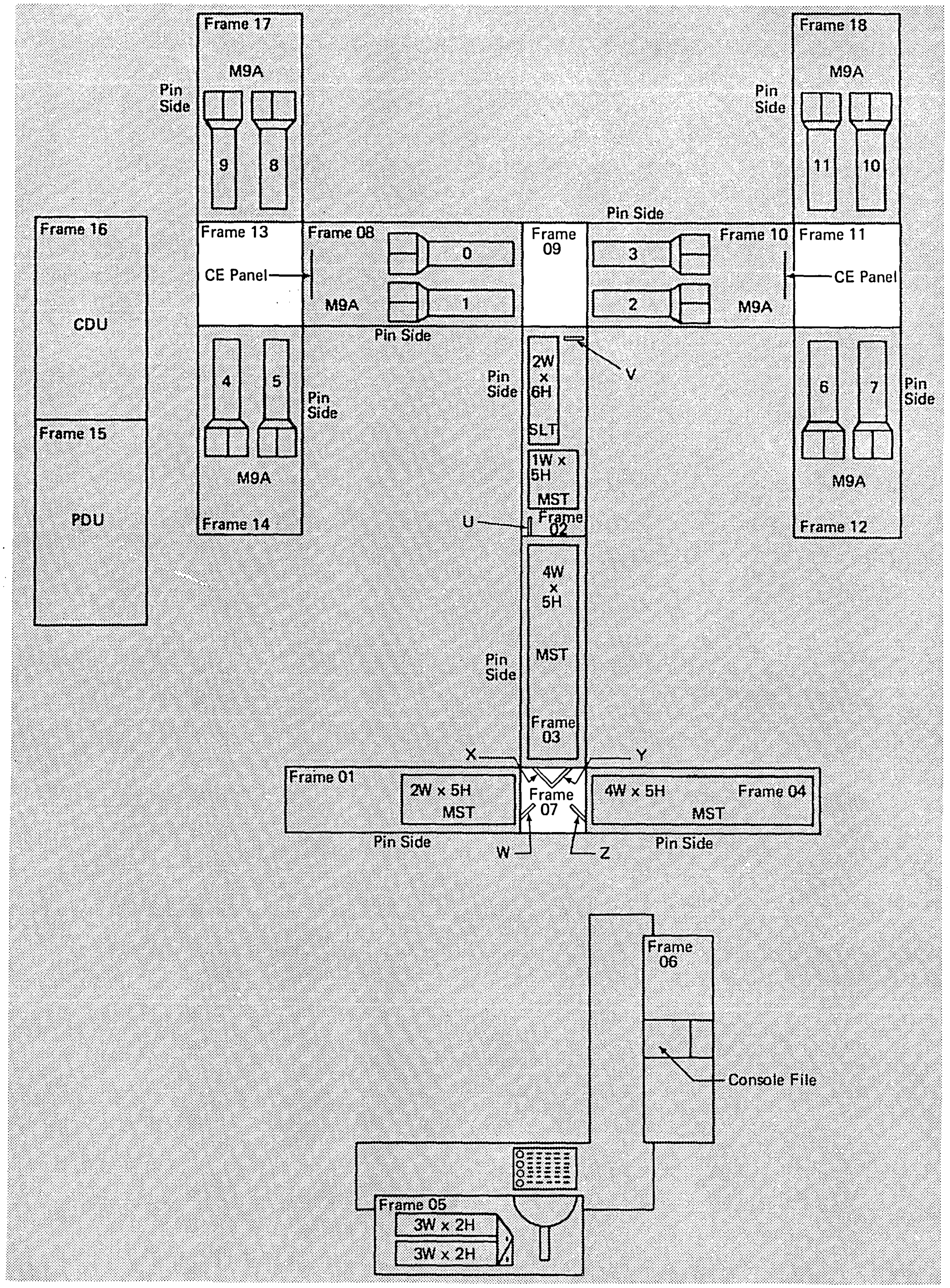
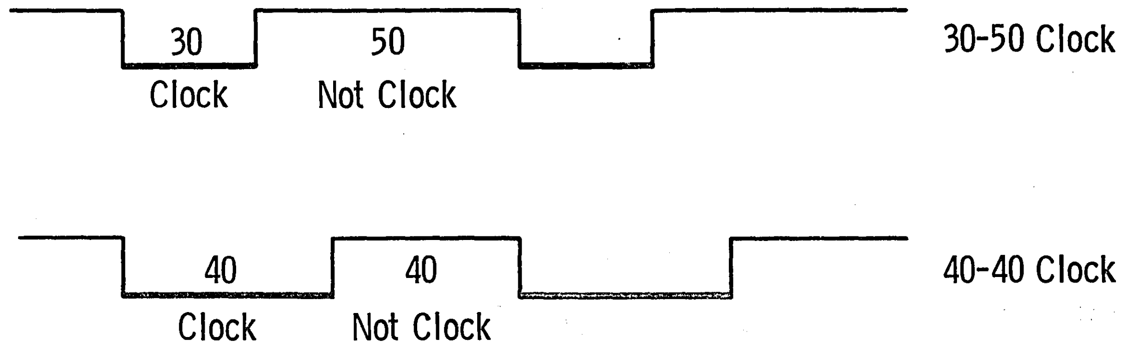


Figure 2-21 - Cable Feed Through Frames



* The term 40-40 refers to the clock and not-clock portions of the 80-nsec clock cycle. In this case, both the clock and not-clock portions of a clock period are equal to 40 nsec. In the 30-50 clock is 30 nsec and not-clock is 50 nsec.

Figure 2-22 - Basic Clock Shapes

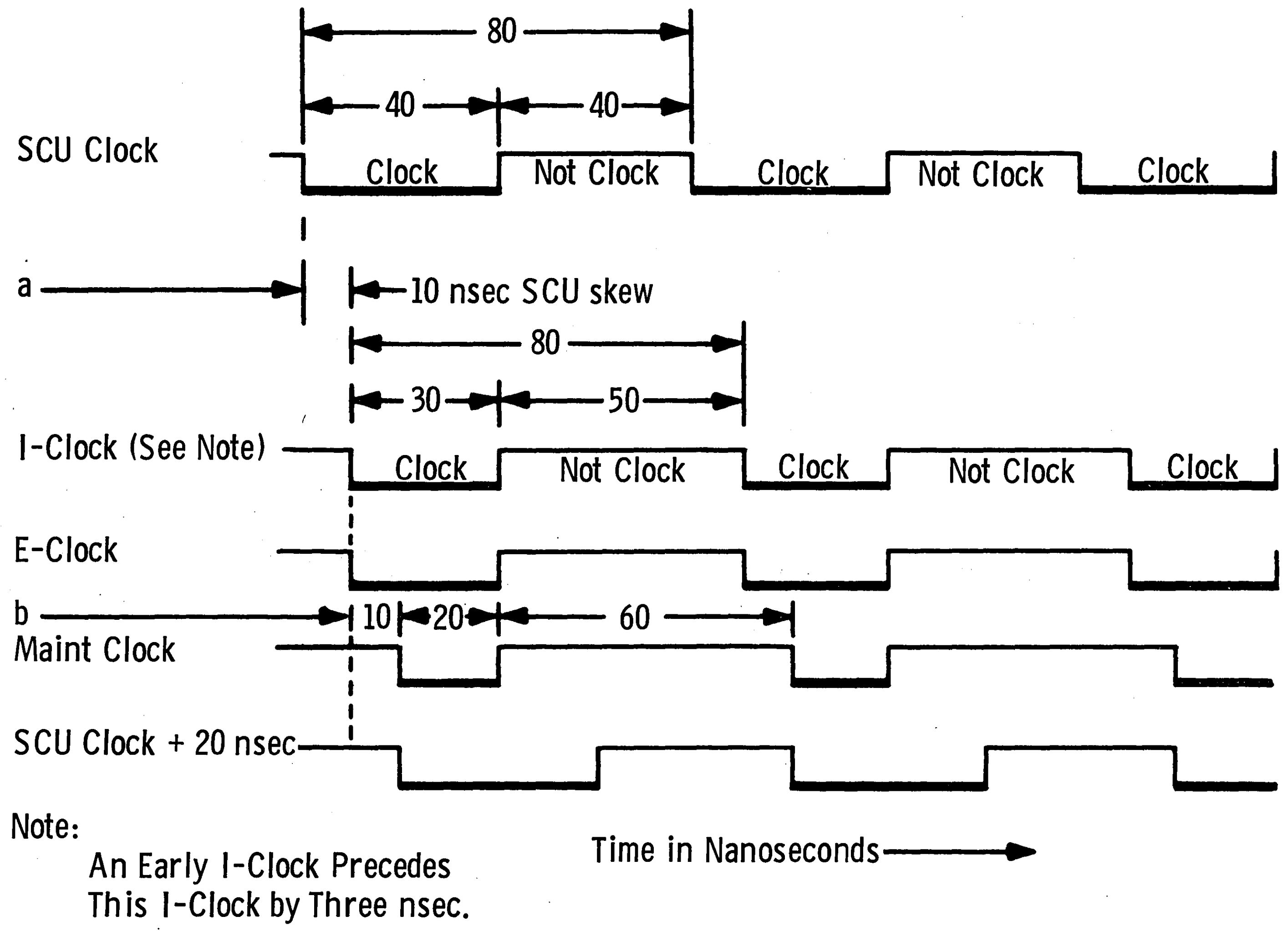


Figure 2-23 - Clock Alignment

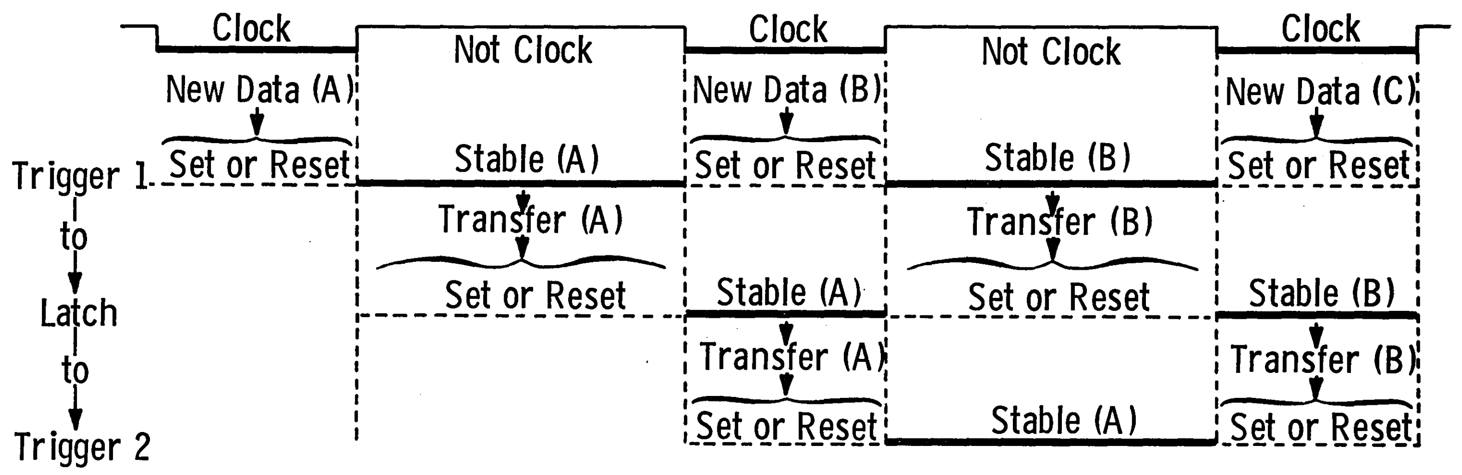


Figure 2-24 - Trigger and Latch Relationship

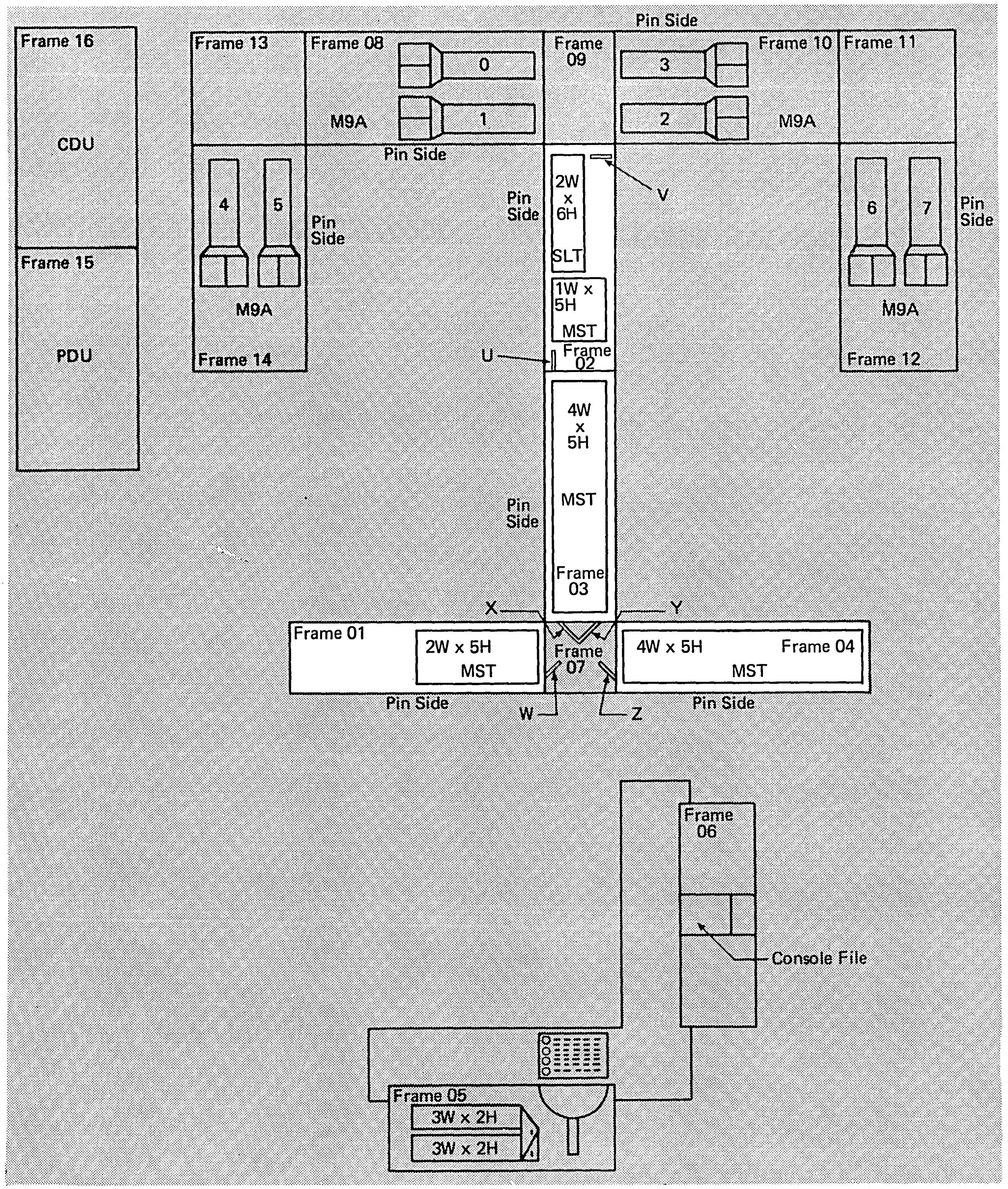


Figure 2-25 - CPU

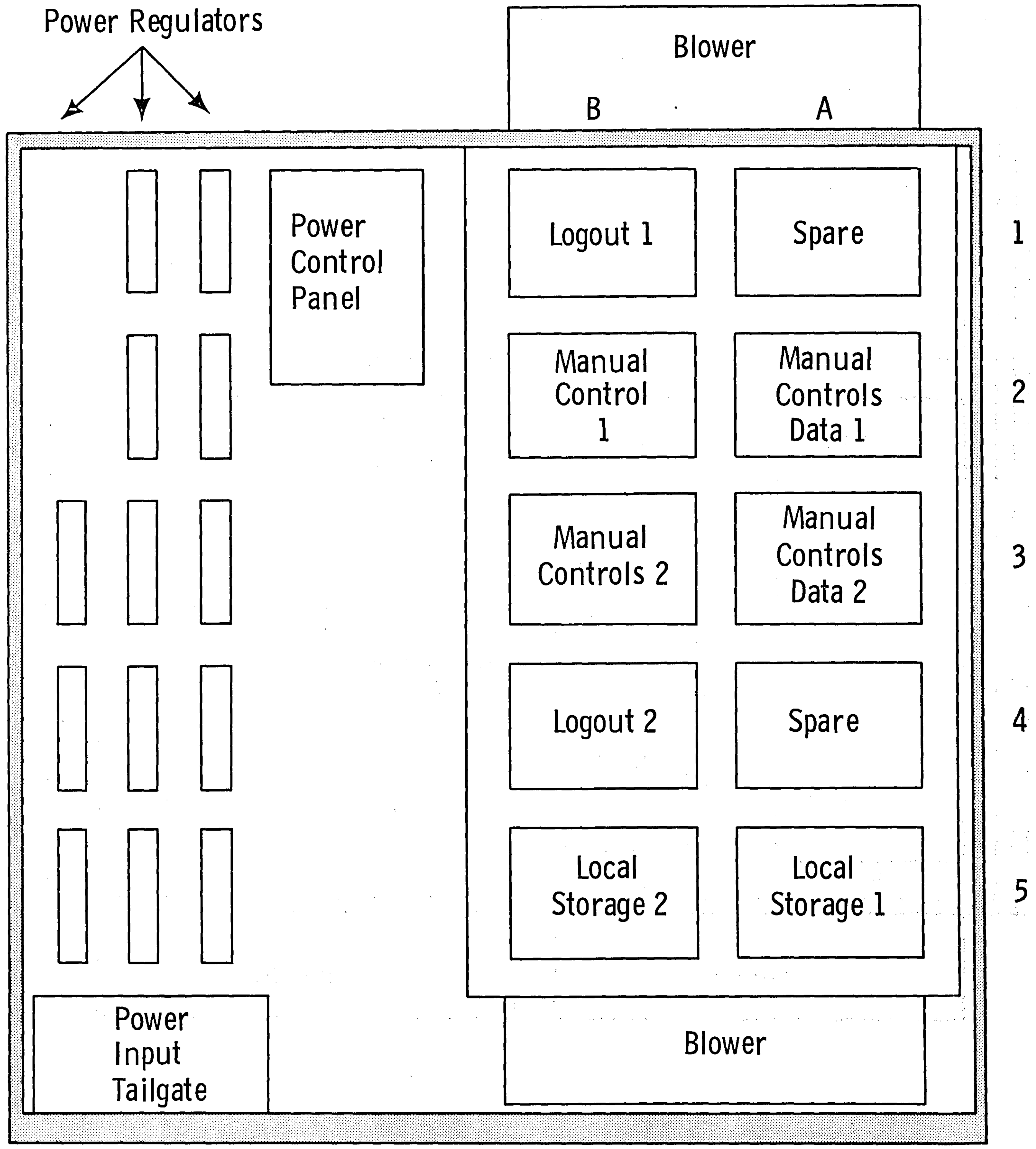


Figure 2-26 - Frame 01 Pin Side

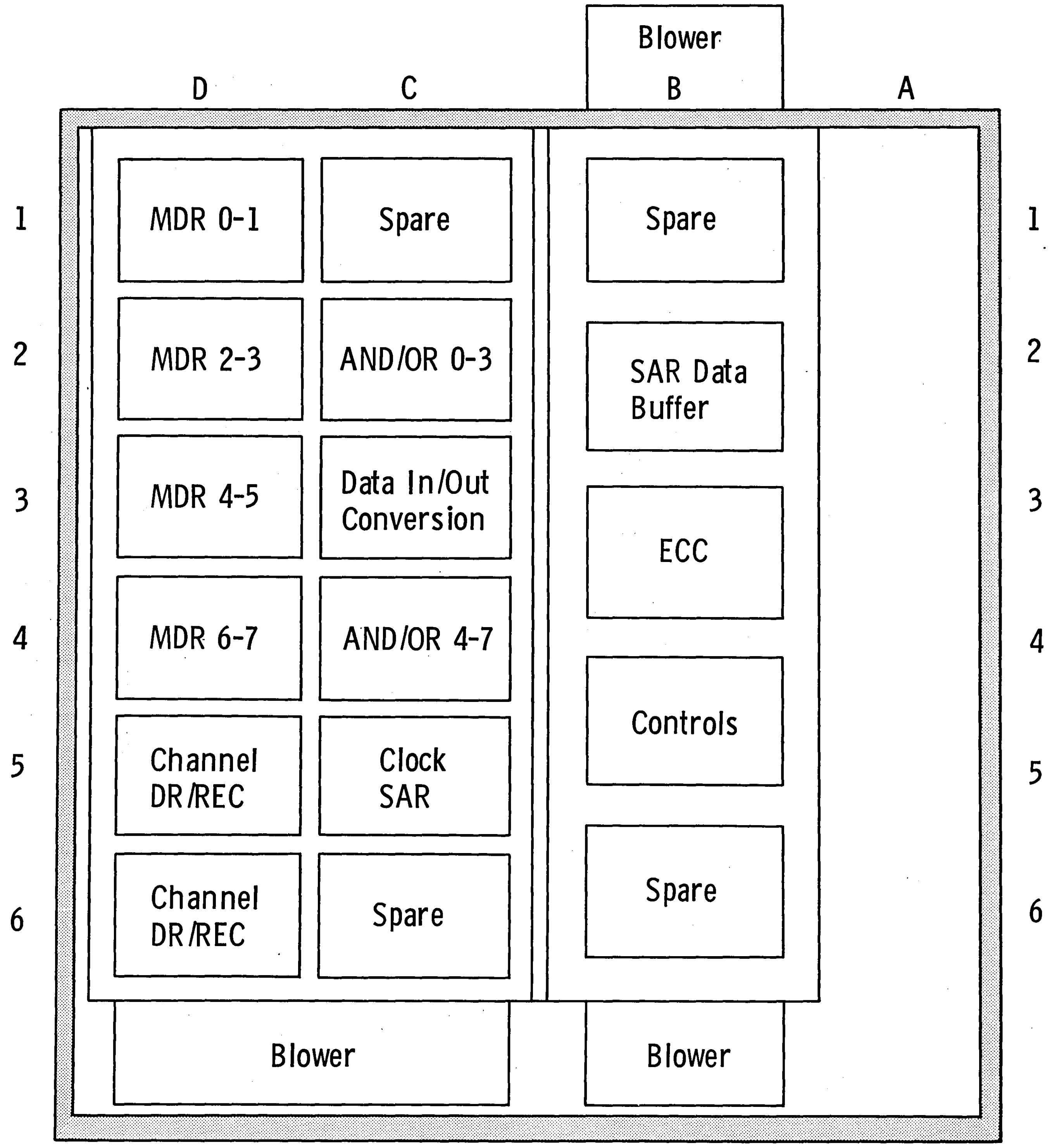


Figure 2-27 - Frame 02 Pin Side

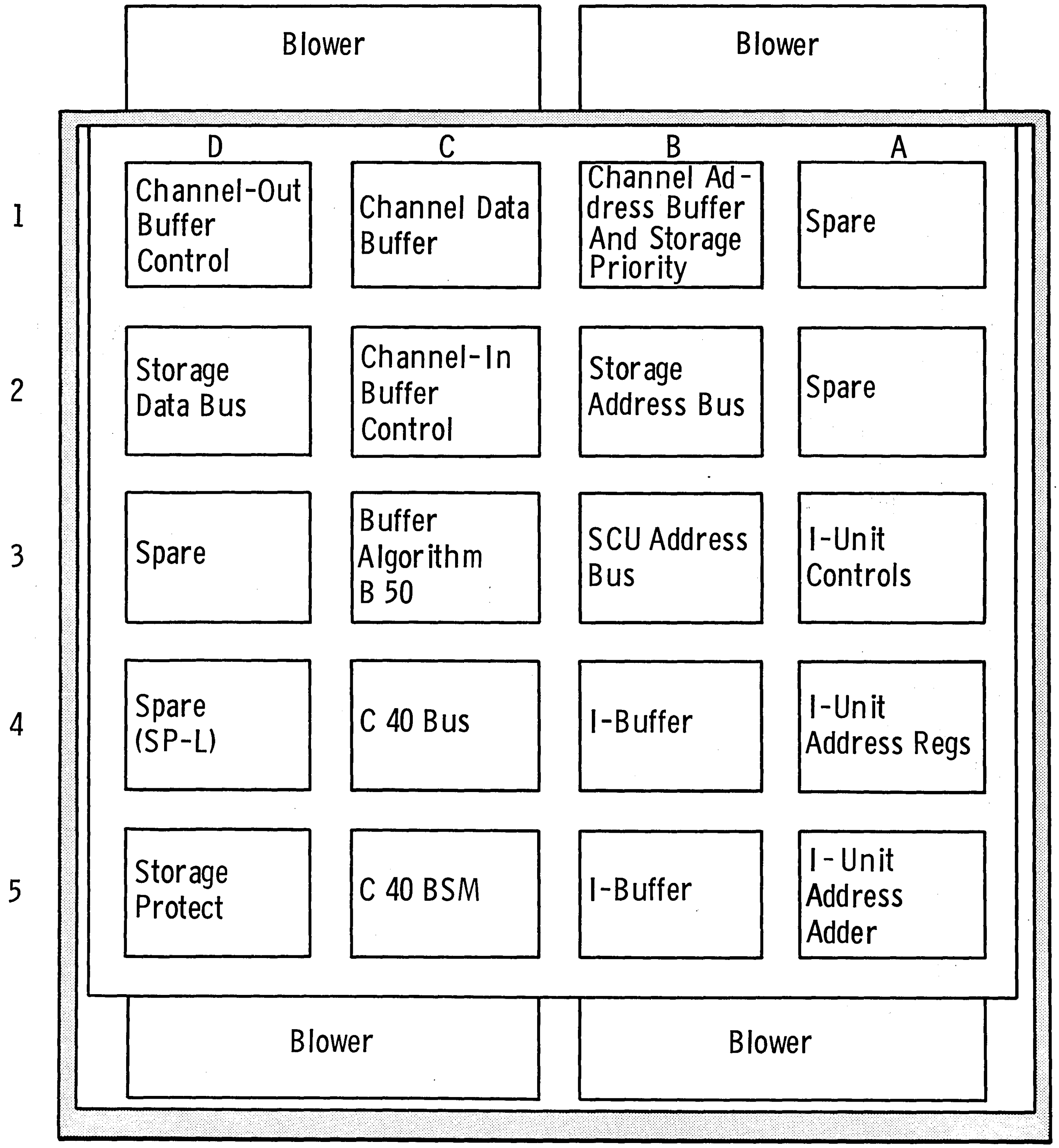


Figure 2-28 - Frame 03 Pin Side

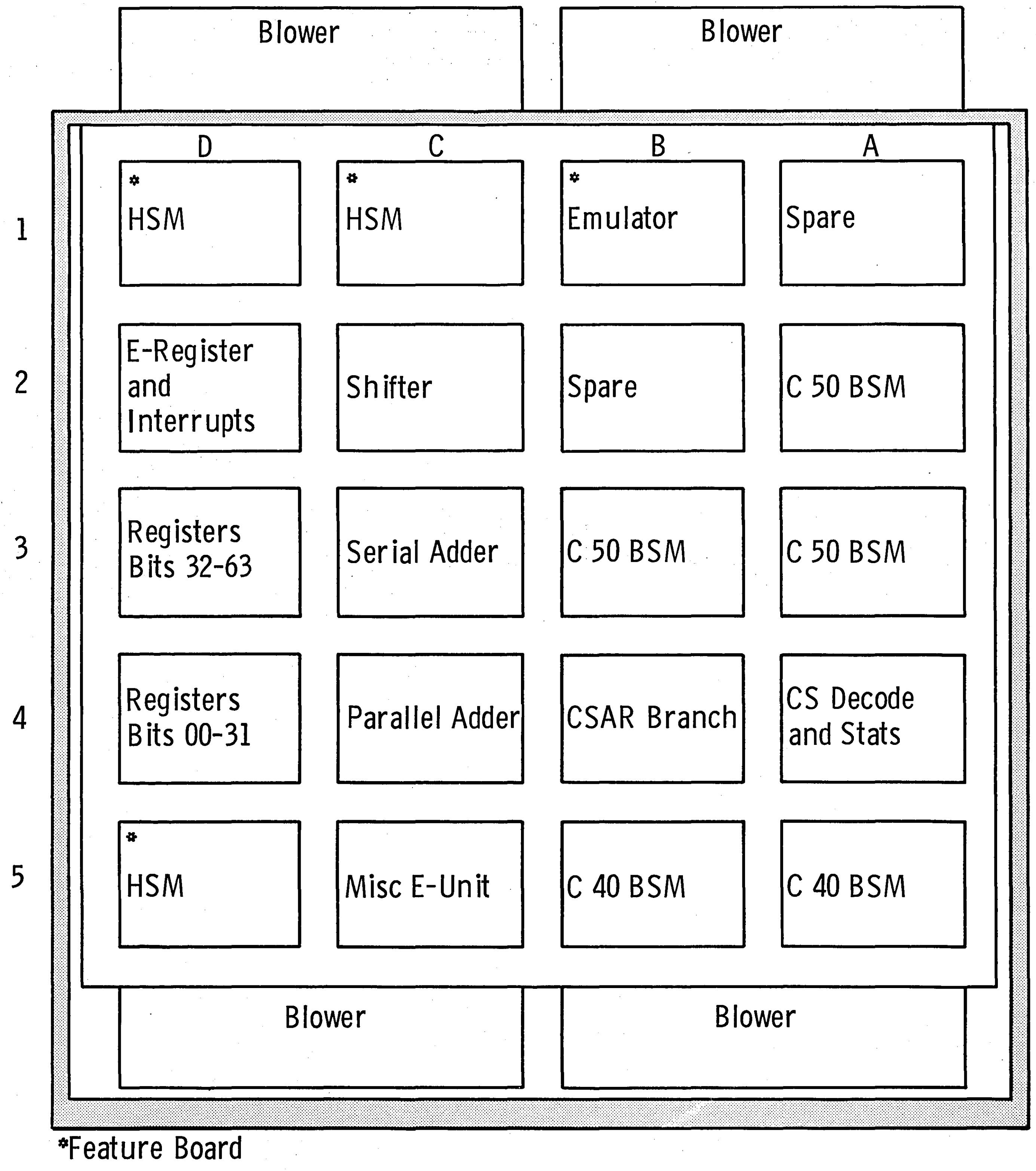


Figure 2-29 - Frame 04 Pin Side

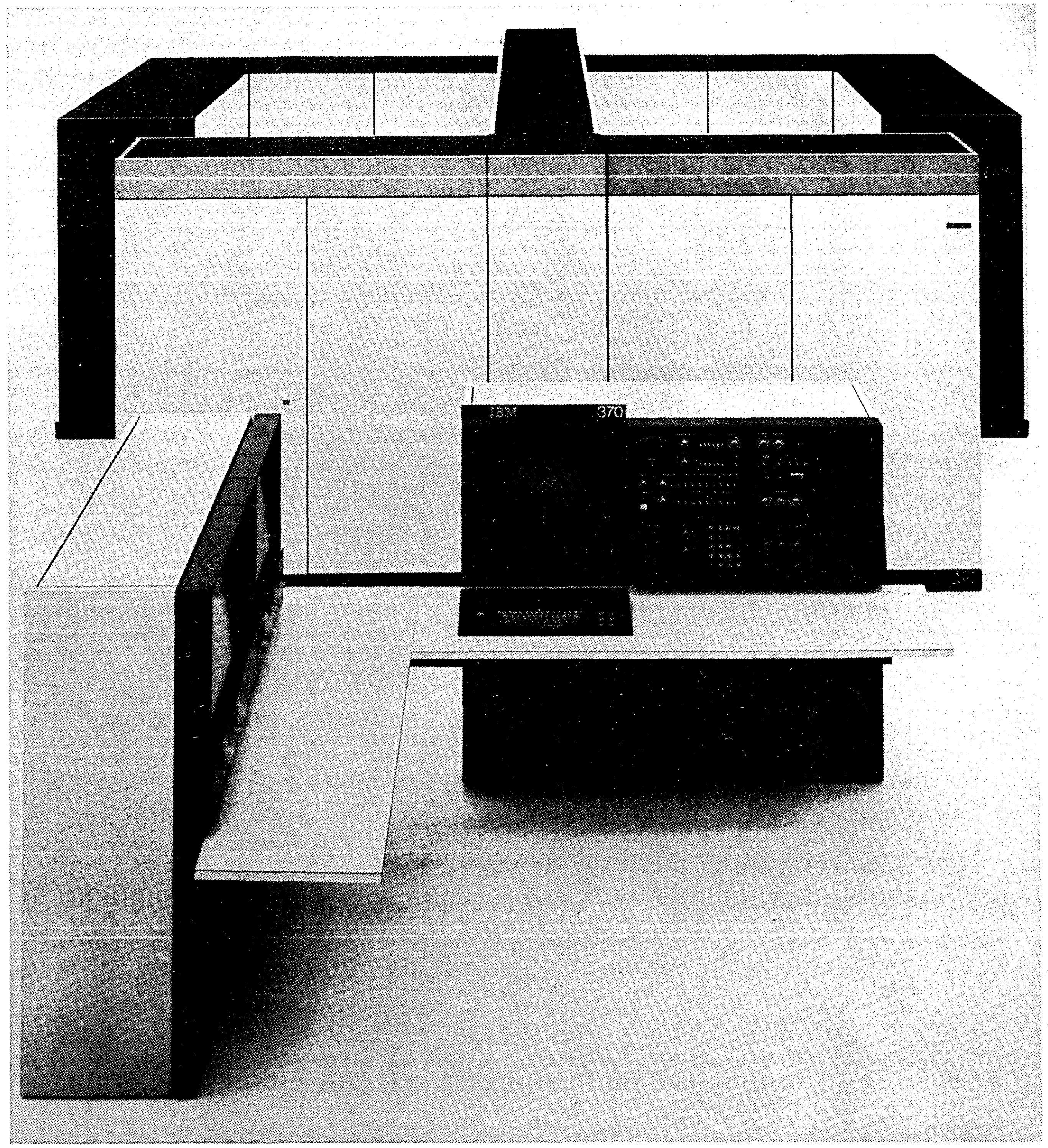


Figure 2-30 - Model 165

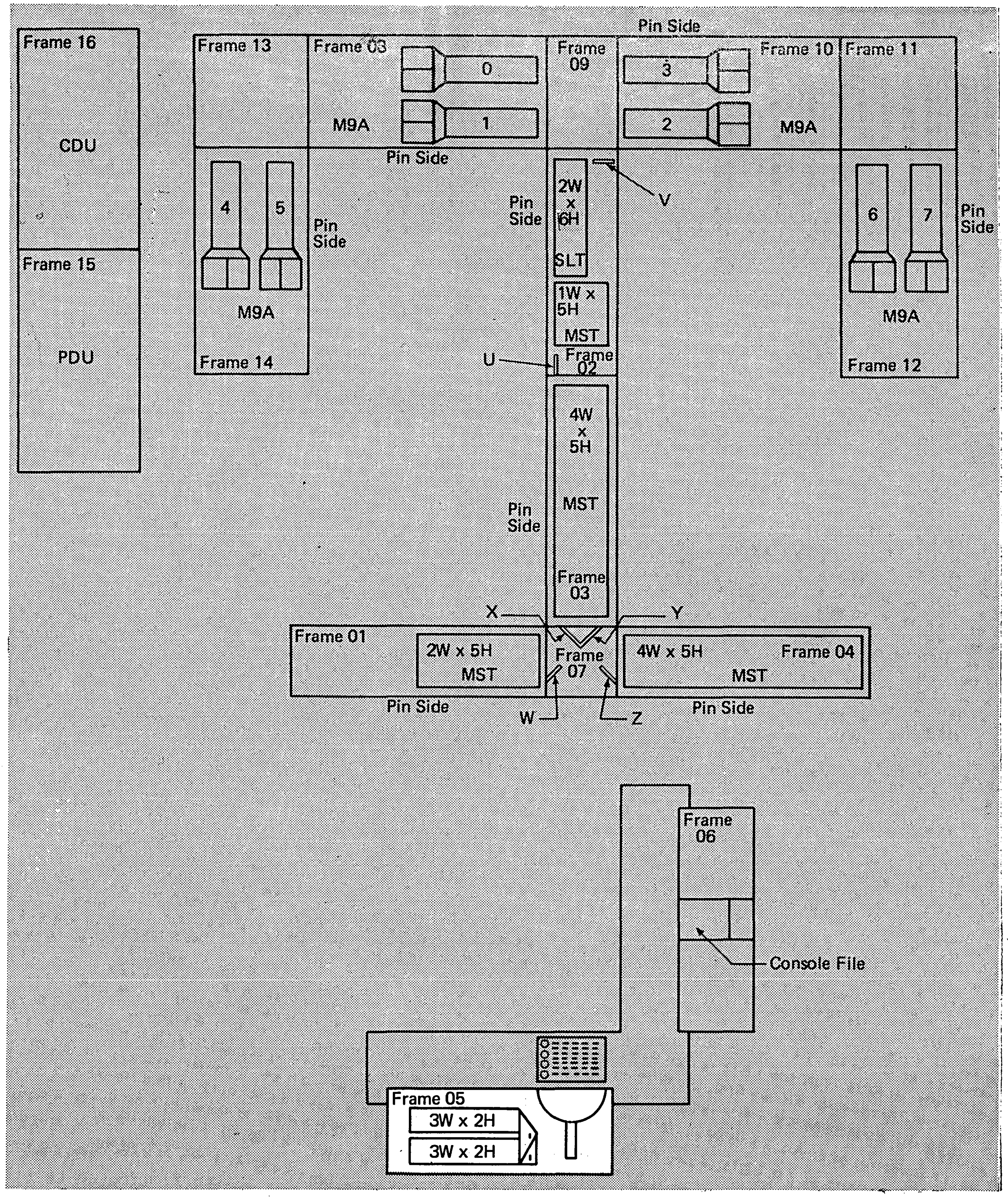


Figure 2-31 - System Control Panel

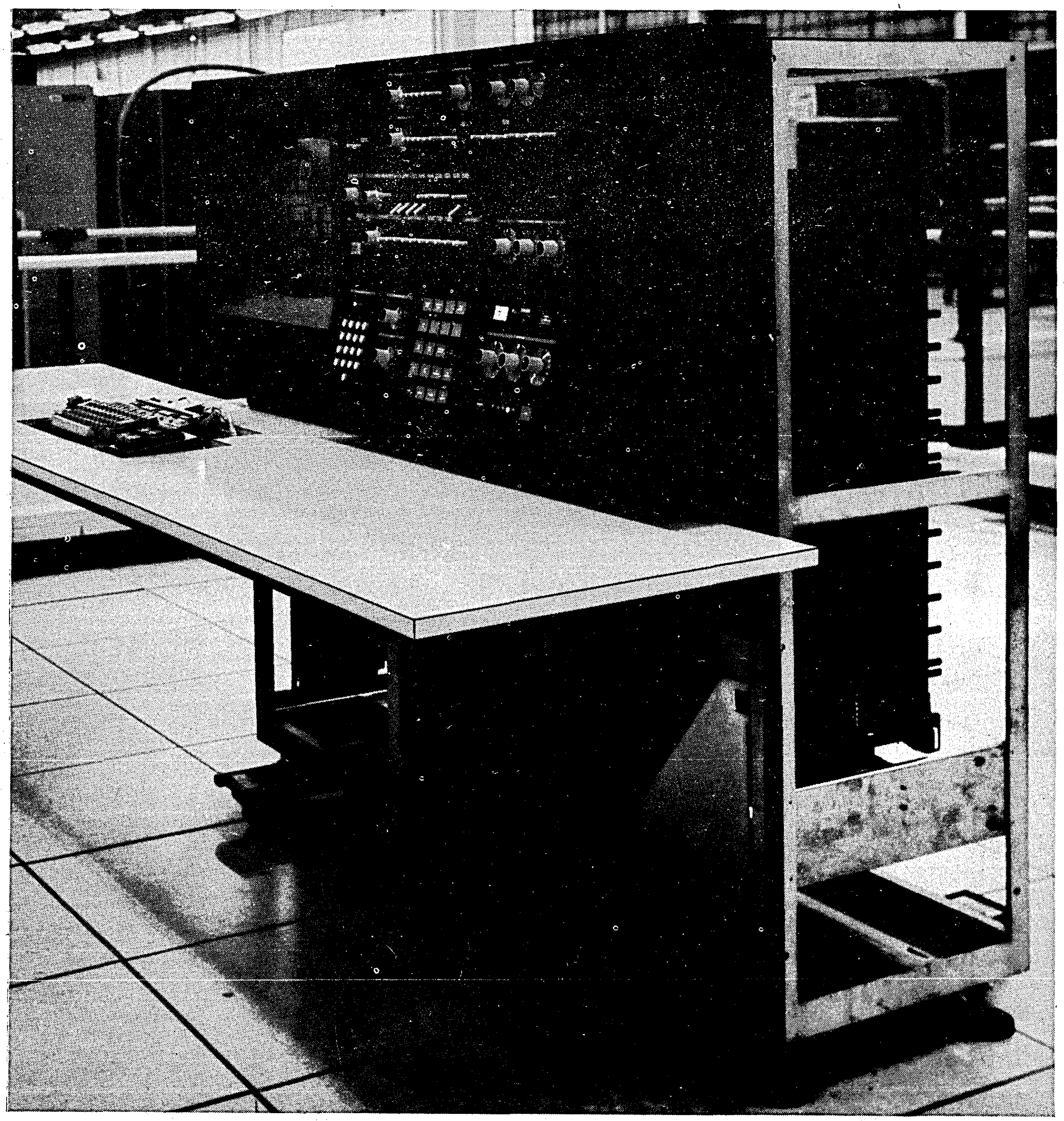


Figure 2-32 - System Control Panel

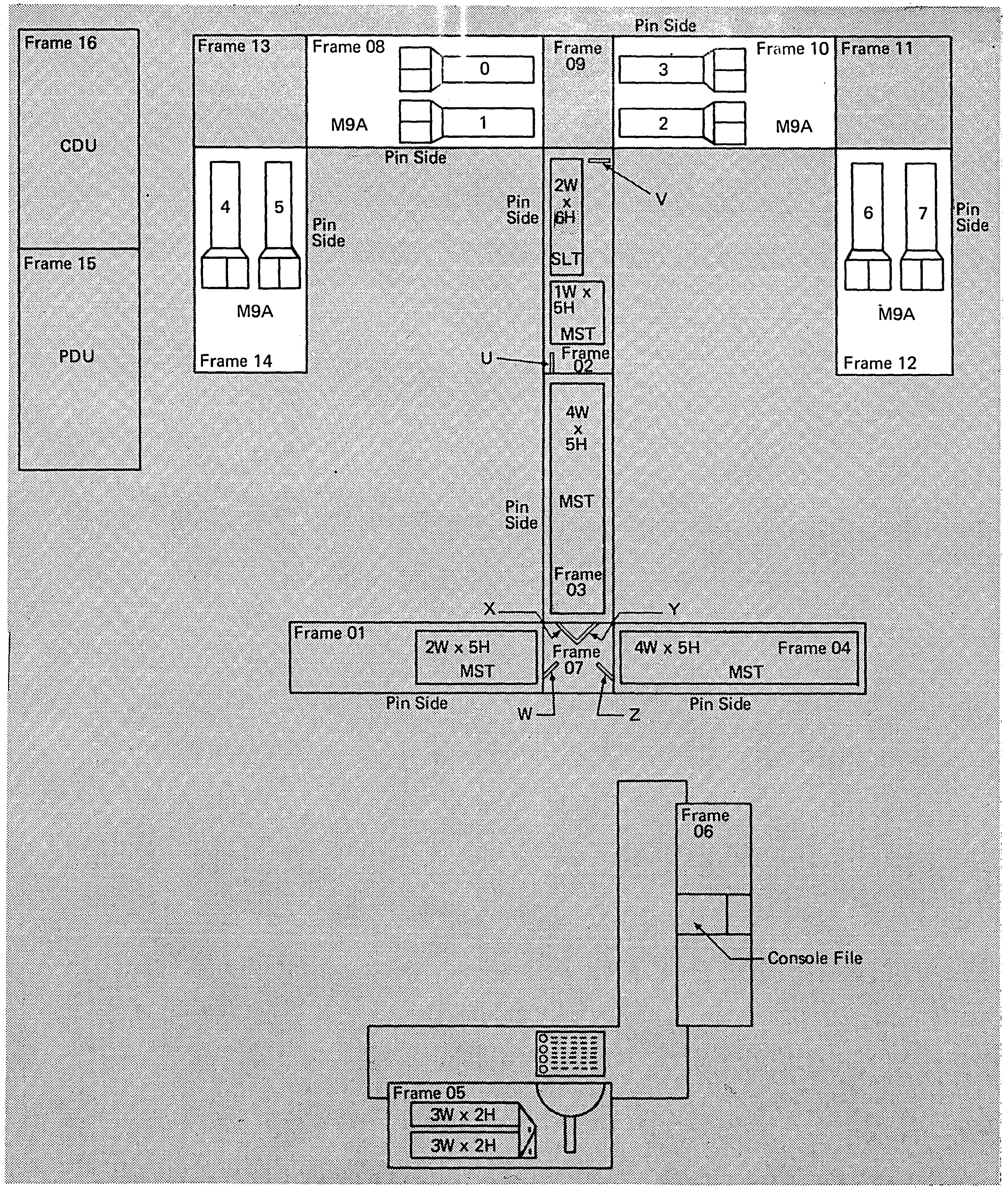


Figure 2-33 - Main Storage

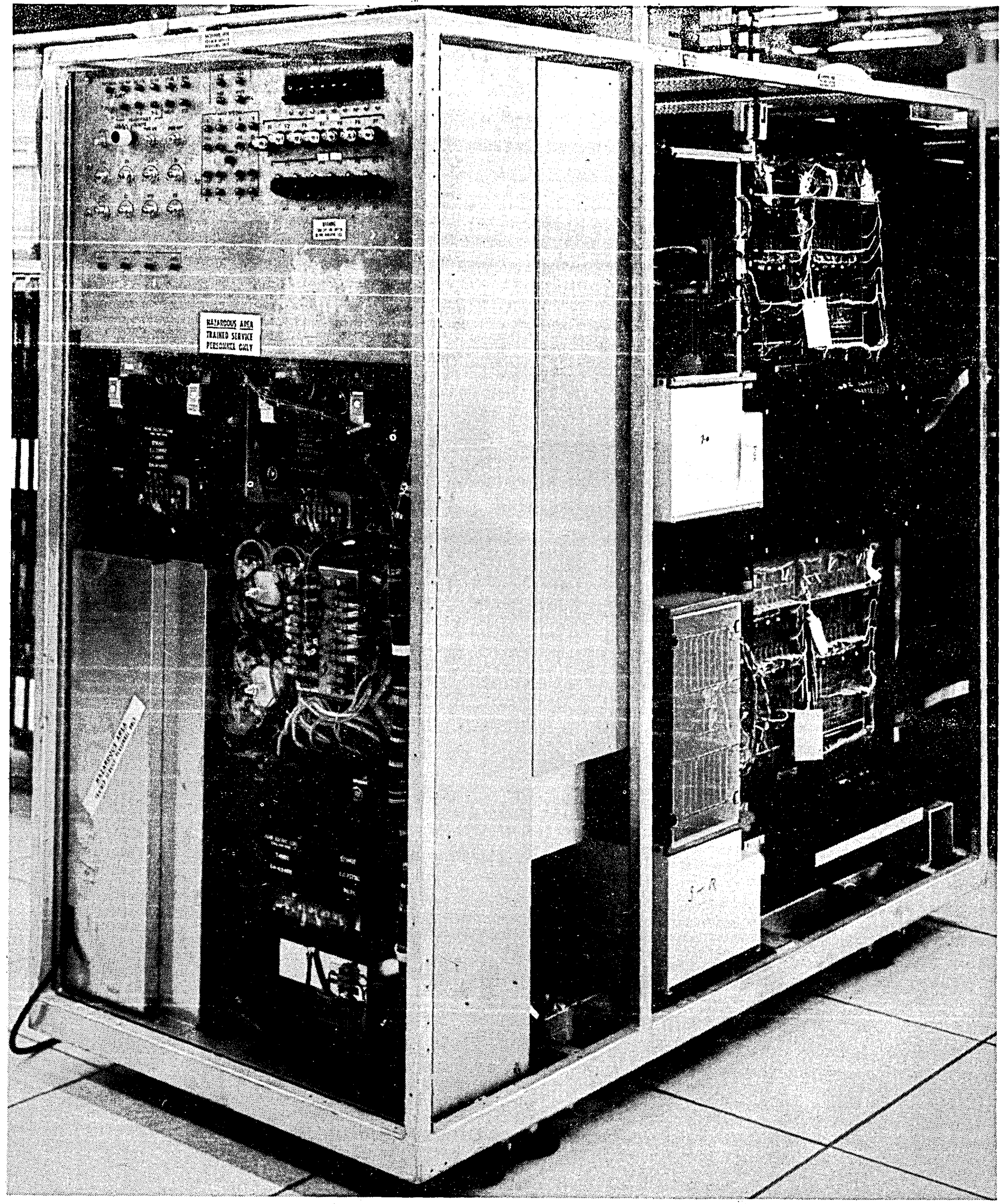


Figure 2-34 - M9A

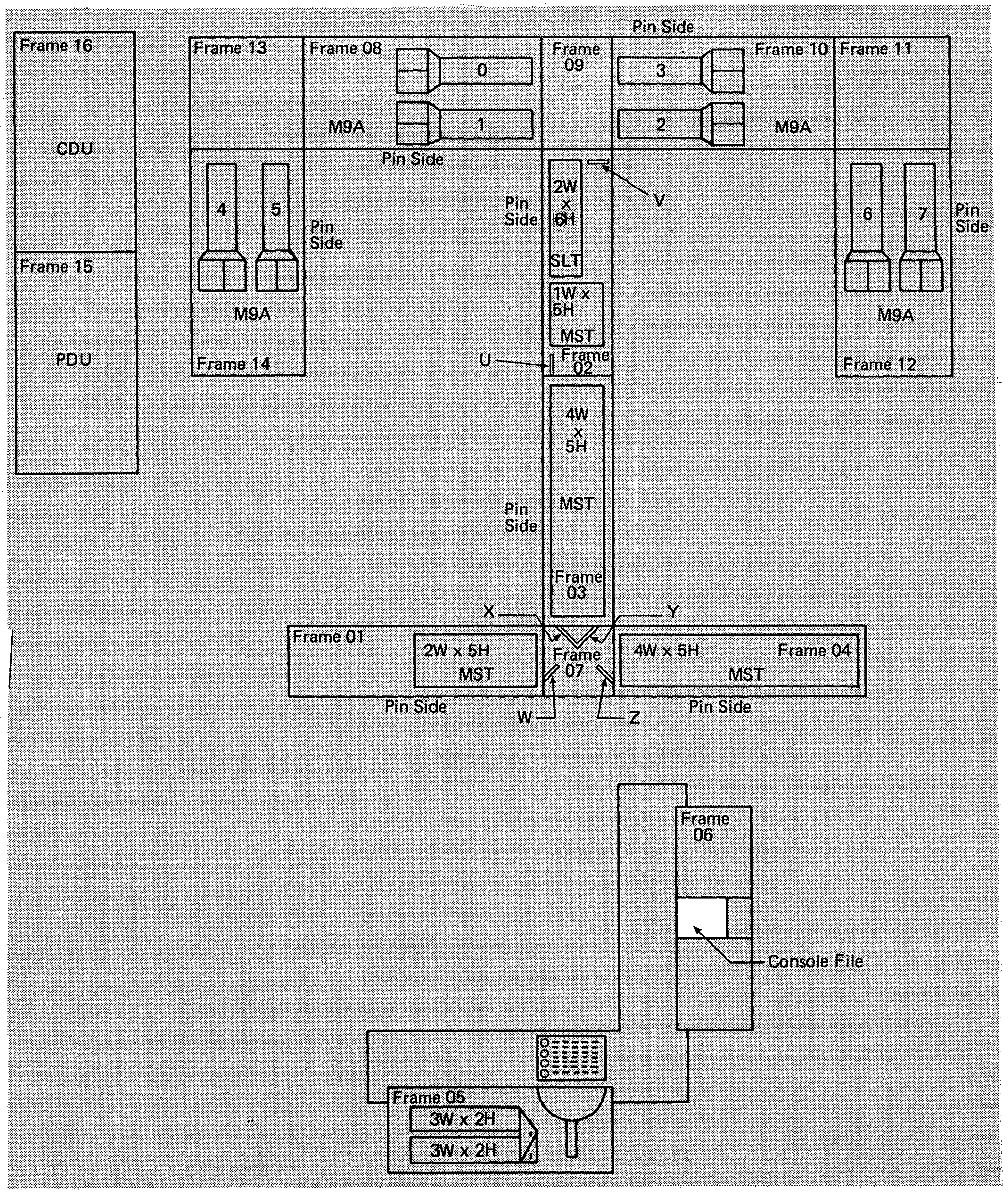


Figure 2-35 - Console File

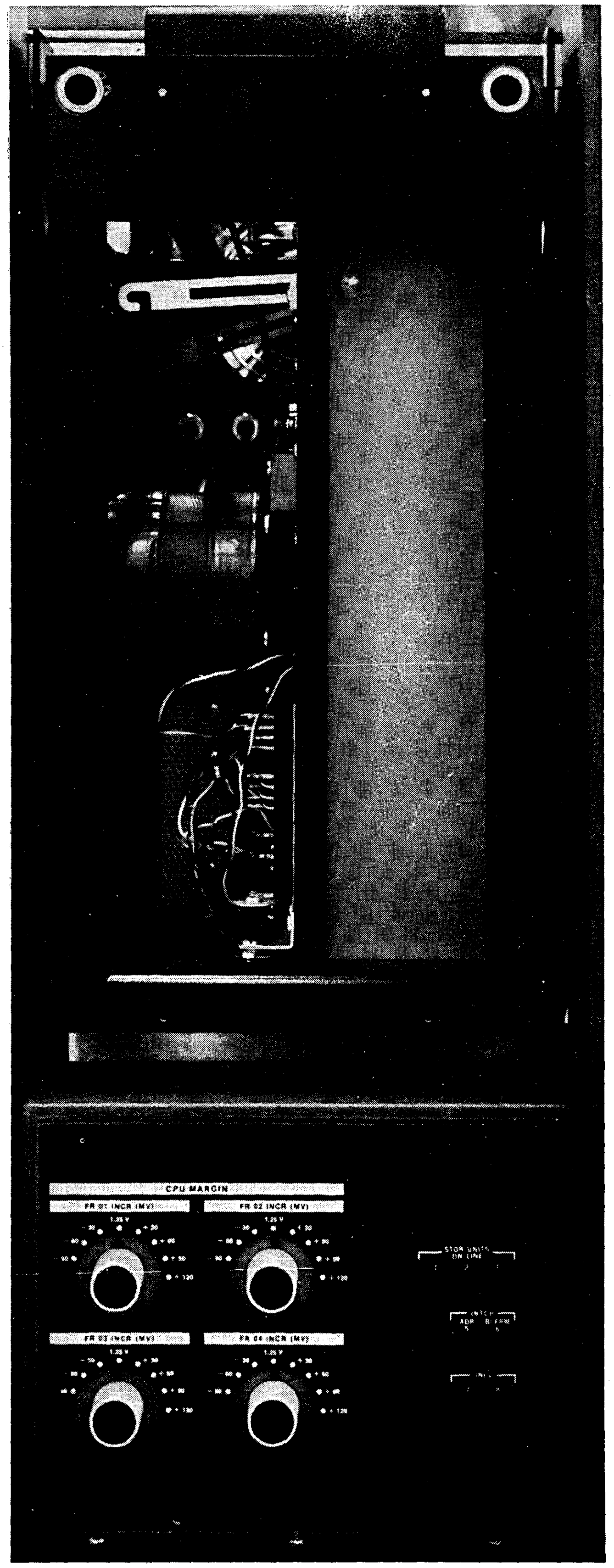


Figure 2-36 - Console File

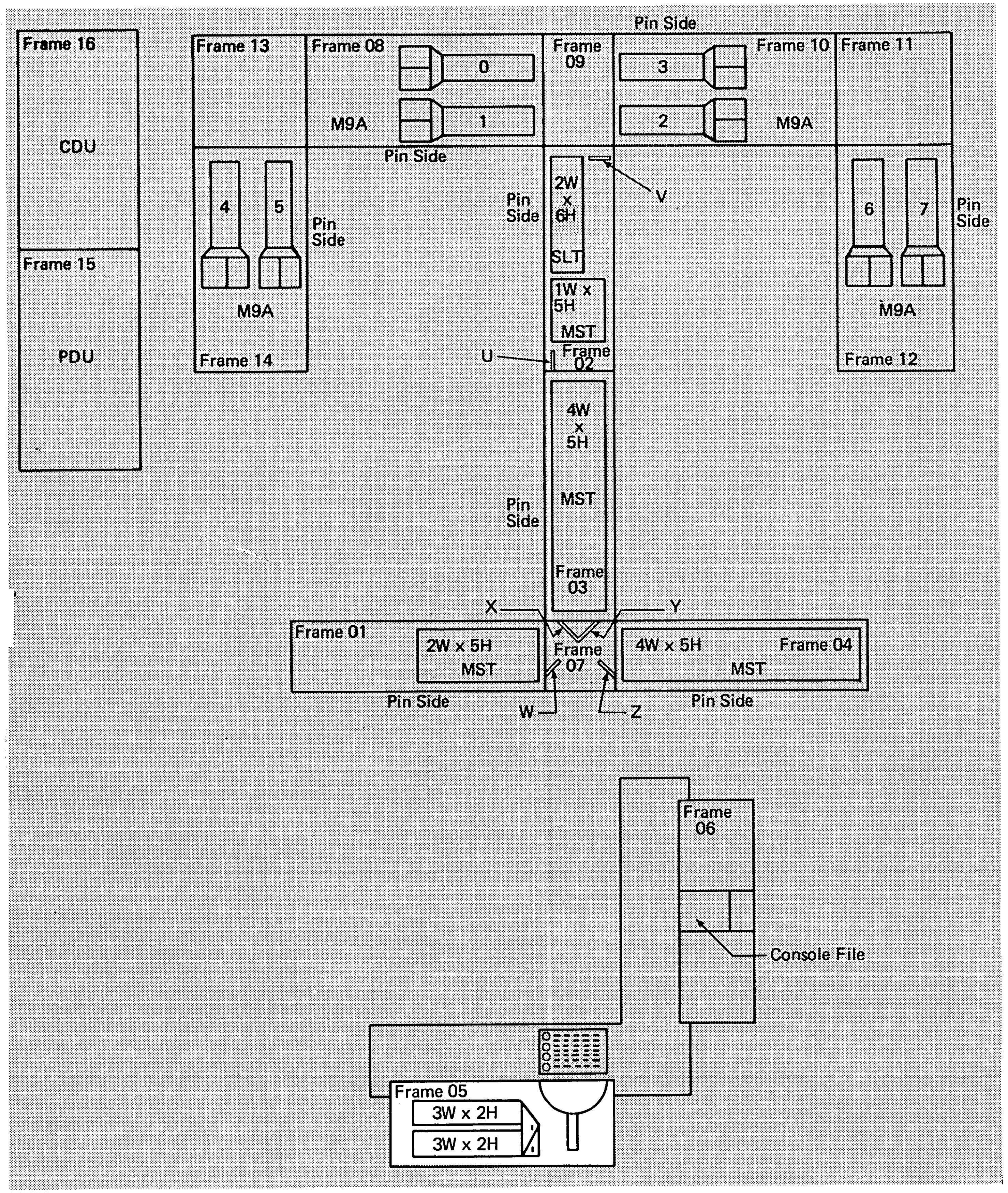


Figure 2-37 - Frame 5

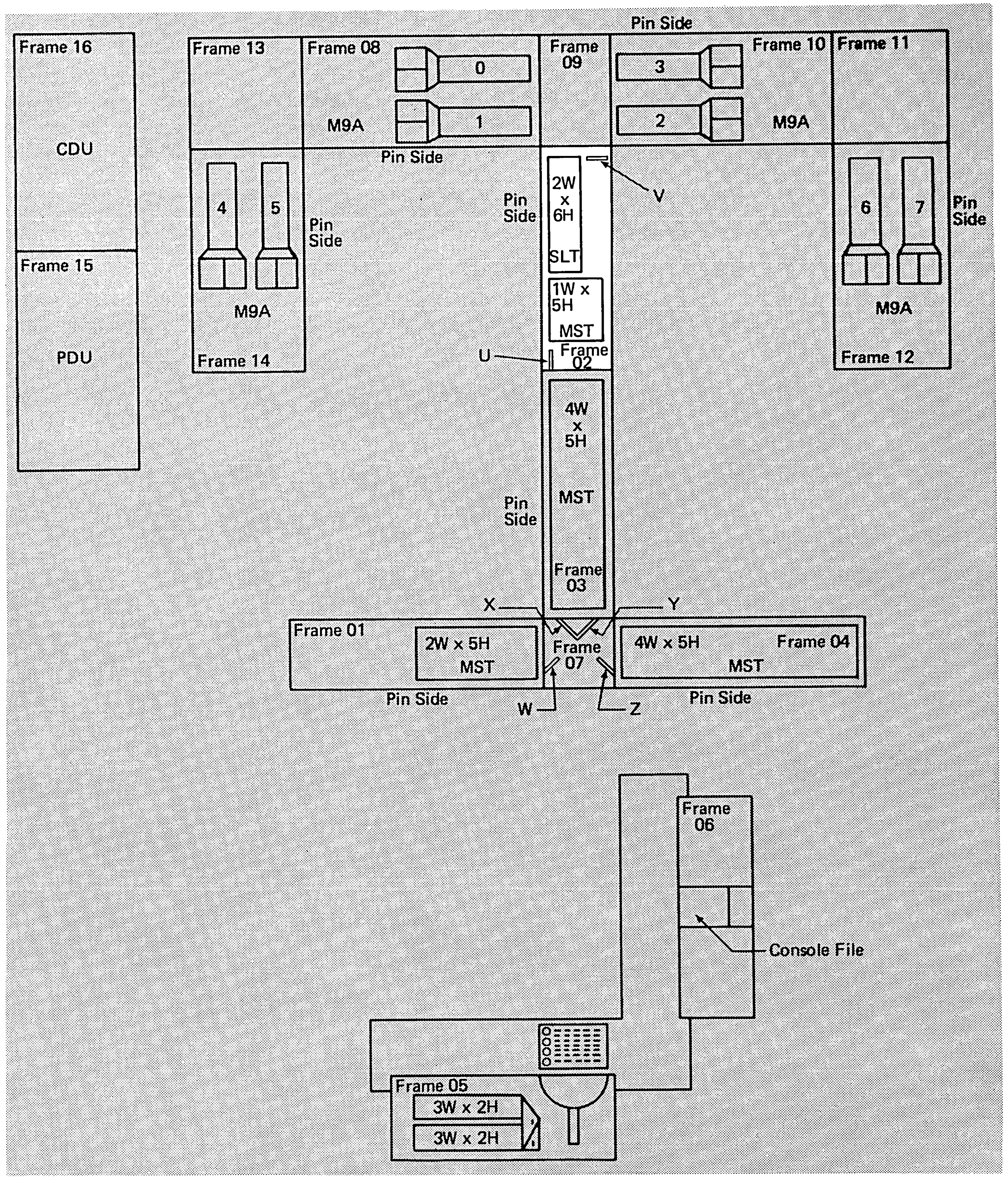


Figure 2-38 - Frame 02

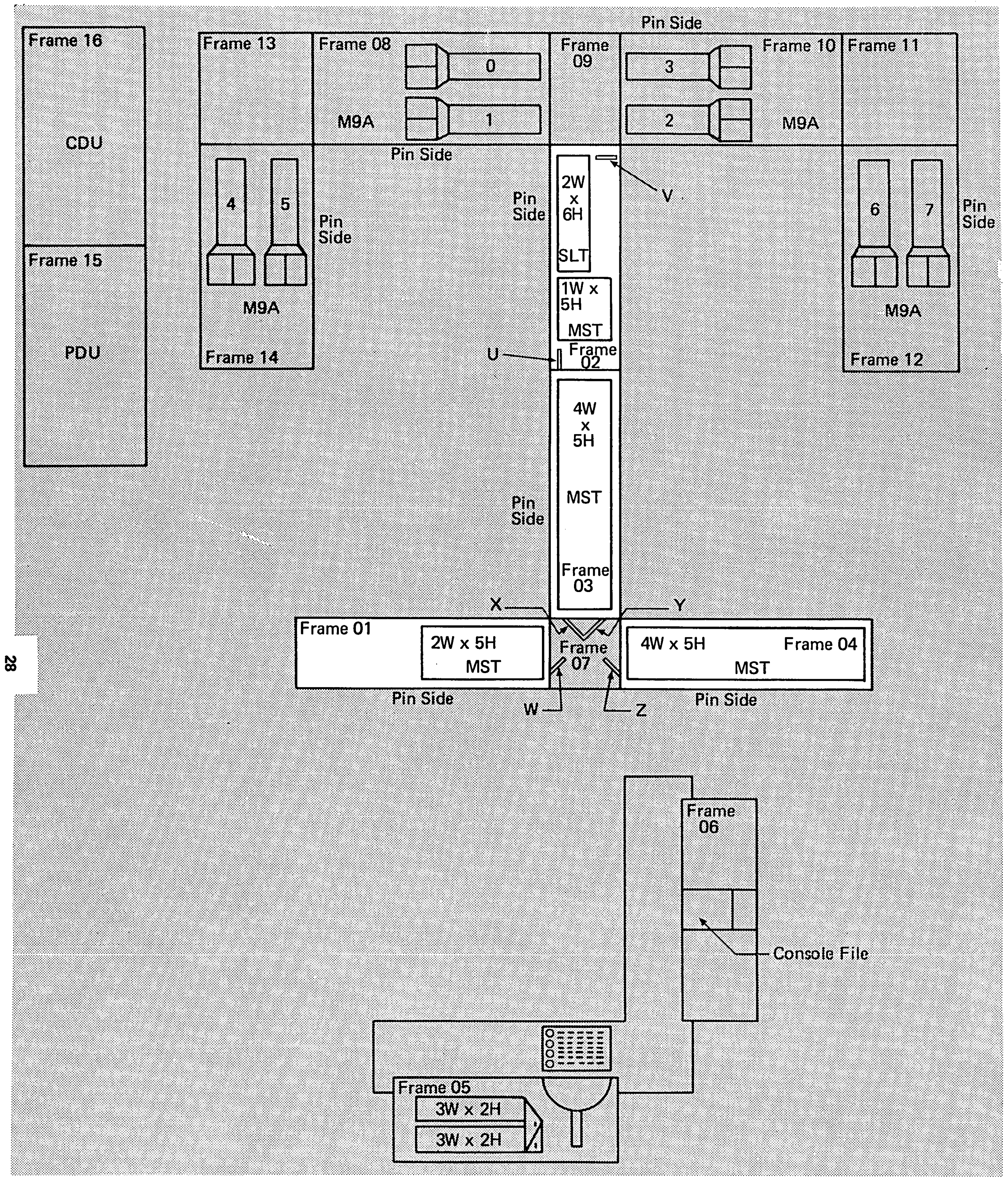


Figure 2-39 - CPU Frames

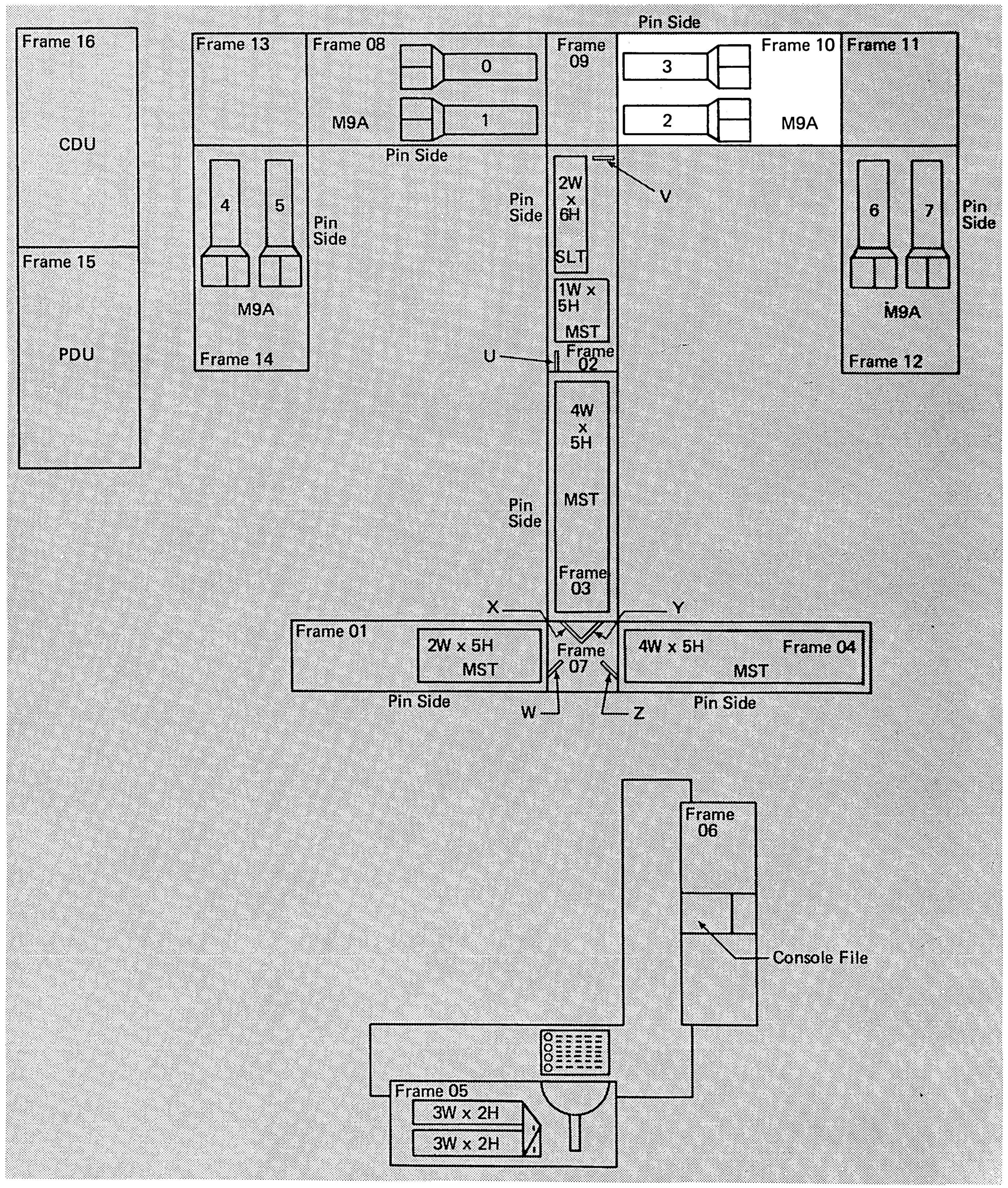


Figure 2-40 - Frame 10

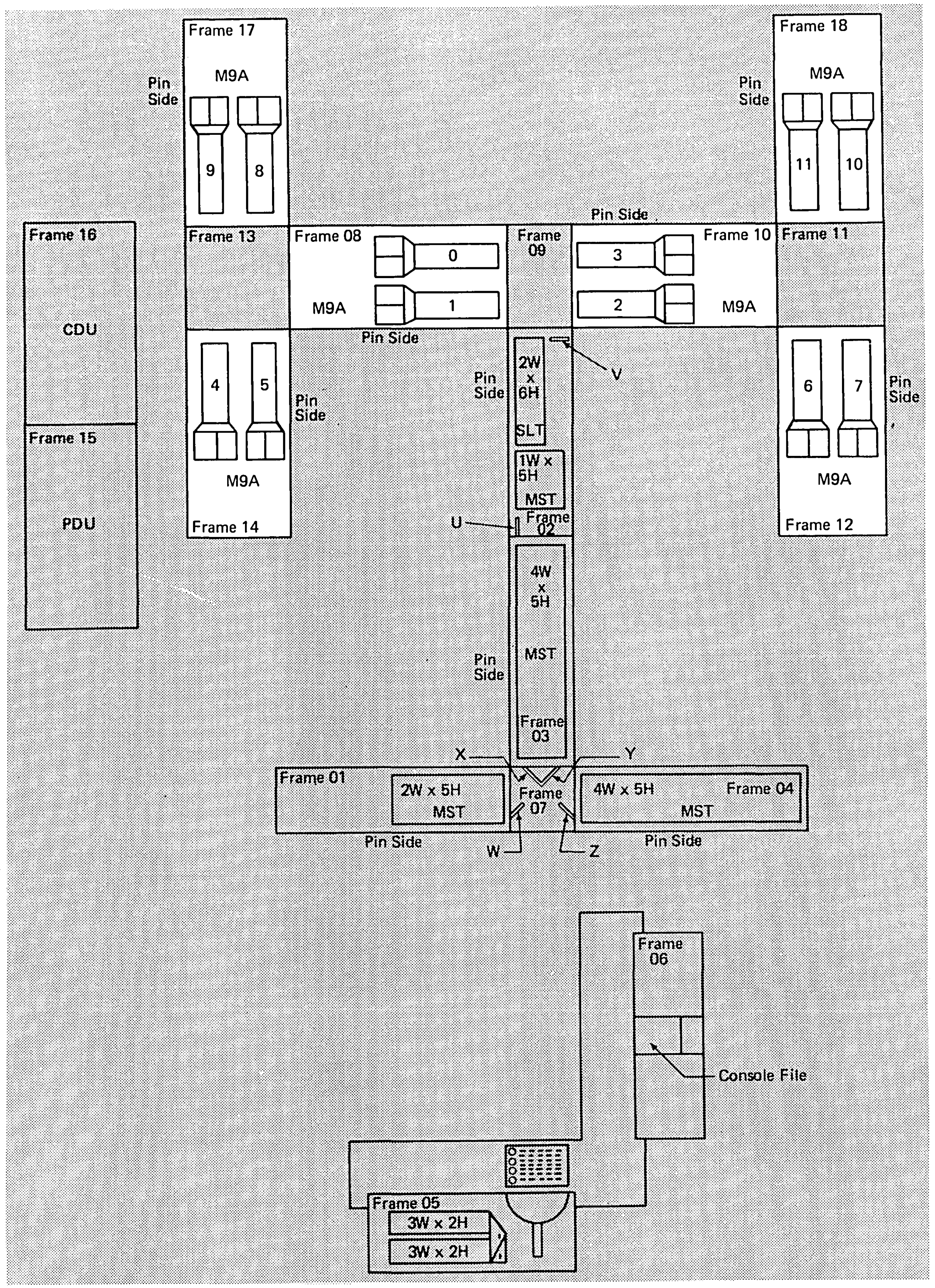


Figure 2-41 - Main Storage Frames

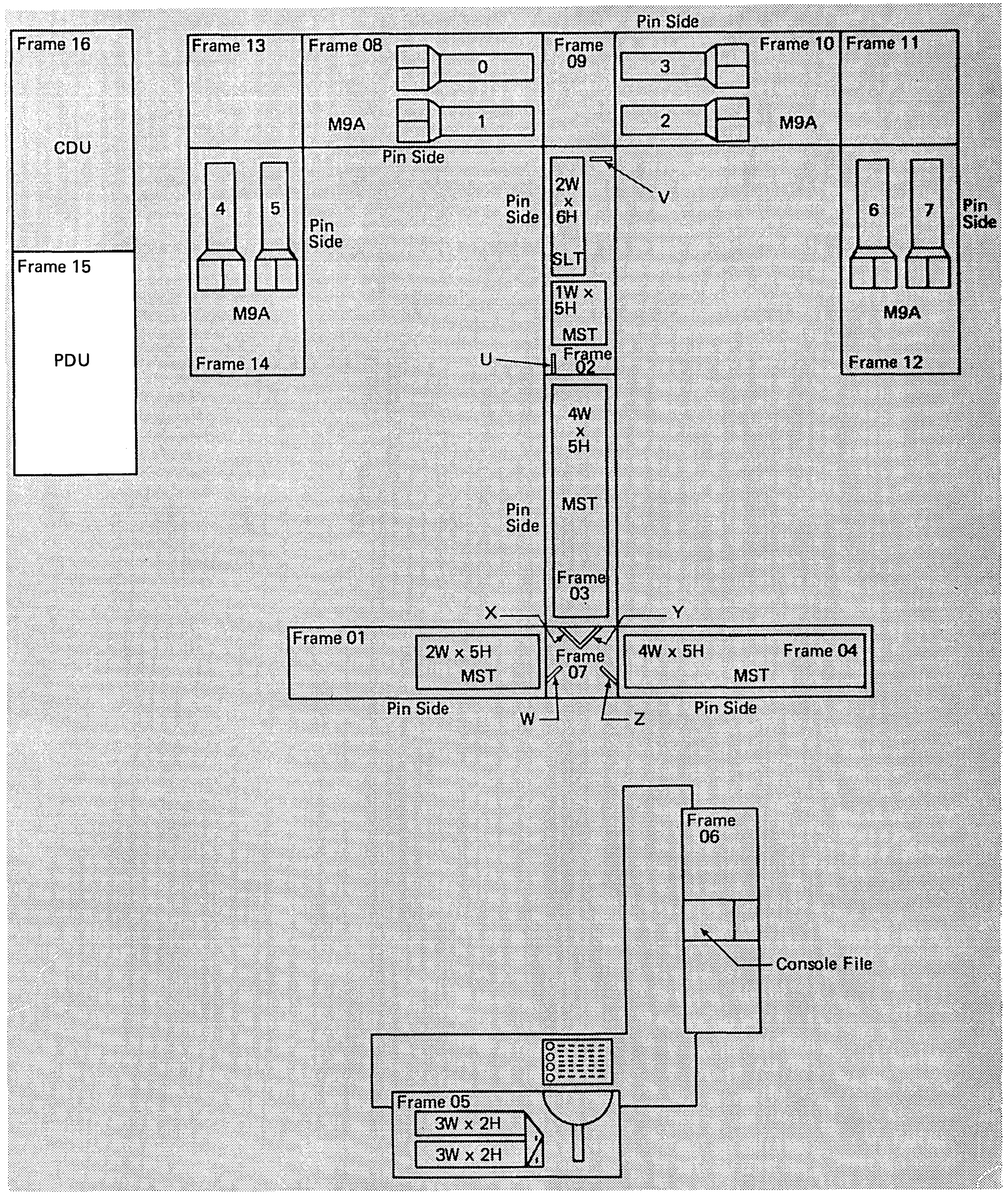


Figure 2-42 - Power Distribution Unit

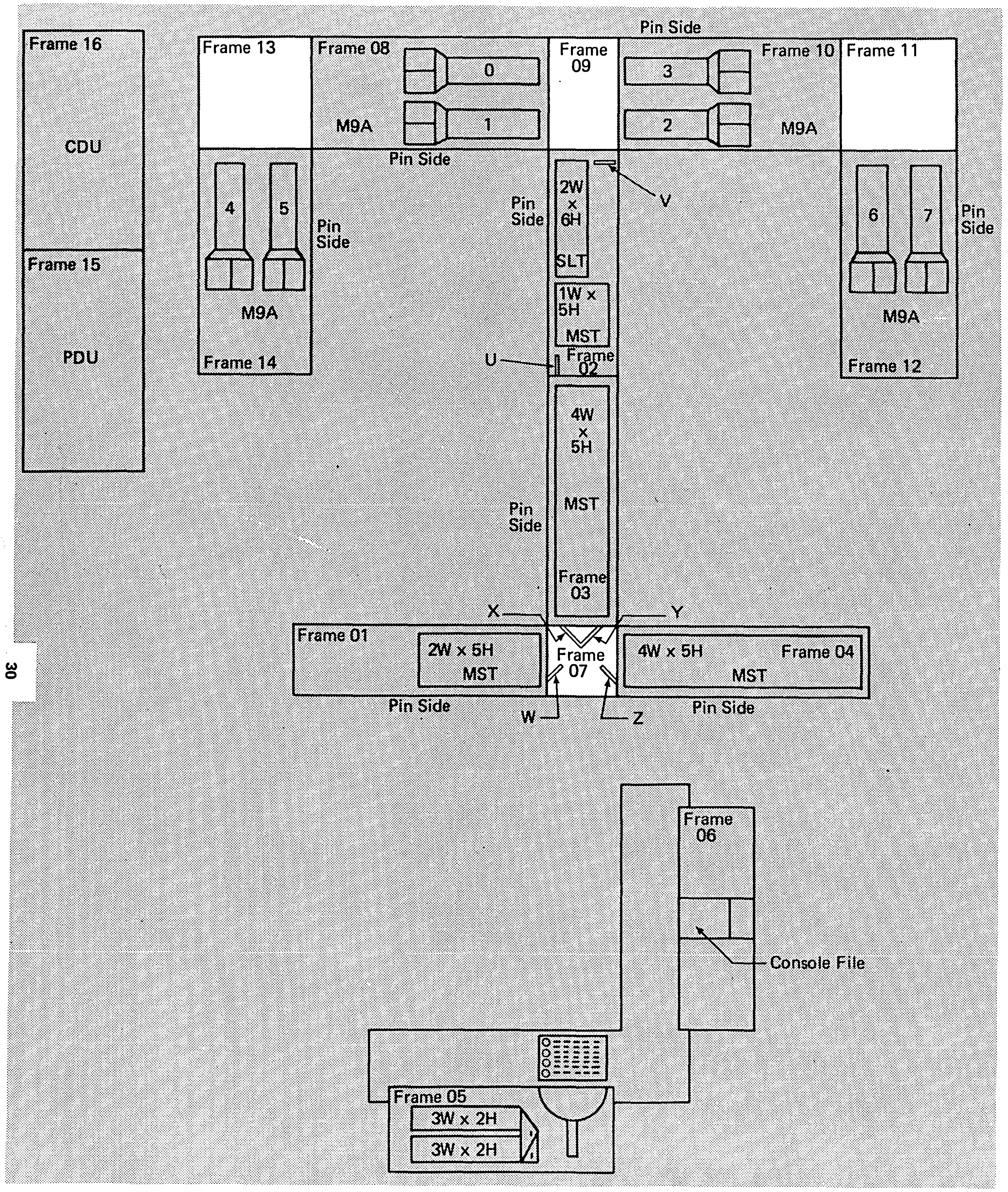


Figure 2-43 - Dummy Frames

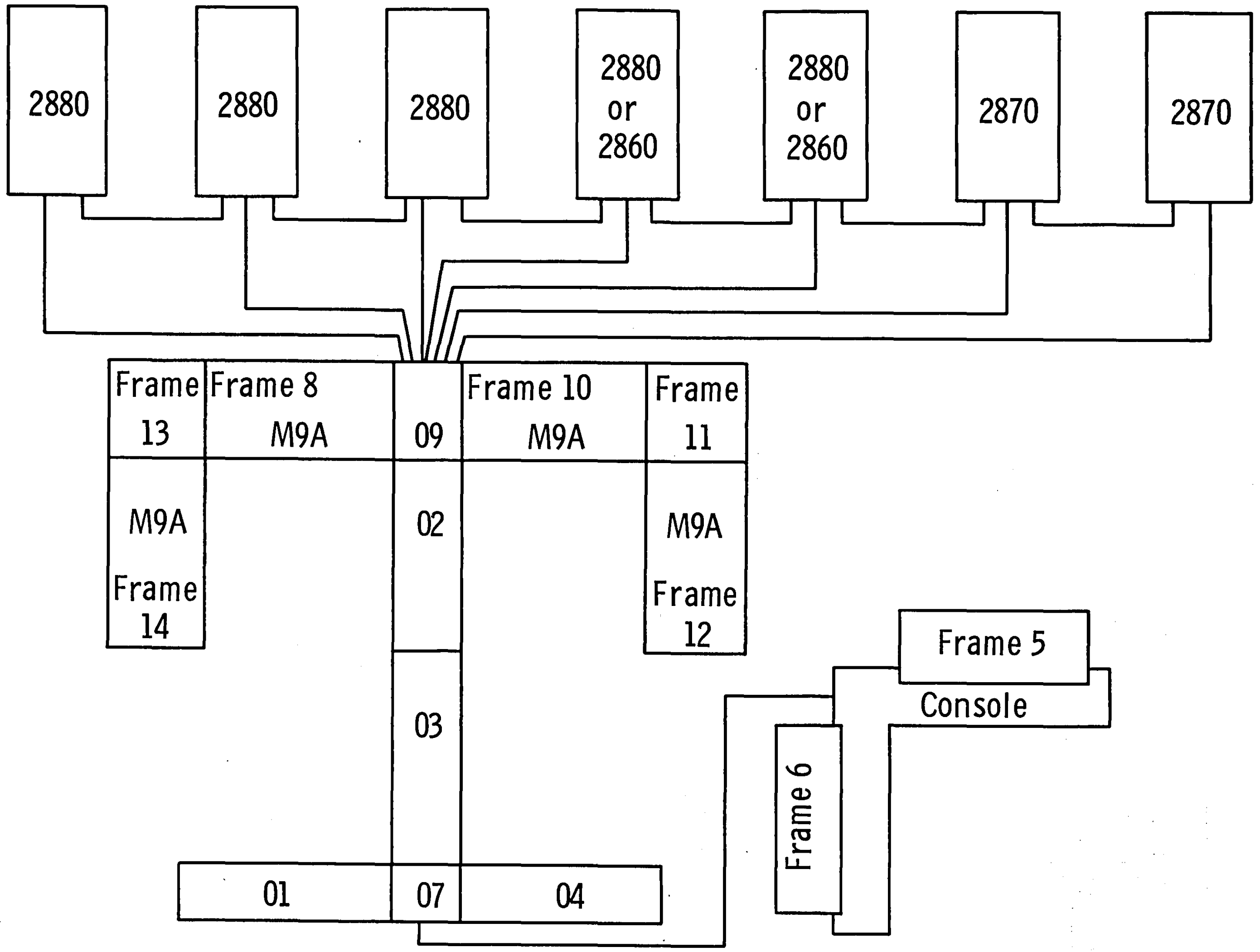


Figure 2-44 - Cabling

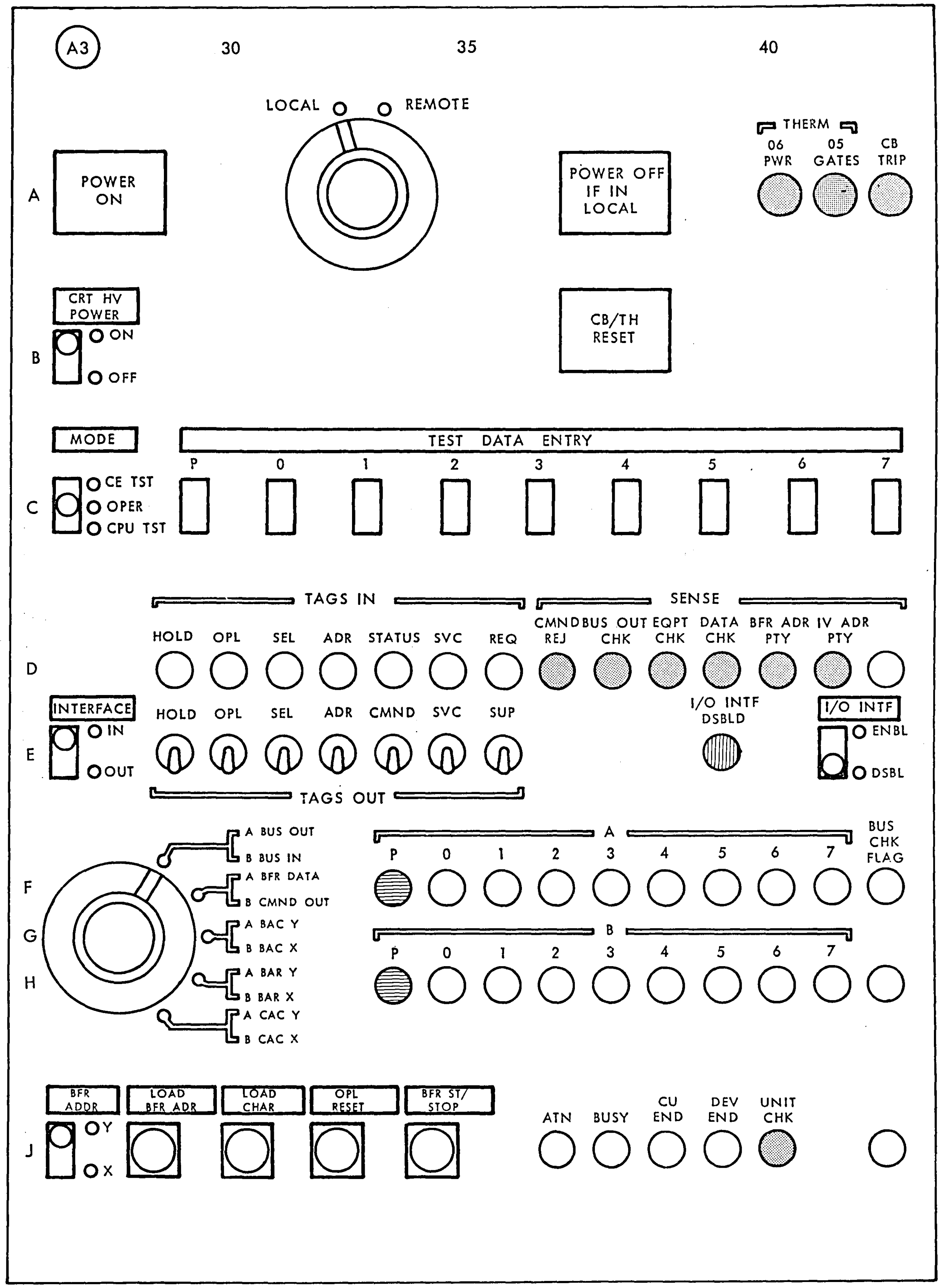


Figure 3-1 - Session 3 Question 1

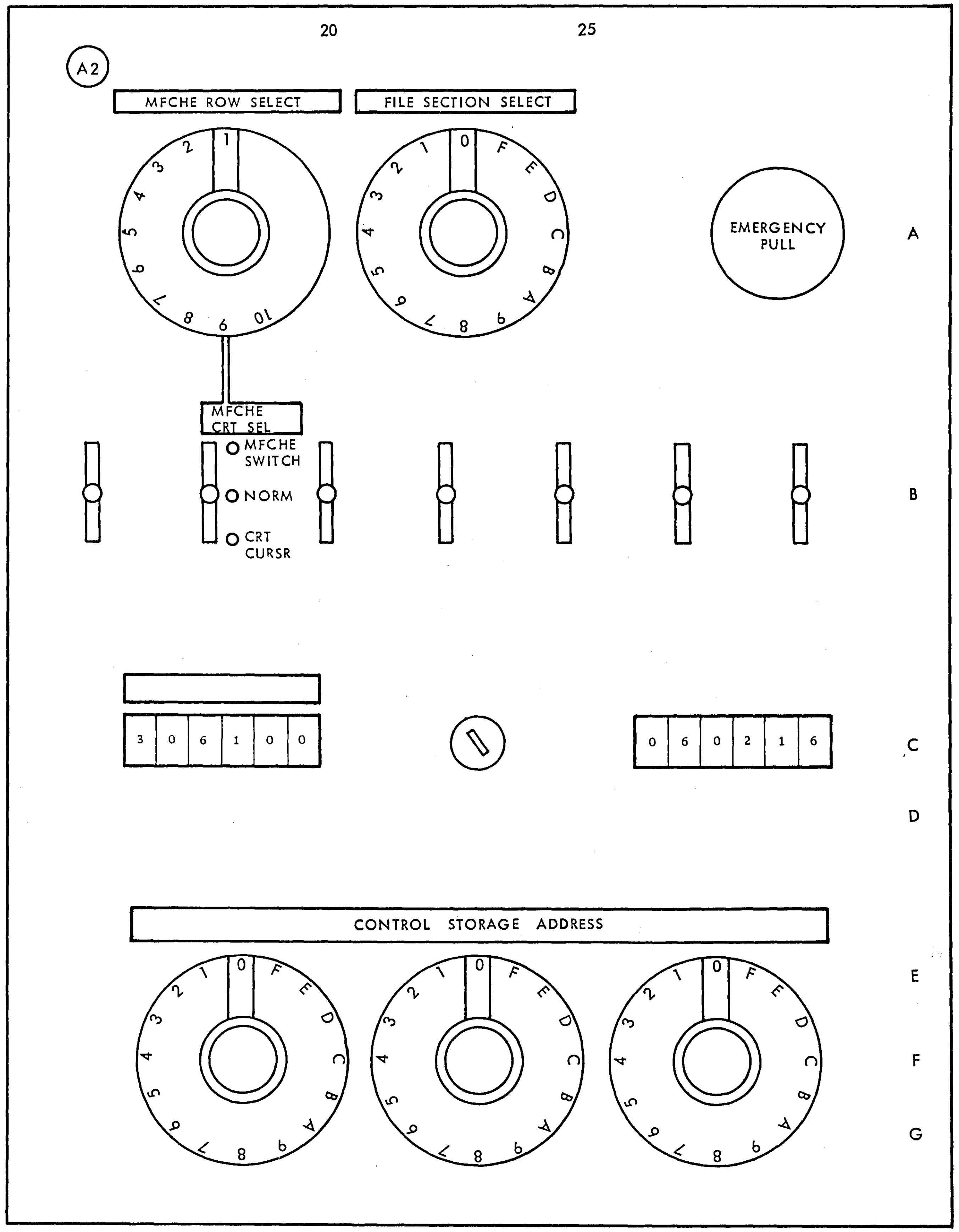


Figure 3-2 - Session 3 Question 2

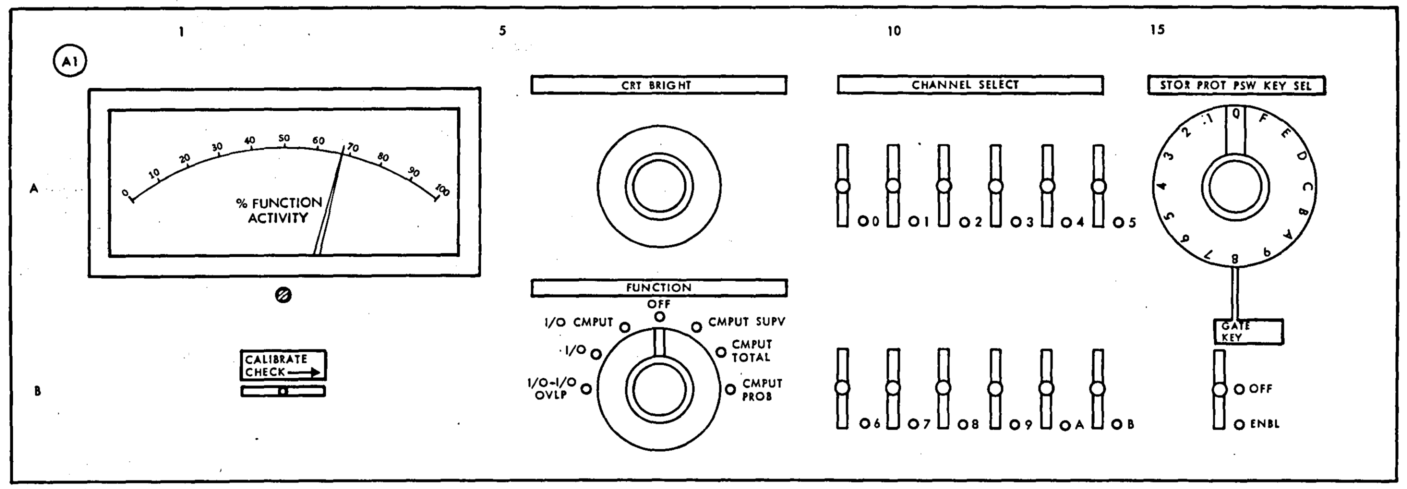


Figure 3-3 - Session 3 Question 3

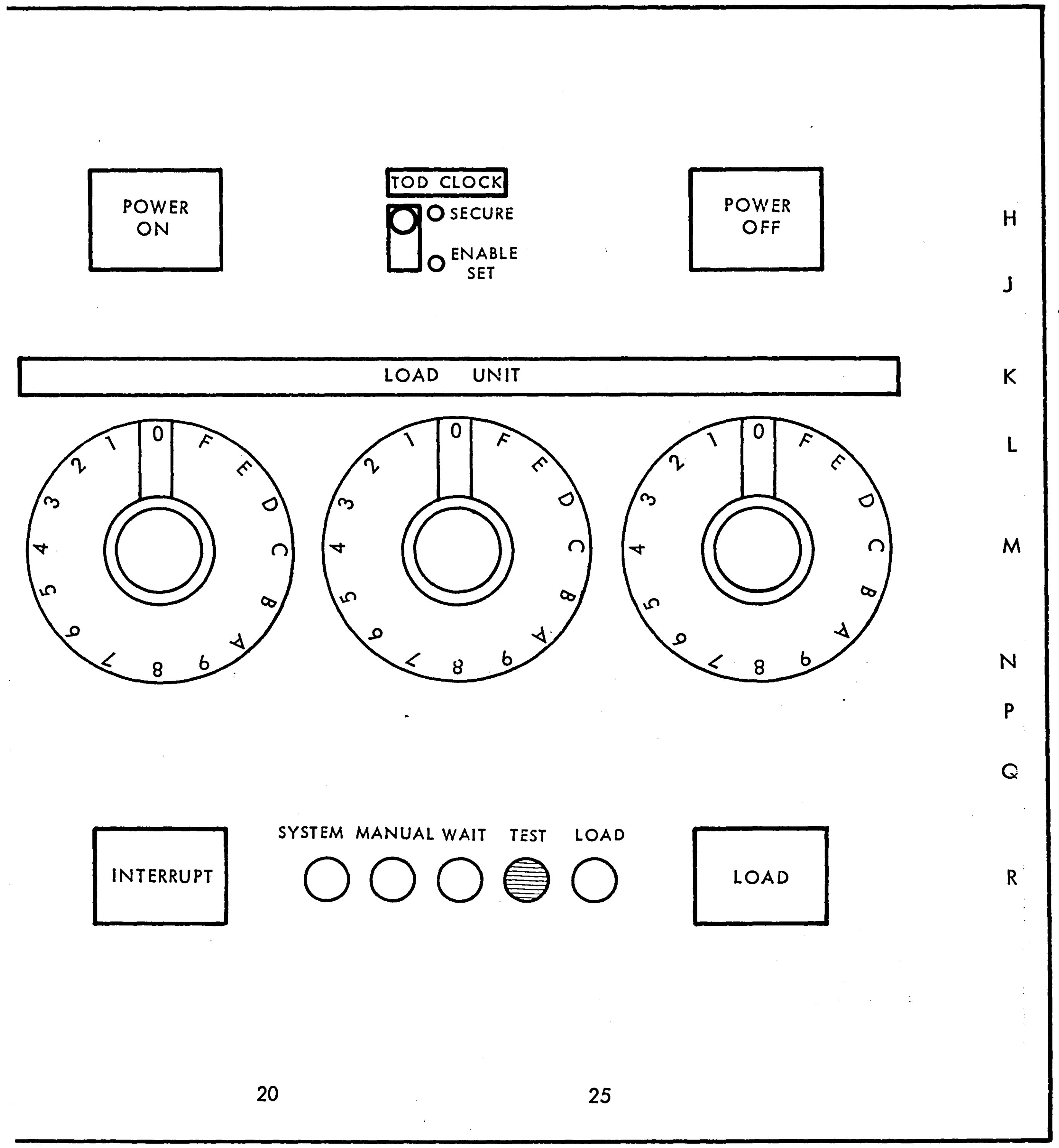


Figure 3-4 - Session 3 Question 4

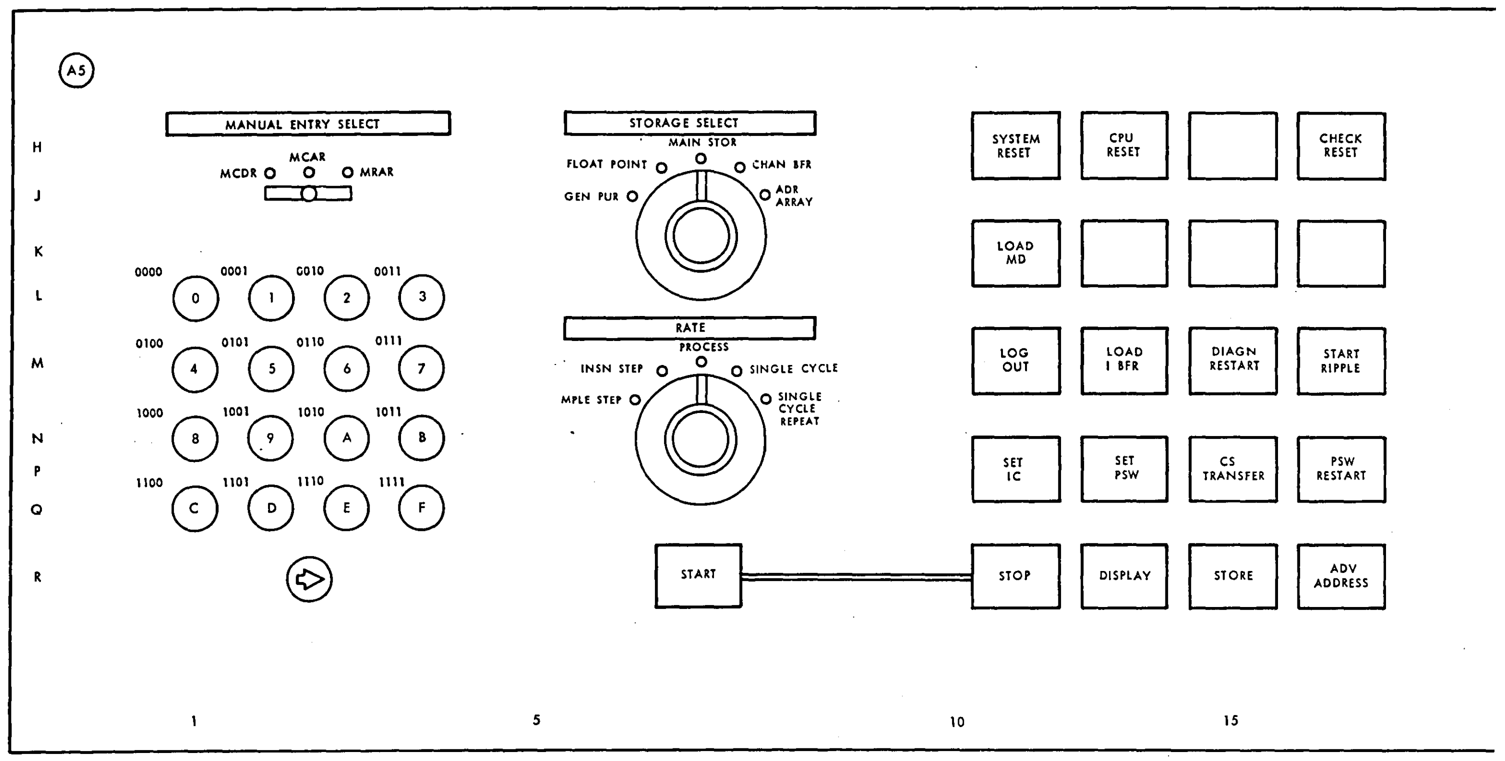


Figure 3-5 - Session 3 Question 5

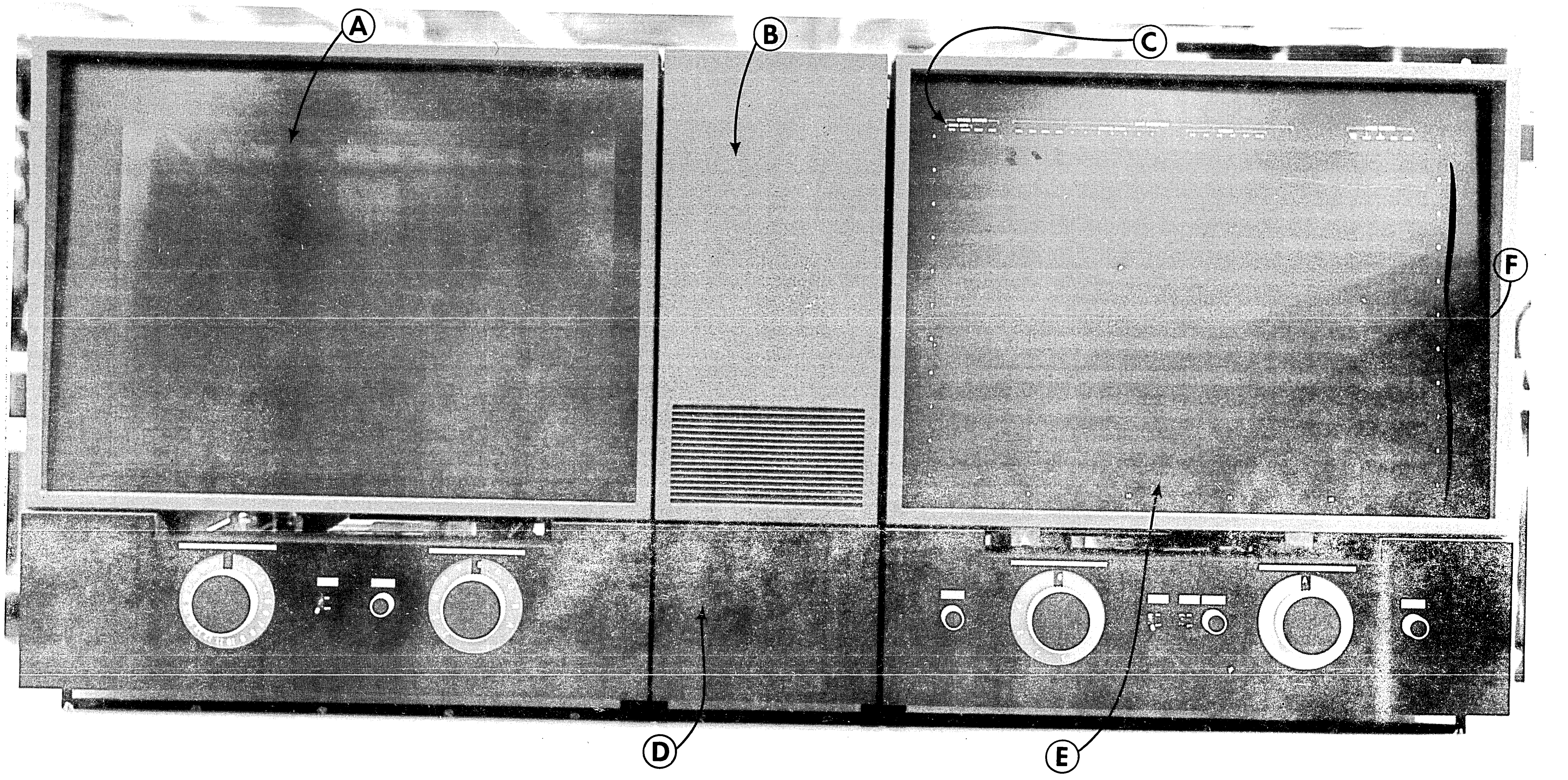


Figure 3-6 - Session 3 Questions 6-11

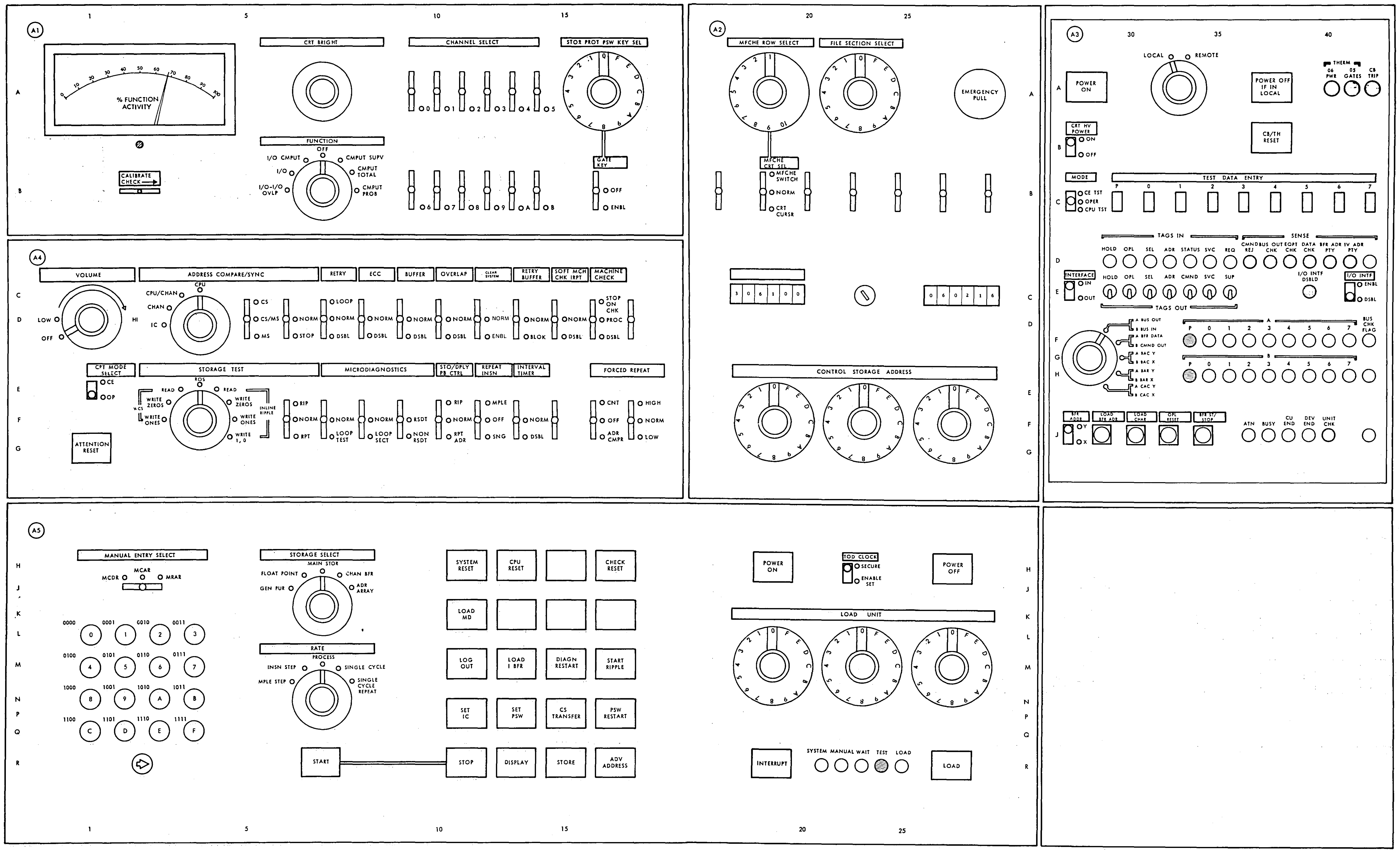


Figure 3-7 - Control Panel

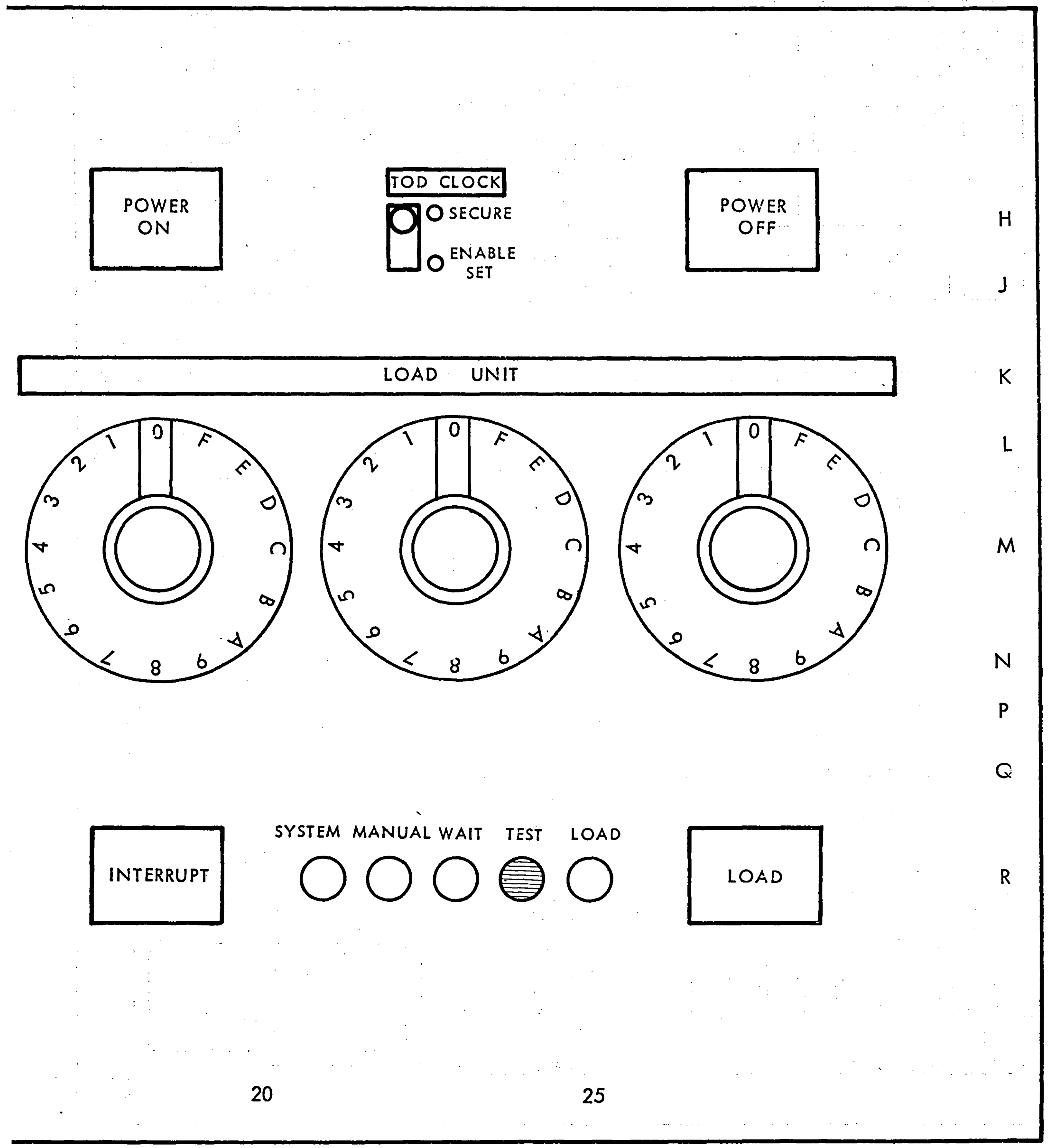
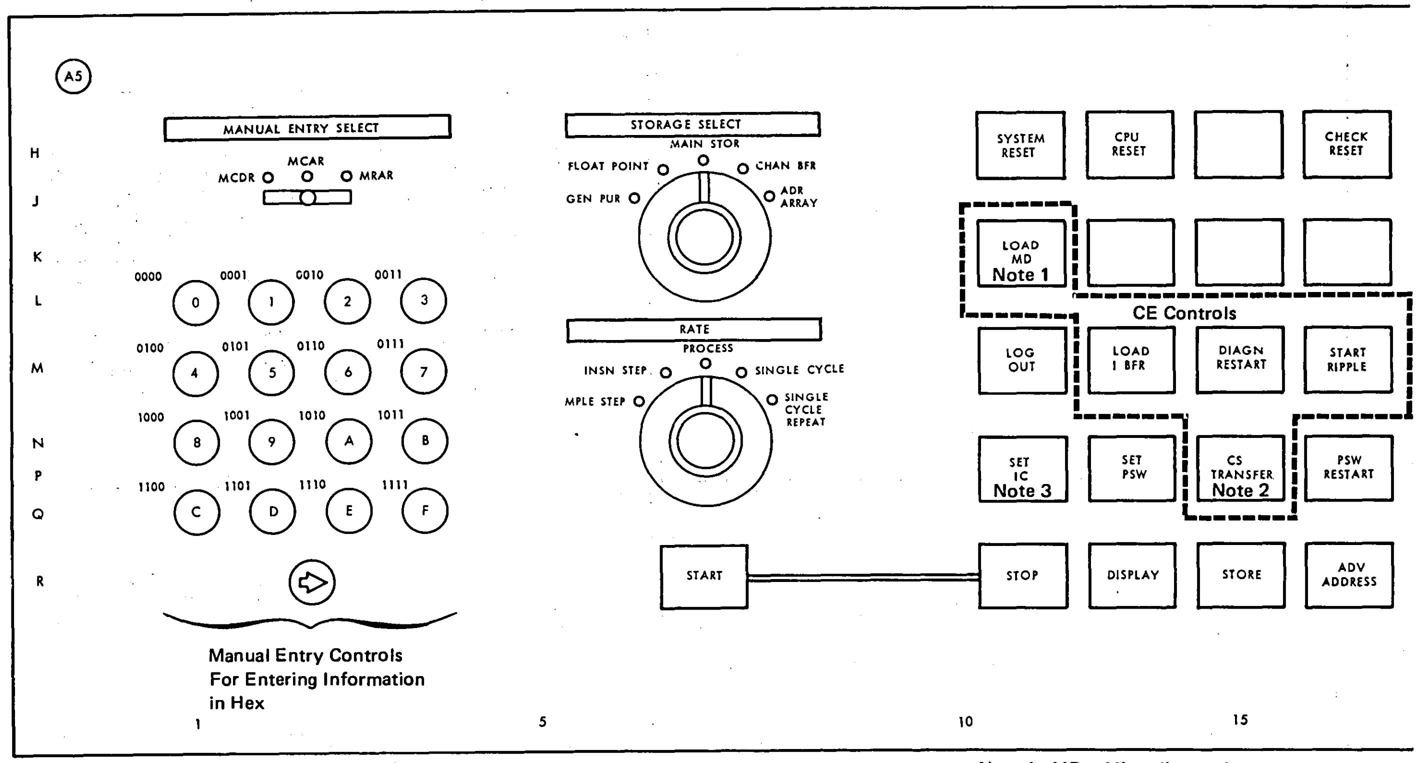


Figure 3-8 - Operator Controls



Note 1 MD = Microdiagnostics
Note 2 CS = Control Storage
Note 3 IC = Instruction Counter

Figure 3-9 - Intervention Controls

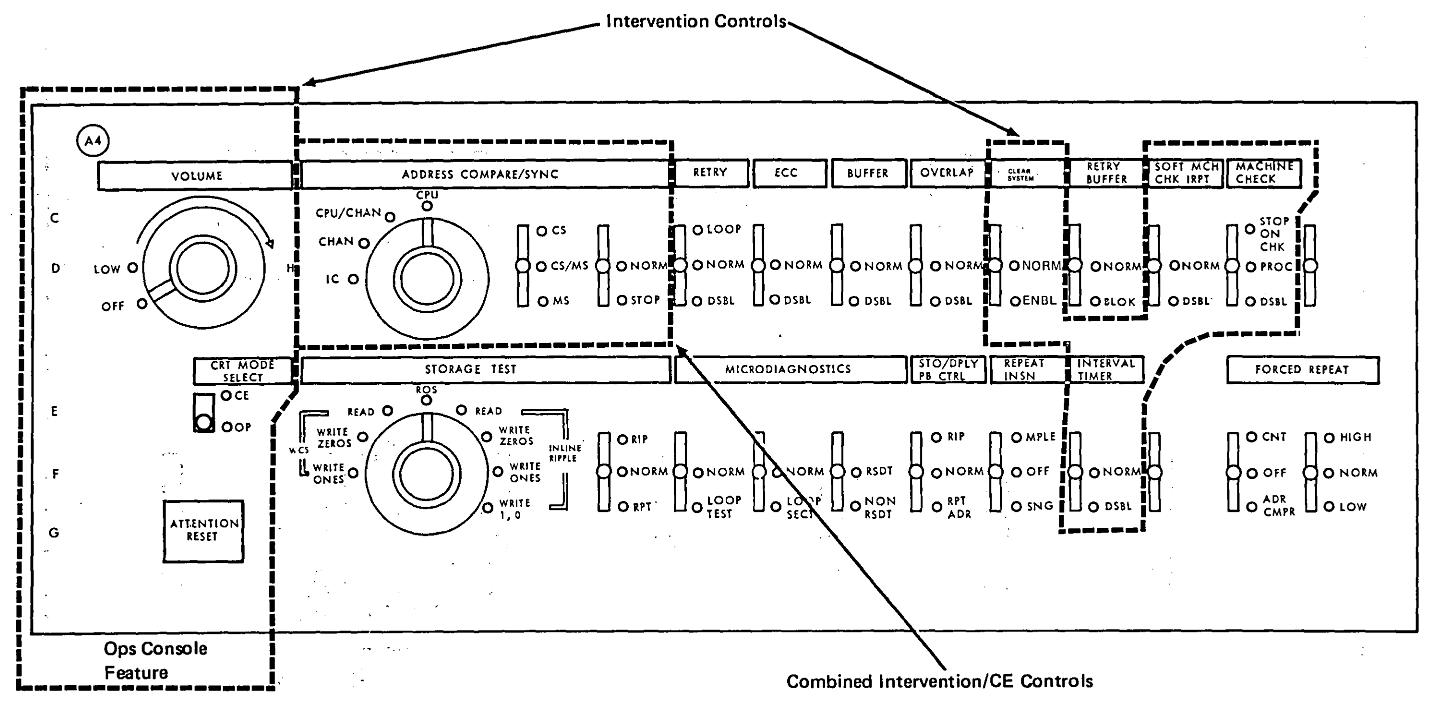


Figure 3-10 - CE/Intervention Controls

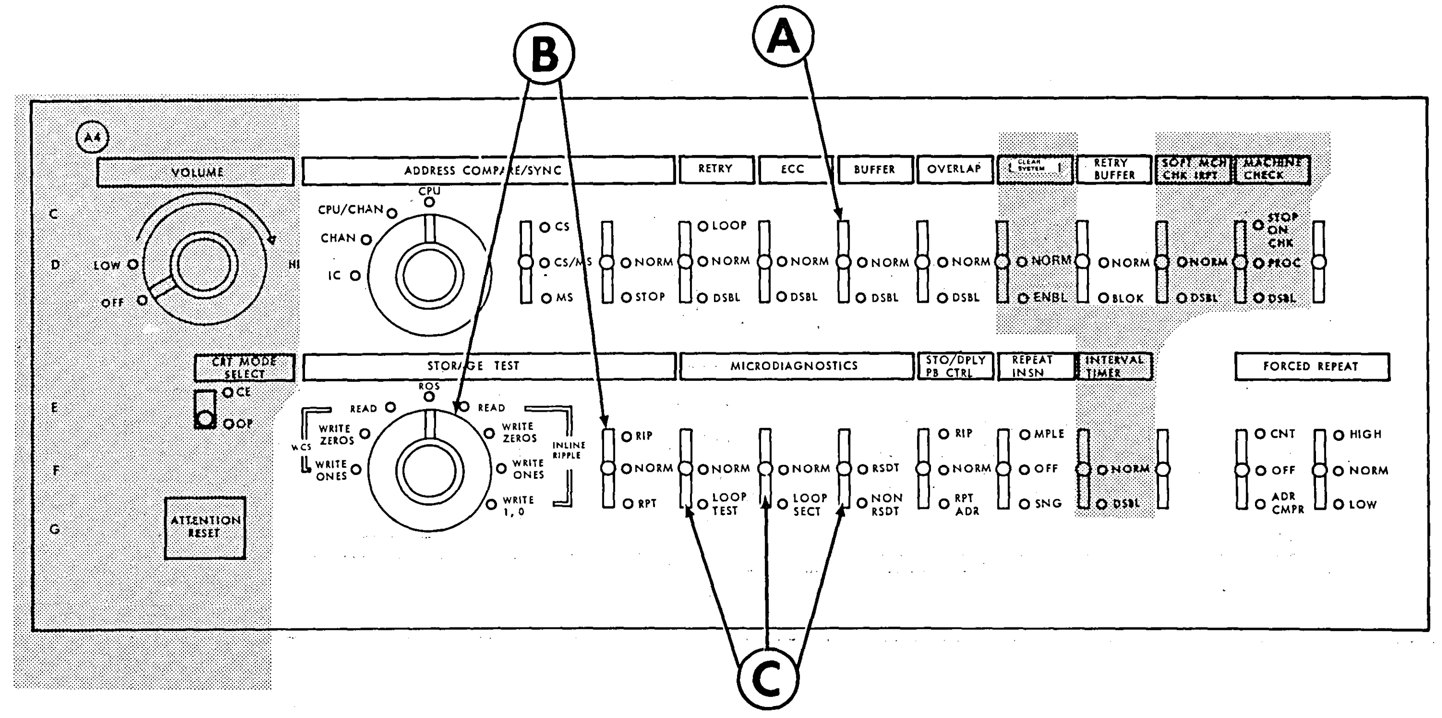


Figure 3-11 - CE Controls

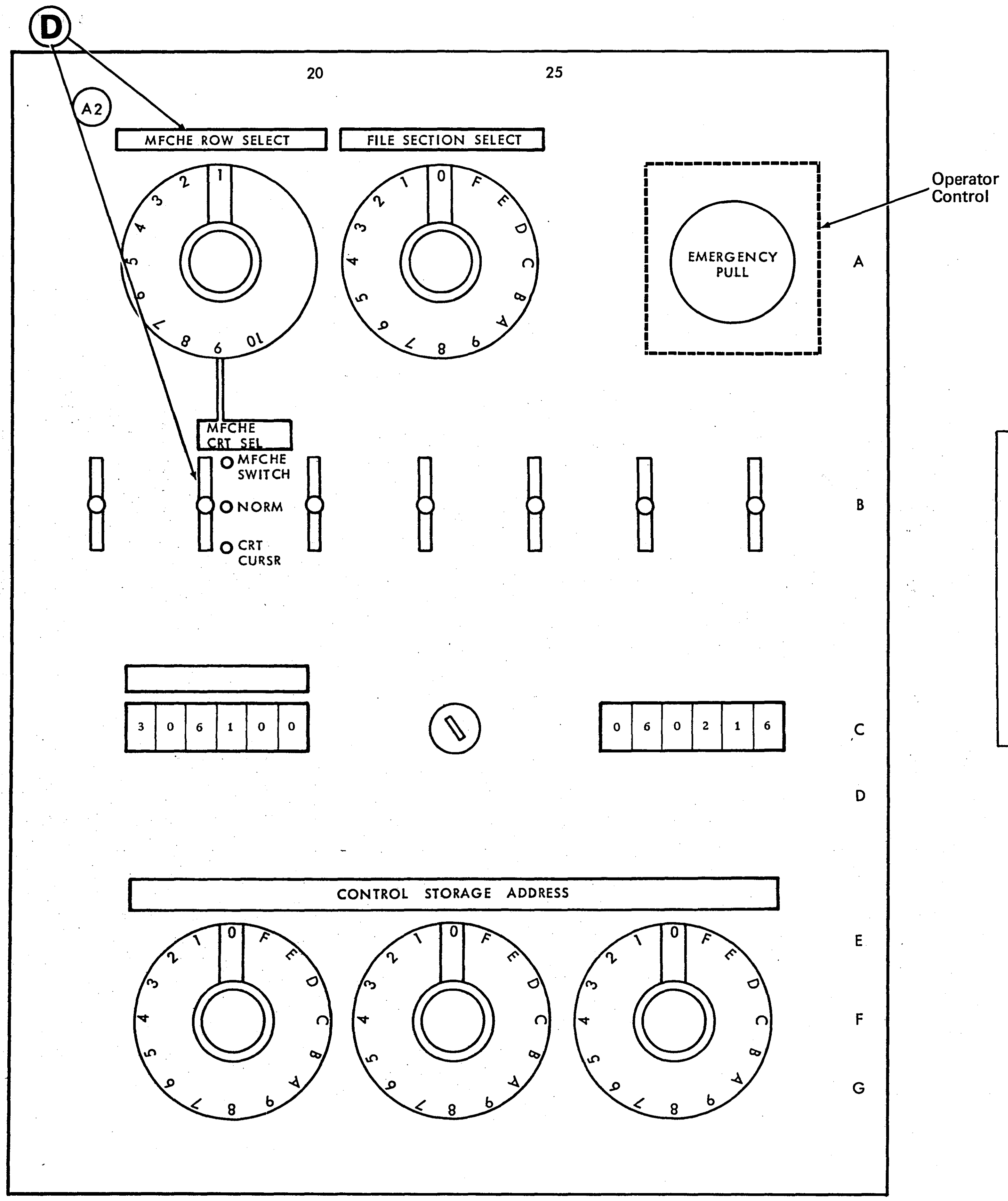


Figure 3-12 - CE Controls

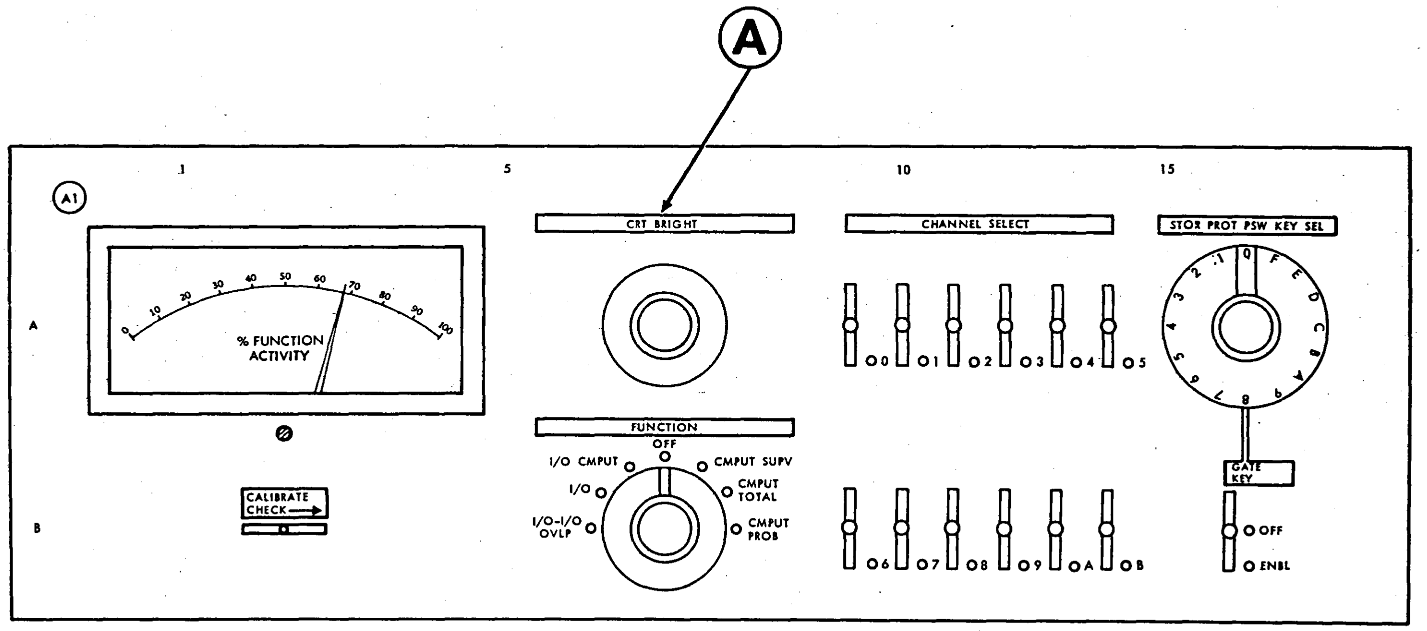


Figure 3-13 - SAM

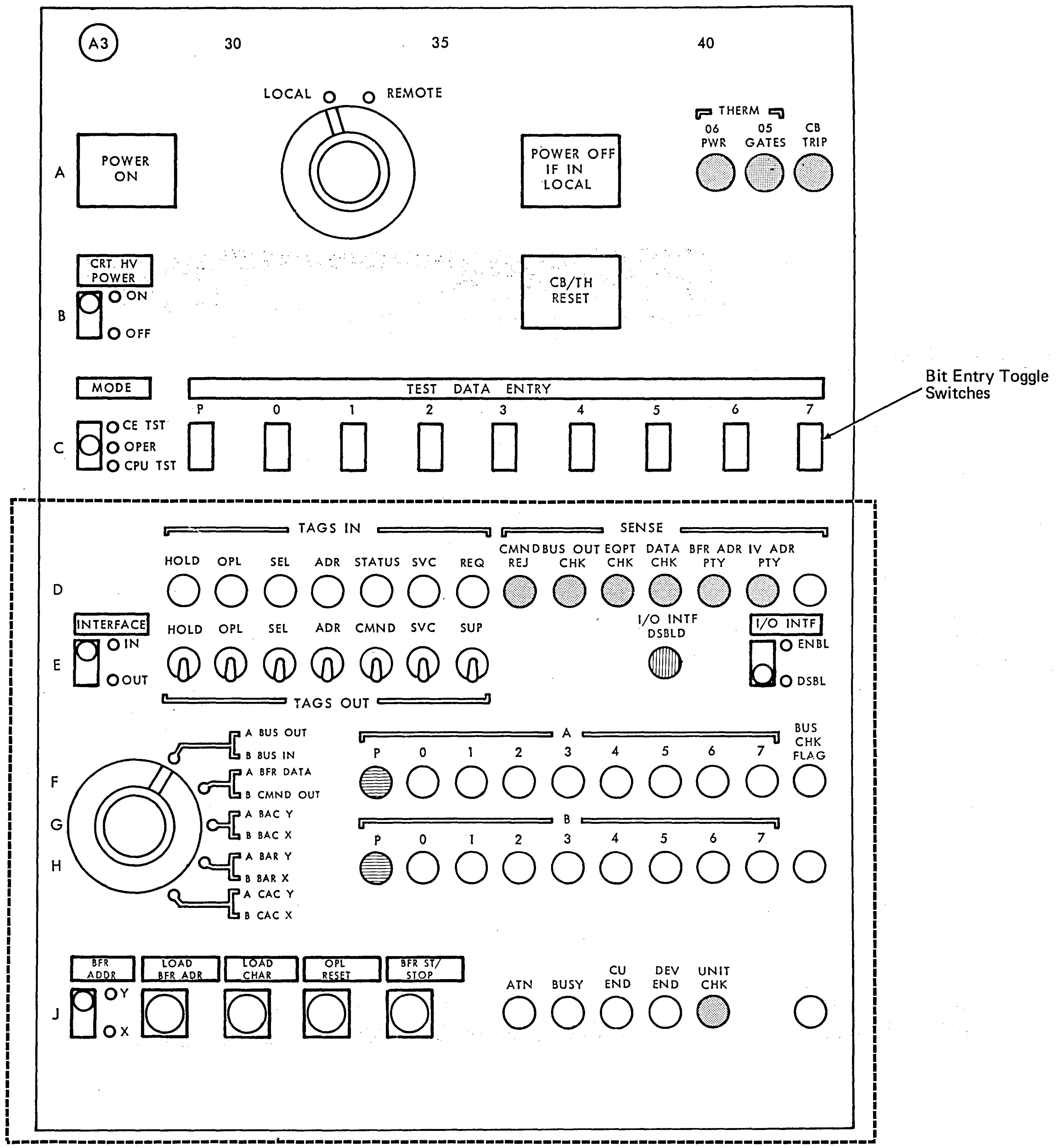


Figure 3-14 - CE Panel for Maintenance Console and Ops Console Feature

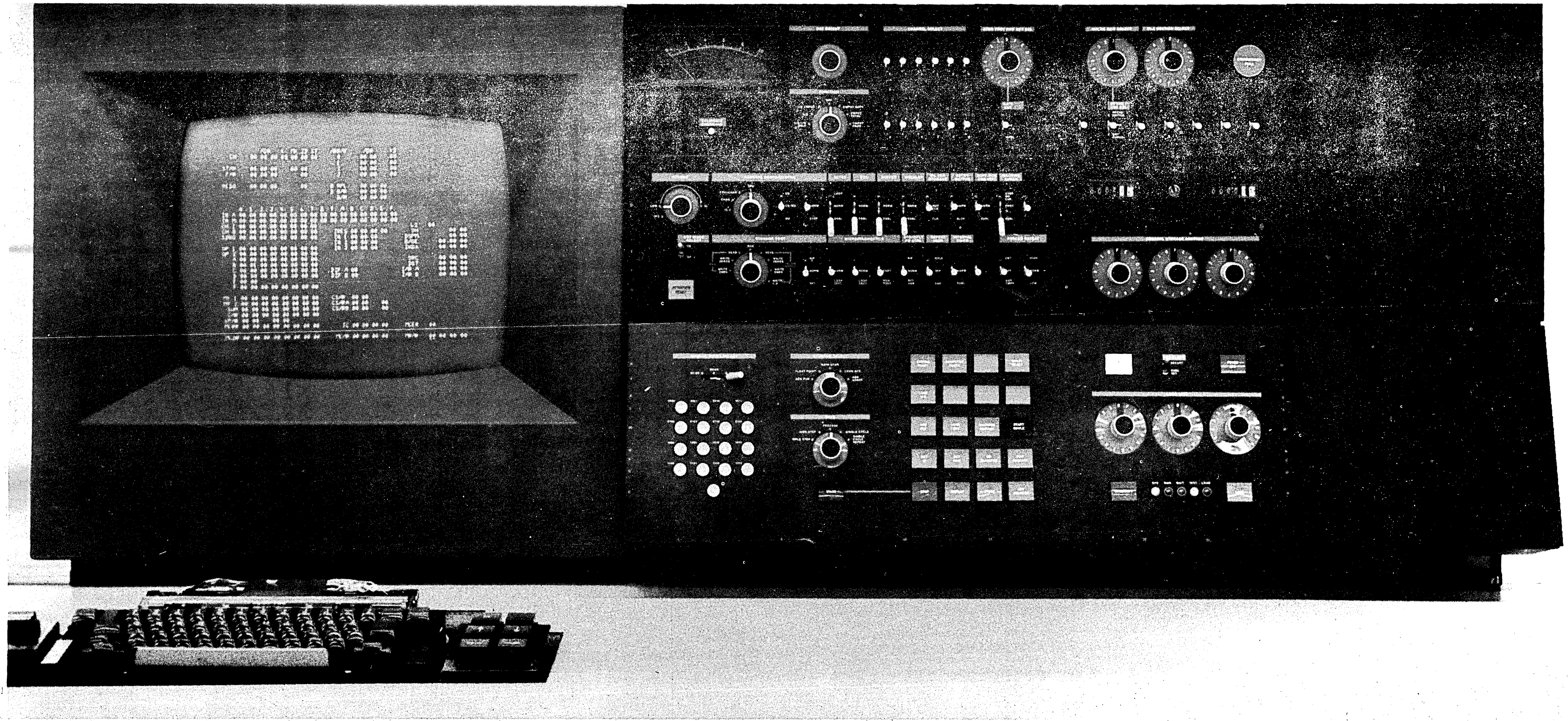


Figure 3-15 -System Control Panel

```
            ADR   K  MK ID ST    ARRAY    ADR     VD
   CH    XX XX XX 0X XX XX 0X      0     XX XX    XX
                                   1     XX XX    XX
 STAR    XX XX XX    0X XX         2     XX XX    XX
  FAR    XX XX XX       XX         3     XX XX    XX

 REDO    XX XX XX       XX       B RPL   XX XX XX
                                 D BAR   XX XX XX
                                   BAR   XX XX XX

         1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
 IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
 IB M XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
 CH I XX XX XX XX XX XX XX XX                            L2  XX
 CH O XX XX XX XX XX XX XX XX     IREG XX XX XX XX       SH       XX XX
 SDBO XX XX XX XX XX XX XX XX     IQ 1 XX XX XX       DSPM       0X XX
                                  IQ 2 XX XX XX       BASE    XX XX XX
 OP 1 XX XX XX XX XX XX XX XX     IQ 3 XX XX XX       INDX    XX XX XX
 OP 2 XX XX XX XX XX XX XX XX
    A XX XX XX XX XX XX XX XX                          SRC    XX XX XX
    B XX XX XX XX XX XX XX XX                          DST    XX XX XX
    C XX XX XX XX XX XX XX XX    DIF A XX            IAR A    XX XX XX
    D XX XX XX XX XX XX XX XX    DIF B XX            IAR B    XX XX XX
    E XX XX
    F XX XX XX XX XX XX XX XX

                                 CSAR  0X XX
                                 CSARA 0X XX    0X
                                 CSARB 0X XX    0X

MCRR XX XX XX XX XX XX XX XX       IC XX XX XX XX    MCER    XX

MCDR XX XX XX XX XX XX XX XX    MCAR  XX XX XX XX    MRAR    XX XX XX XX
```

Figure 3-16 - CRT Display

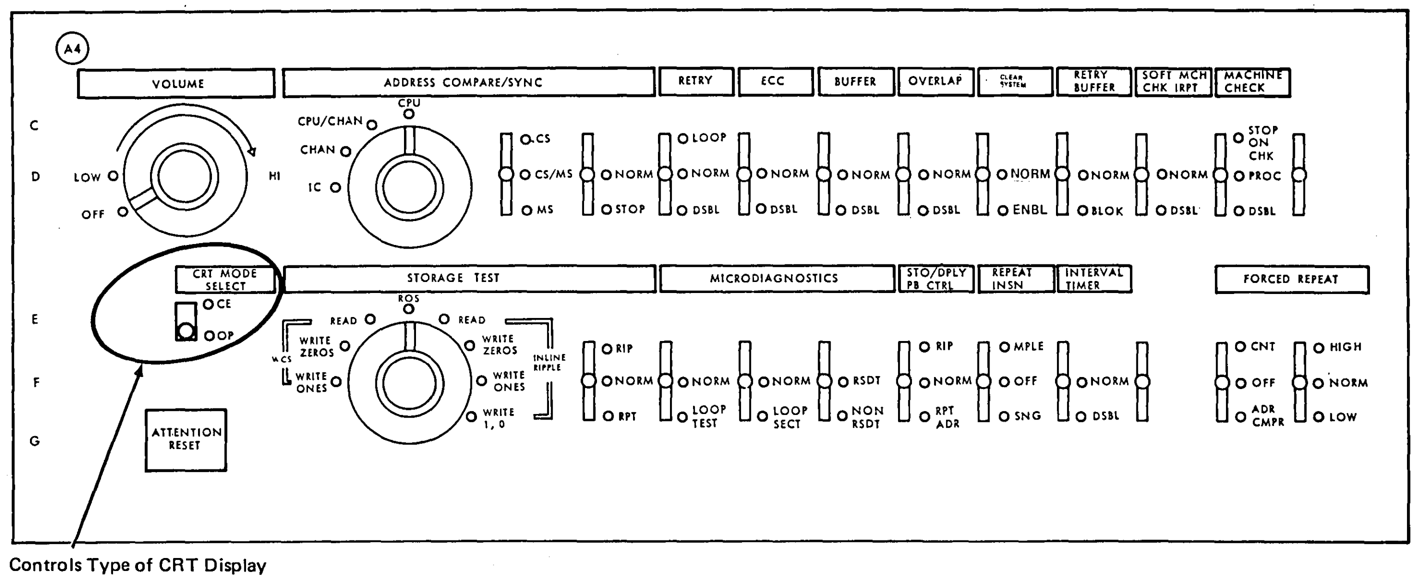
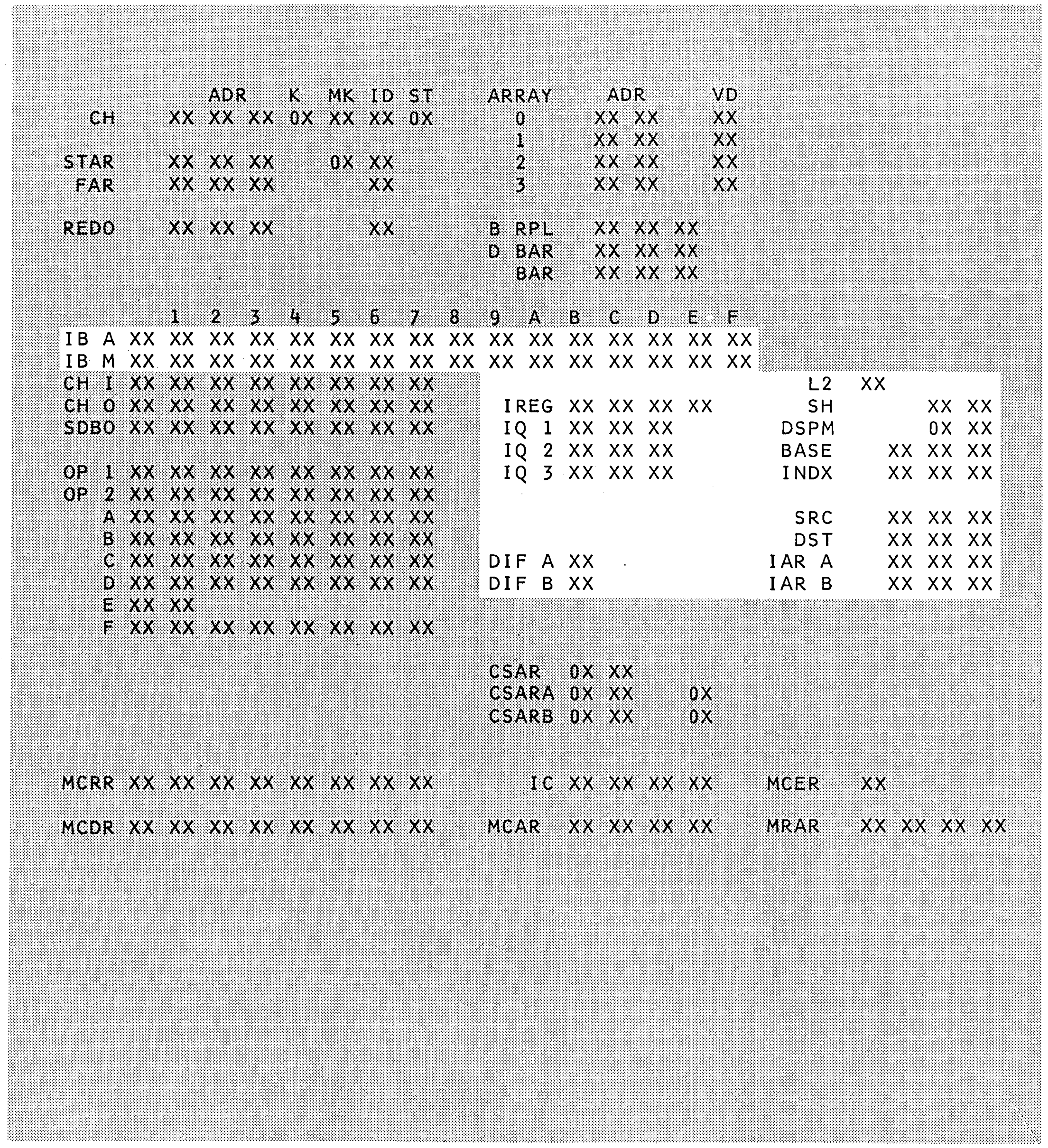
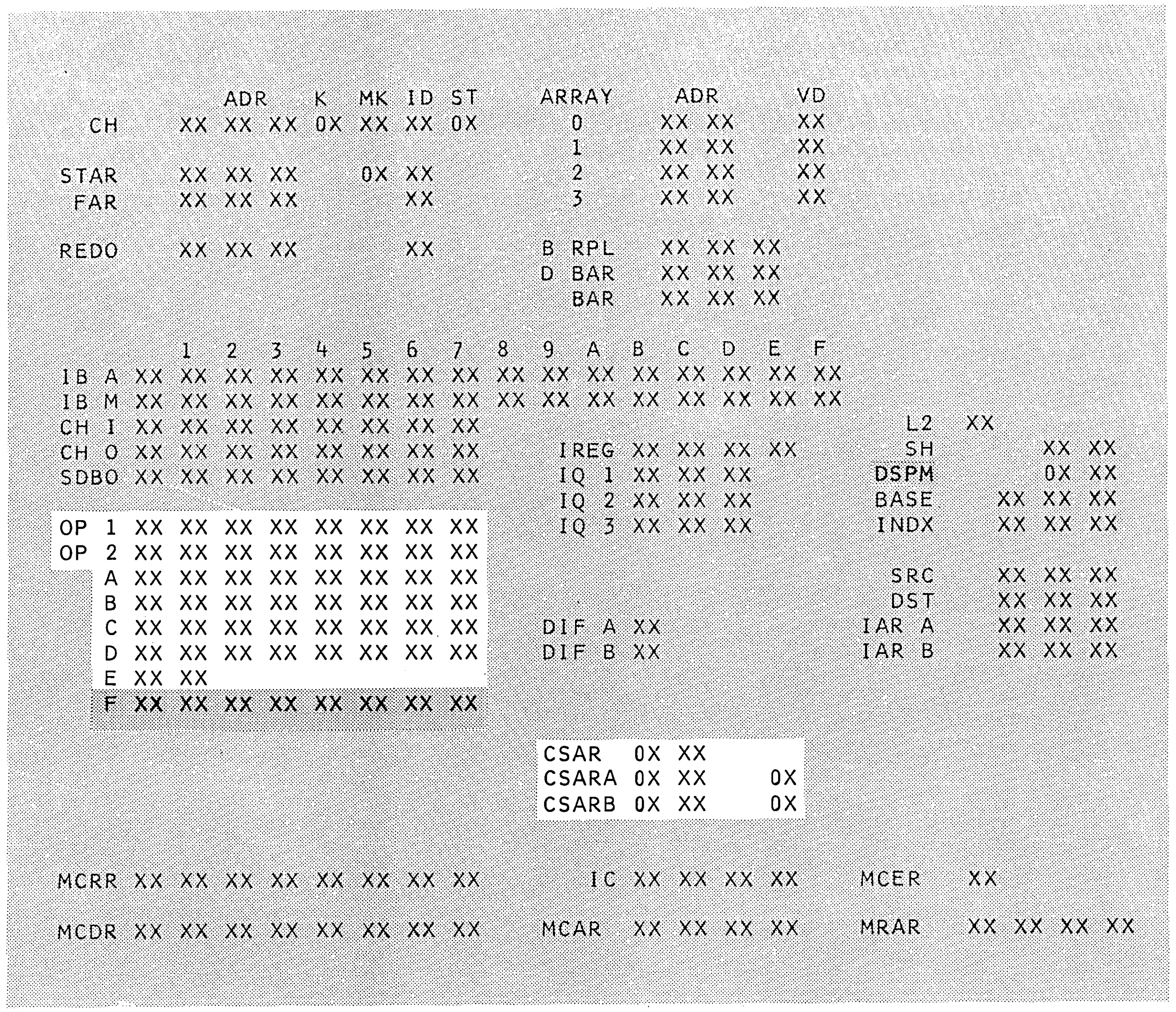


Figure 3-17 - CRT Mode Select Switch



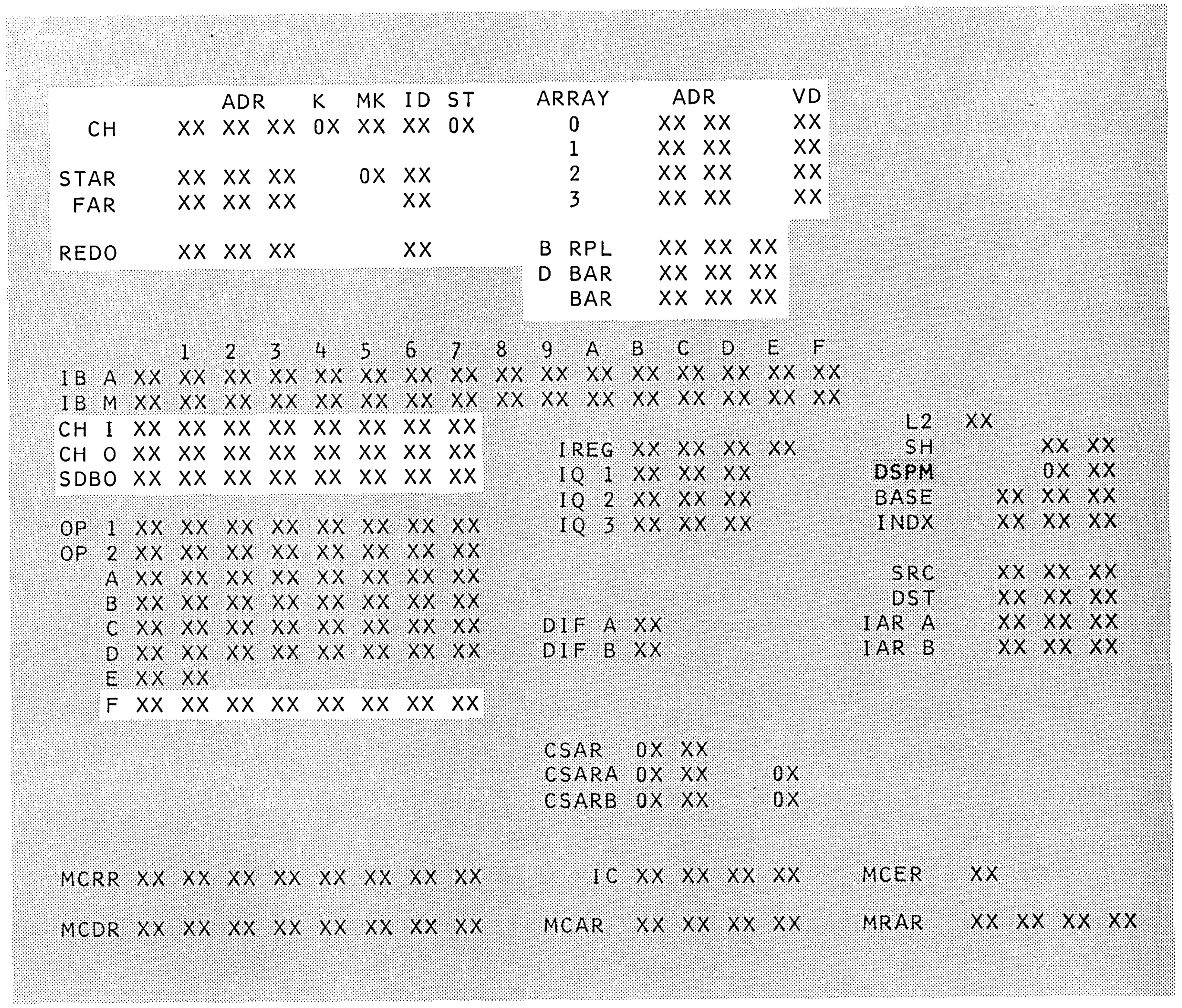
```
           ADR   K  MK ID ST    ARRAY    ADR     VD
  CH    XX XX XX 0X XX XX 0X      0     XX XX     XX
                                  1     XX XX     XX
STAR    XX XX XX    0X XX         2     XX XX     XX
 FAR    XX XX XX       XX         3     XX XX     XX

REDO    XX XX XX       XX       B RPL   XX XX XX
                                D BAR   XX XX XX
                                  BAR   XX XX XX

         1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
IB M XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
CH I XX XX XX XX XX XX XX XX                            L2  XX
CH O XX XX XX XX XX XX XX XX     IREG XX XX XX XX       SH       XX XX
SDBO XX XX XX XX XX XX XX XX     IQ 1 XX XX XX        DSPM       0X XX
                                 IQ 2 XX XX XX        BASE    XX XX XX
OP 1 XX XX XX XX XX XX XX XX     IQ 3 XX XX XX        INDX    XX XX XX
OP 2 XX XX XX XX XX XX XX XX
   A XX XX XX XX XX XX XX XX                           SRC    XX XX XX
   B XX XX XX XX XX XX XX XX                           DST    XX XX XX
   C XX XX XX XX XX XX XX XX    DIF A XX             IAR A    XX XX XX
   D XX XX XX XX XX XX XX XX    DIF B XX             IAR B    XX XX XX
   E XX XX
   F XX XX XX XX XX XX XX XX

                                CSAR  0X XX
                                CSARA 0X XX    0X
                                CSARB 0X XX    0X


MCRR XX XX XX XX XX XX XX XX       IC XX XX XX XX    MCER   XX

MCDR XX XX XX XX XX XX XX XX    MCAR  XX XX XX XX    MRAR   XX XX XX XX
```

Figure 3-18 - CRT Displayed I-Unit Registers



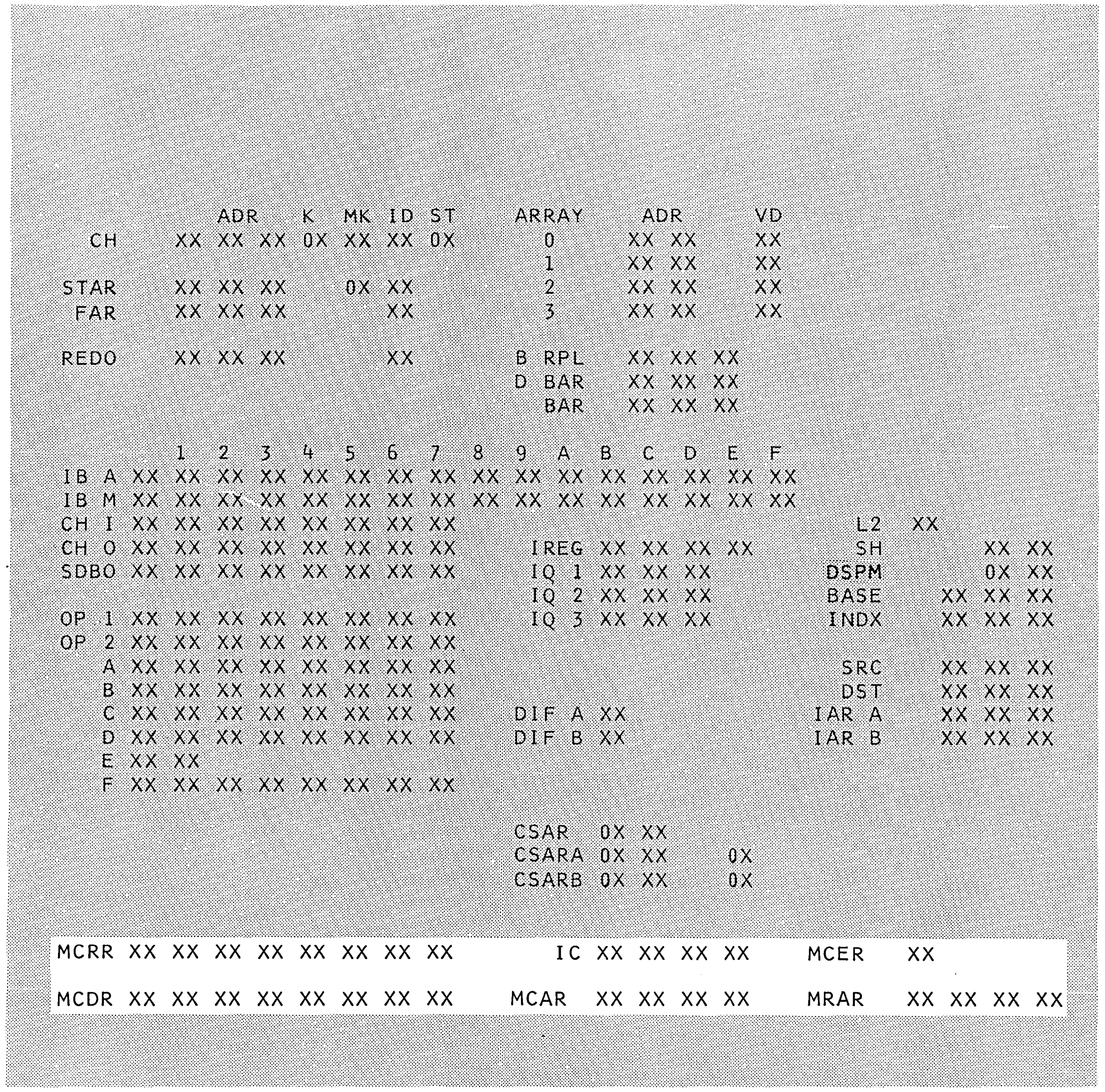
```
            ADR    K  MK ID ST      ARRAY     ADR      VD
  CH     XX XX XX 0X XX XX 0X         0      XX XX     XX
                                      1      XX XX     XX
STAR     XX XX XX    0X XX            2      XX XX     XX
 FAR     XX XX XX       XX            3      XX XX     XX

REDO     XX XX XX       XX          B RPL    XX XX XX
                                    D BAR    XX XX XX
                                      BAR    XX XX XX

           1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
IB M XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
CH I XX XX XX XX XX XX XX XX                            L2  XX
CH O XX XX XX XX XX XX XX XX     IREG XX XX XX XX       SH         XX XX
SDBO XX XX XX XX XX XX XX XX     IQ 1 XX XX XX        DSPM         0X XX
                                 IQ 2 XX XX XX        BASE      XX XX XX
OP 1 XX XX XX XX XX XX XX XX     IQ 3 XX XX XX        INDX      XX XX XX
OP 2 XX XX XX XX XX XX XX XX
   A XX XX XX XX XX XX XX XX                           SRC      XX XX XX
   B XX XX XX XX XX XX XX XX                           DST      XX XX XX
   C XX XX XX XX XX XX XX XX   DIF A XX              IAR A      XX XX XX
   D XX XX XX XX XX XX XX XX   DIF B XX              IAR B      XX XX XX
   E XX XX
   F XX XX XX XX XX XX XX XX

                               CSAR  0X XX
                               CSARA 0X XX    0X
                               CSARB 0X XX    0X

MCRR XX XX XX XX XX XX XX XX        IC XX XX XX XX      MCER   XX

MCDR XX XX XX XX XX XX XX XX   MCAR  XX XX XX XX        MRAR   XX XX XX XX
```

Figure 3-19 CRT Displayed E-Unit Registers



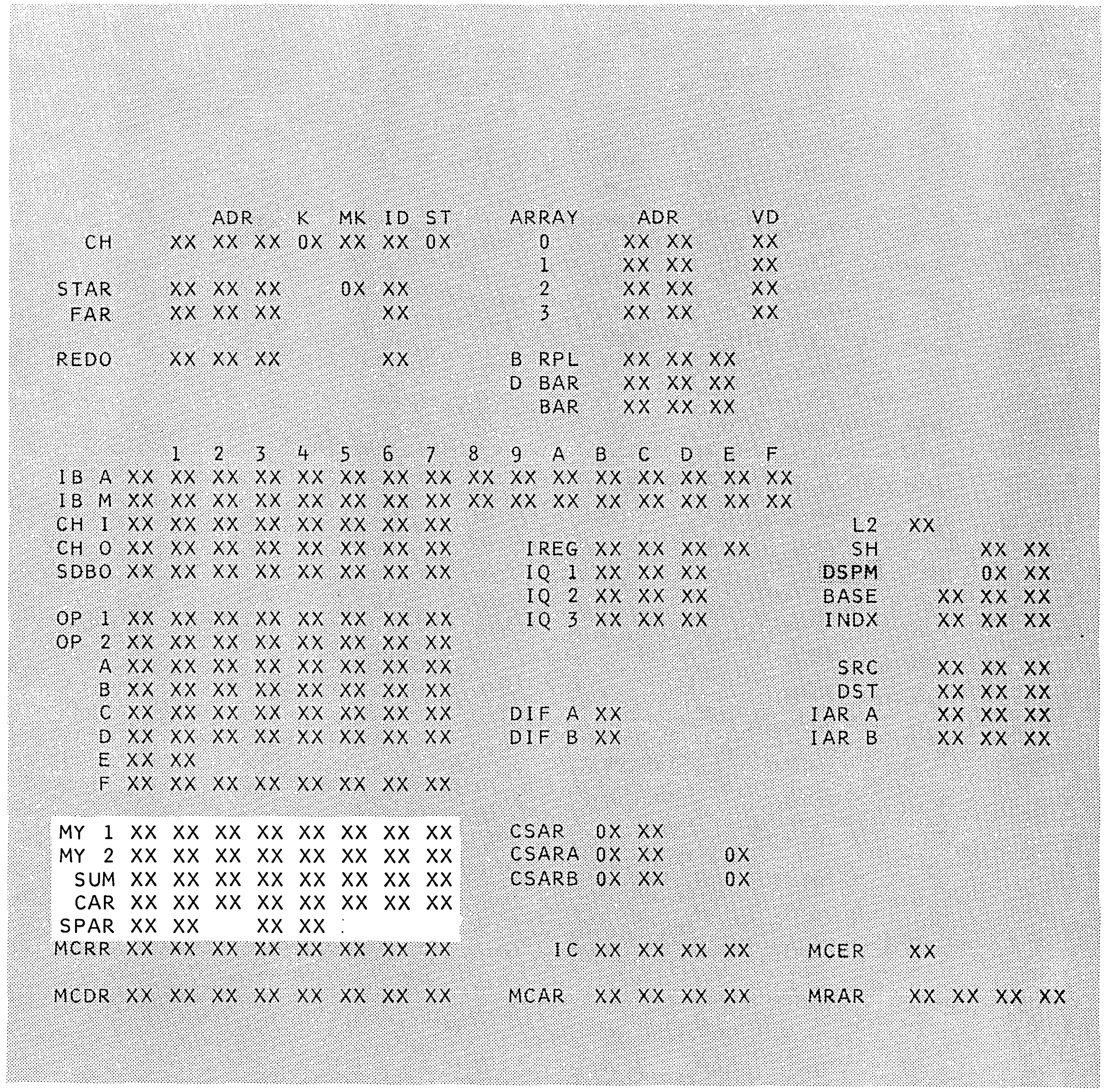
```
            ADR    K  MK ID ST      ARRAY     ADR      VD
  CH     XX XX XX 0X XX XX 0X         0      XX XX     XX
                                      1      XX XX     XX
STAR     XX XX XX    0X XX            2      XX XX     XX
 FAR     XX XX XX       XX            3      XX XX     XX

REDO     XX XX XX       XX          B RPL    XX XX XX
                                    D BAR    XX XX XX
                                      BAR    XX XX XX

           1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
IB M XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
CH I XX XX XX XX XX XX XX XX                            L2  XX
CH O XX XX XX XX XX XX XX XX     IREG XX XX XX XX       SH         XX XX
SDBO XX XX XX XX XX XX XX XX     IQ 1 XX XX XX        DSPM         0X XX
                                 IQ 2 XX XX XX        BASE      XX XX XX
OP 1 XX XX XX XX XX XX XX XX     IQ 3 XX XX XX        INDX      XX XX XX
OP 2 XX XX XX XX XX XX XX XX
   A XX XX XX XX XX XX XX XX                           SRC      XX XX XX
   B XX XX XX XX XX XX XX XX                           DST      XX XX XX
   C XX XX XX XX XX XX XX XX   DIF A XX              IAR A      XX XX XX
   D XX XX XX XX XX XX XX XX   DIF B XX              IAR B      XX XX XX
   E XX XX
   F XX XX XX XX XX XX XX XX

                               CSAR  0X XX
                               CSARA 0X XX    0X
                               CSARB 0X XX    0X

MCRR XX XX XX XX XX XX XX XX        IC XX XX XX XX      MCER   XX

MCDR XX XX XX XX XX XX XX XX   MCAR  XX XX XX XX        MRAR   XX XX XX XX
```

Figure 3-20 CRT Displayed SCU Registers



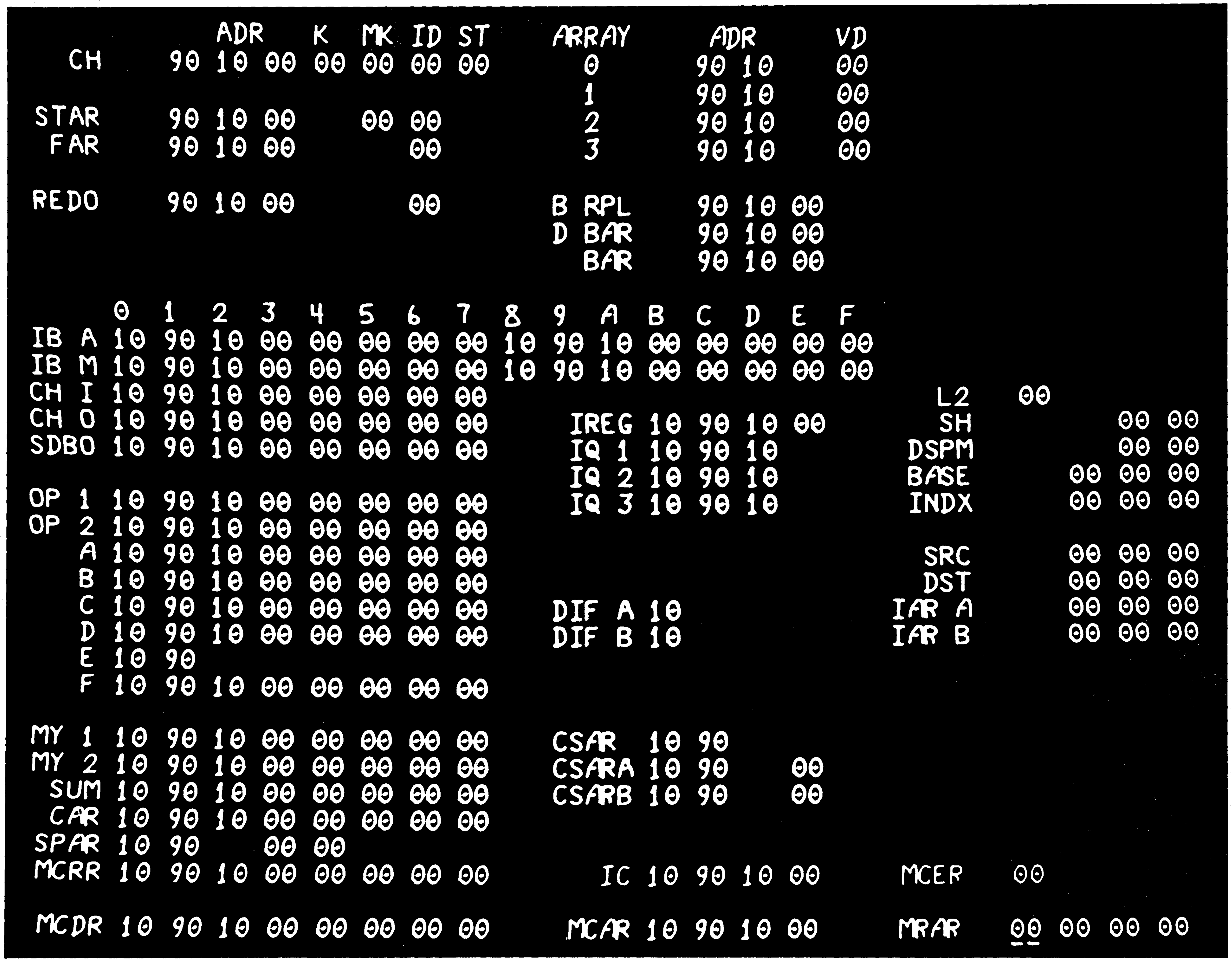
```
                 ADR    K  MK ID ST     ARRAY     ADR      VD
   CH        XX XX XX 0X XX XX 0X         0      XX XX     XX
                                          1      XX XX     XX
  STAR       XX XX XX    0X XX            2      XX XX     XX
   FAR       XX XX XX       XX            3      XX XX     XX

  REDO       XX XX XX       XX          B RPL   XX XX XX
                                        D BAR   XX XX XX
                                          BAR   XX XX XX

          1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
  IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
  IB M XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
  CH I XX XX XX XX XX XX XX XX                                   L2  XX
  CH O XX XX XX XX XX XX XX XX     IREG XX XX XX XX              SH        XX XX
  SDBO XX XX XX XX XX XX XX XX     IQ 1 XX XX XX               DSPM        0X XX
                                   IQ 2 XX XX XX               BASE     XX XX XX
  OP 1 XX XX XX XX XX XX XX XX     IQ 3 XX XX XX               INDX     XX XX XX
  OP 2 XX XX XX XX XX XX XX XX
     A XX XX XX XX XX XX XX XX                                  SRC     XX XX XX
     B XX XX XX XX XX XX XX XX                                  DST     XX XX XX
     C XX XX XX XX XX XX XX XX    DIF A XX                    IAR A     XX XX XX
     D XX XX XX XX XX XX XX XX    DIF B XX                    IAR B     XX XX XX
     E XX XX
     F XX XX XX XX XX XX XX XX

                                  CSAR  0X XX
                                  CSARA 0X XX     0X
                                  CSARB 0X XX     0X

 MCRR XX XX XX XX XX XX XX XX        IC XX XX XX XX     MCER    XX

 MCDR XX XX XX XX XX XX XX XX     MCAR  XX XX XX XX     MRAR    XX XX XX XX
```

Figure 3-21 CRT Displayed Maint Control Registers



```
                 ADR    K  MK ID ST     ARRAY     ADR      VD
   CH        XX XX XX 0X XX XX 0X         0      XX XX     XX
                                          1      XX XX     XX
  STAR       XX XX XX    0X XX            2      XX XX     XX
   FAR       XX XX XX       XX            3      XX XX     XX

  REDO       XX XX XX       XX          B RPL   XX XX XX
                                        D BAR   XX XX XX
                                          BAR   XX XX XX

          1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
  IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
  IB M XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
  CH I XX XX XX XX XX XX XX XX                                   L2  XX
  CH O XX XX XX XX XX XX XX XX     IREG XX XX XX XX              SH        XX XX
  SDBO XX XX XX XX XX XX XX XX     IQ 1 XX XX XX               DSPM        0X XX
                                   IQ 2 XX XX XX               BASE     XX XX XX
  OP 1 XX XX XX XX XX XX XX XX     IQ 3 XX XX XX               INDX     XX XX XX
  OP 2 XX XX XX XX XX XX XX XX
     A XX XX XX XX XX XX XX XX                                  SRC     XX XX XX
     B XX XX XX XX XX XX XX XX                                  DST     XX XX XX
     C XX XX XX XX XX XX XX XX    DIF A XX                    IAR A     XX XX XX
     D XX XX XX XX XX XX XX XX    DIF B XX                    IAR B     XX XX XX
     E XX XX
     F XX XX XX XX XX XX XX XX

  MY 1 XX XX XX XX XX XX XX XX    CSAR  0X XX
  MY 2 XX XX XX XX XX XX XX XX    CSARA 0X XX     0X
   SUM XX XX XX XX XX XX XX XX    CSARB 0X XX     0X
   CAR XX XX XX XX XX XX XX XX
  SPAR XX XX    XX XX
 MCRR XX XX XX XX XX XX XX XX        IC XX XX XX XX     MCER    XX

 MCDR XX XX XX XX XX XX XX XX     MCAR  XX XX XX XX     MRAR    XX XX XX XX
```

Figure 3-22 CRT Displayed HSM Registers



```
              ADR    K  MK  ID  ST      ARRAY      ADR      VD
   CH      90 10 00 00  00  00  00        0       90 10     00
                                          1       90 10     00
 STAR      90 10 00     00  00            2       90 10     00
  FAR      90 10 00         00            3       90 10     00

 REDO      90 10 00         00          B RPL     90 10 00
                                        D BAR     90 10 00
                                          BAR     90 10 00

       0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
 IB A 10 90 10 00 00 00 00 00 10 90 10 00 00 00 00 00
 IB M 10 90 10 00 00 00 00 00 10 90 10 00 00 00 00 00
 CH I 10 90 10 00 00 00 00 00                              L2   00
 CH O 10 90 10 00 00 00 00 00      IREG 10 90 10 00        SH          00 00
 SDBO 10 90 10 00 00 00 00 00      IQ 1 10 90 10         DSPM          00 00
                                   IQ 2 10 90 10         BASE       00 00 00
 OP 1 10 90 10 00 00 00 00 00      IQ 3 10 90 10         INDX       00 00 00
 OP 2 10 90 10 00 00 00 00 00
    A 10 90 10 00 00 00 00 00                             SRC       00 00 00
    B 10 90 10 00 00 00 00 00                             DST       00 00 00
    C 10 90 10 00 00 00 00 00     DIF A 10               IAR A      00 00 00
    D 10 90 10 00 00 00 00 00     DIF B 10               IAR B      00 00 00
    E 10 90
    F 10 90 10 00 00 00 00 00

 MY 1 10 90 10 00 00 00 00 00     CSAR  10 90
 MY 2 10 90 10 00 00 00 00 00     CSARA 10 90    00
  SUM 10 90 10 00 00 00 00 00     CSARB 10 90    00
  CAR 10 90 10 00 00 00 00 00
 SPAR 10 90    00 00
 MCRR 10 90 10 00 00 00 00 00        IC 10 90 10 00       MCER  00

 MCDR 10 90 10 00 00 00 00 00      MCAR 10 90 10 00       MRAR  00 00 00 00
```

Figure 3-23 CRT CE Mode

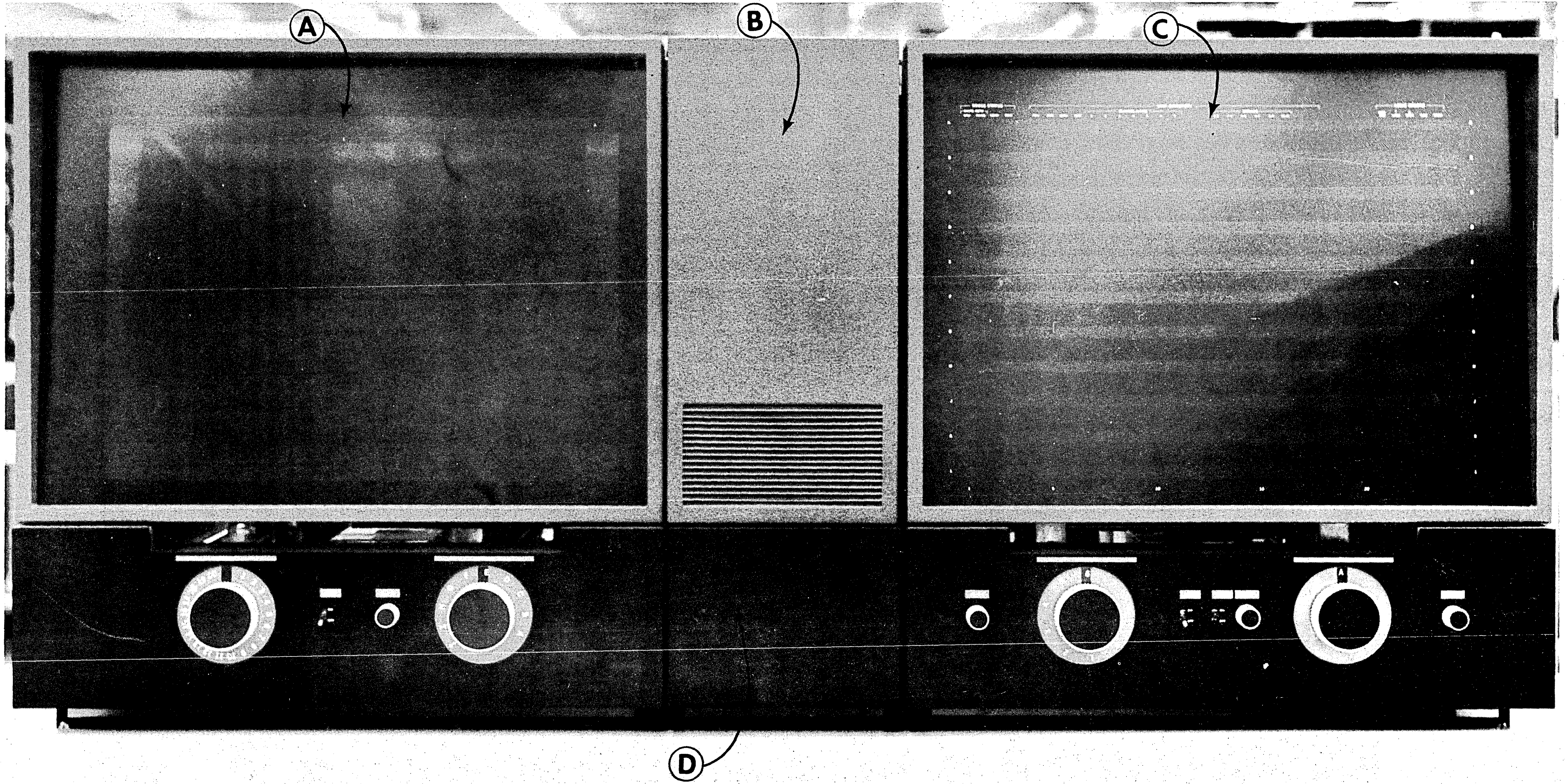


Figure 3-24 Microfiche Console

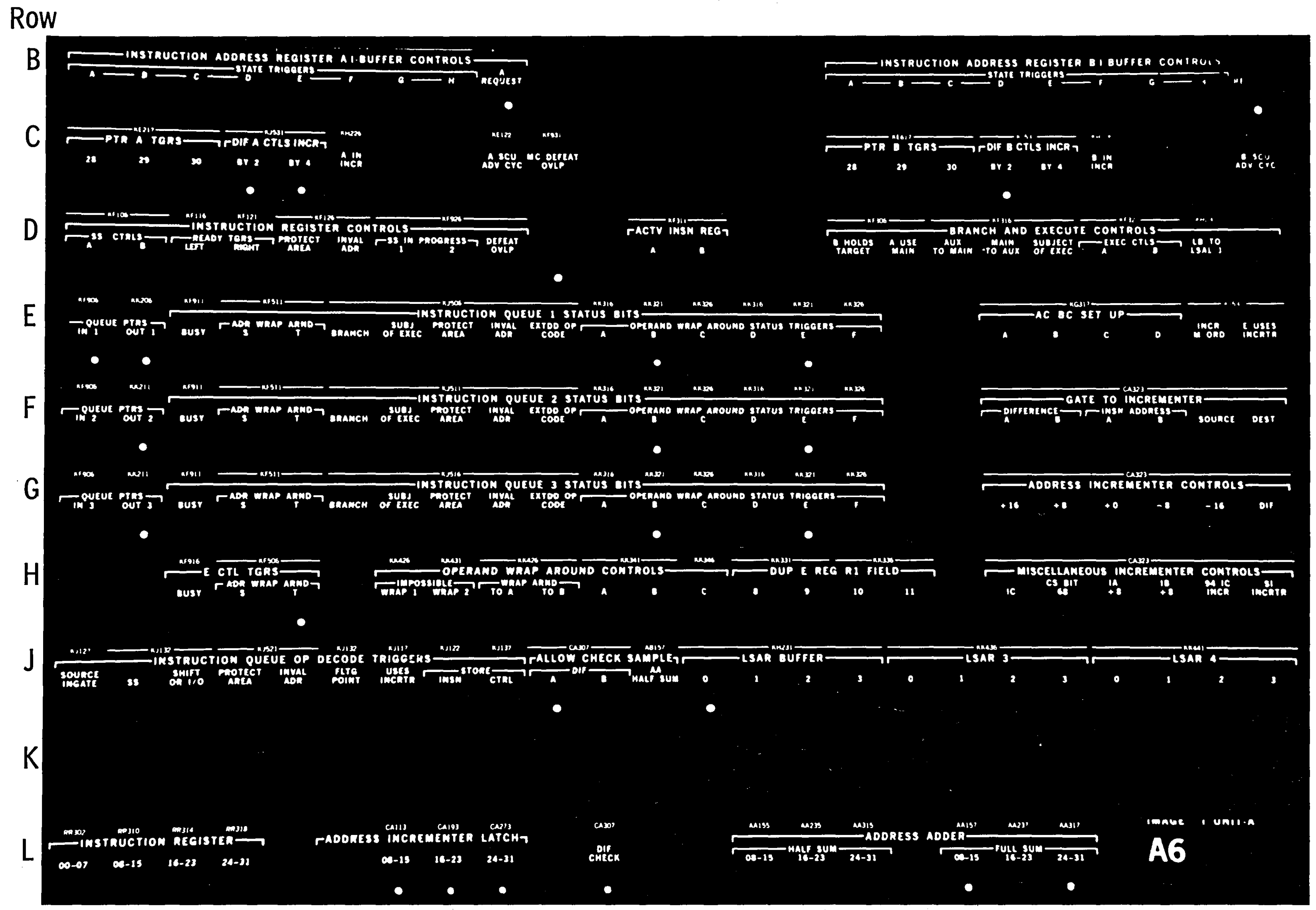


Figure 3-25 Indicator Display

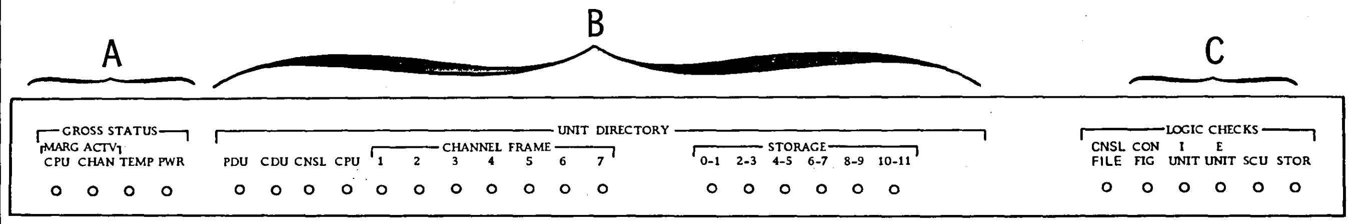


Figure 3-26 Row A Indicators

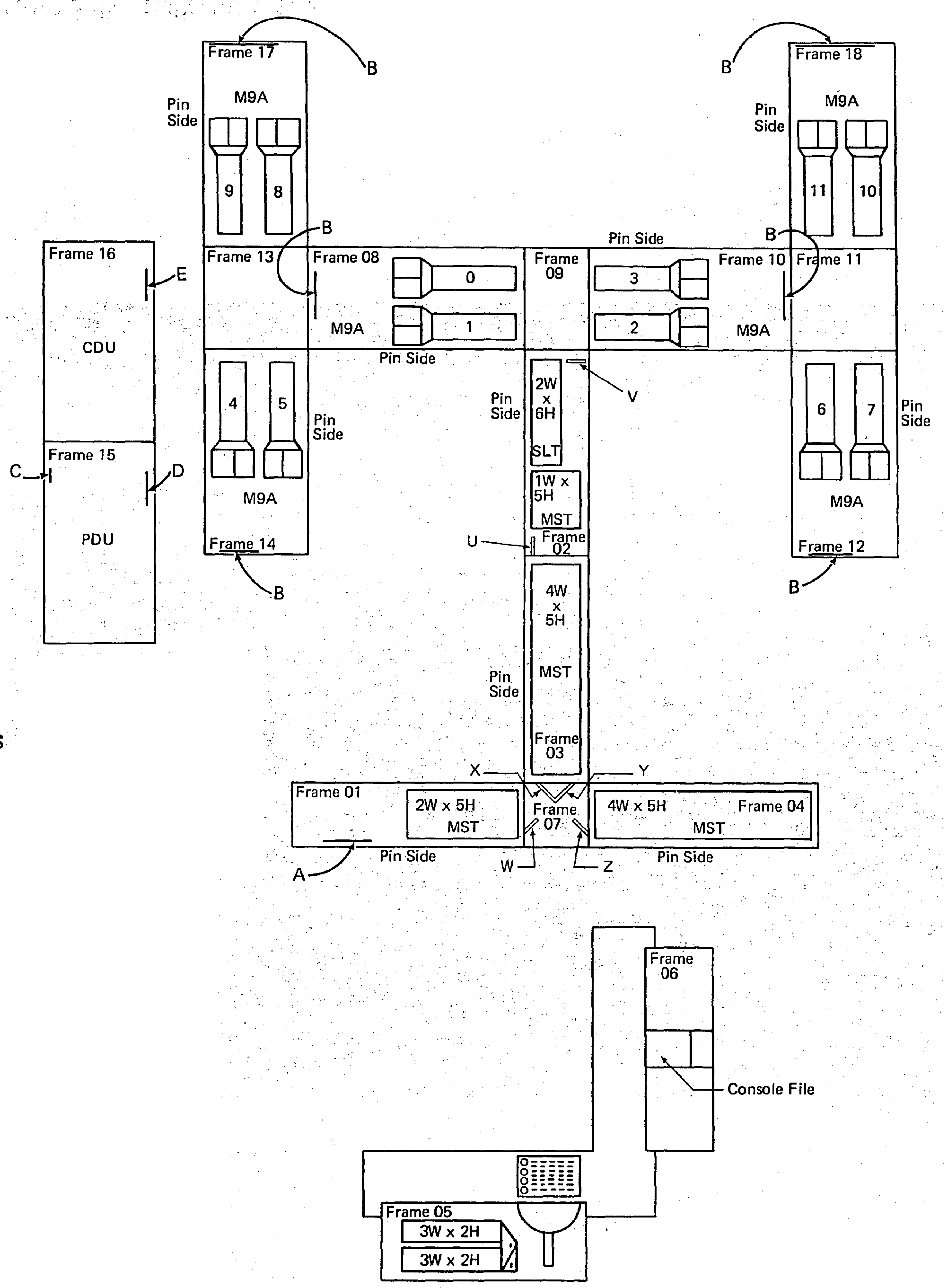


Figure 3-27 CE Panels

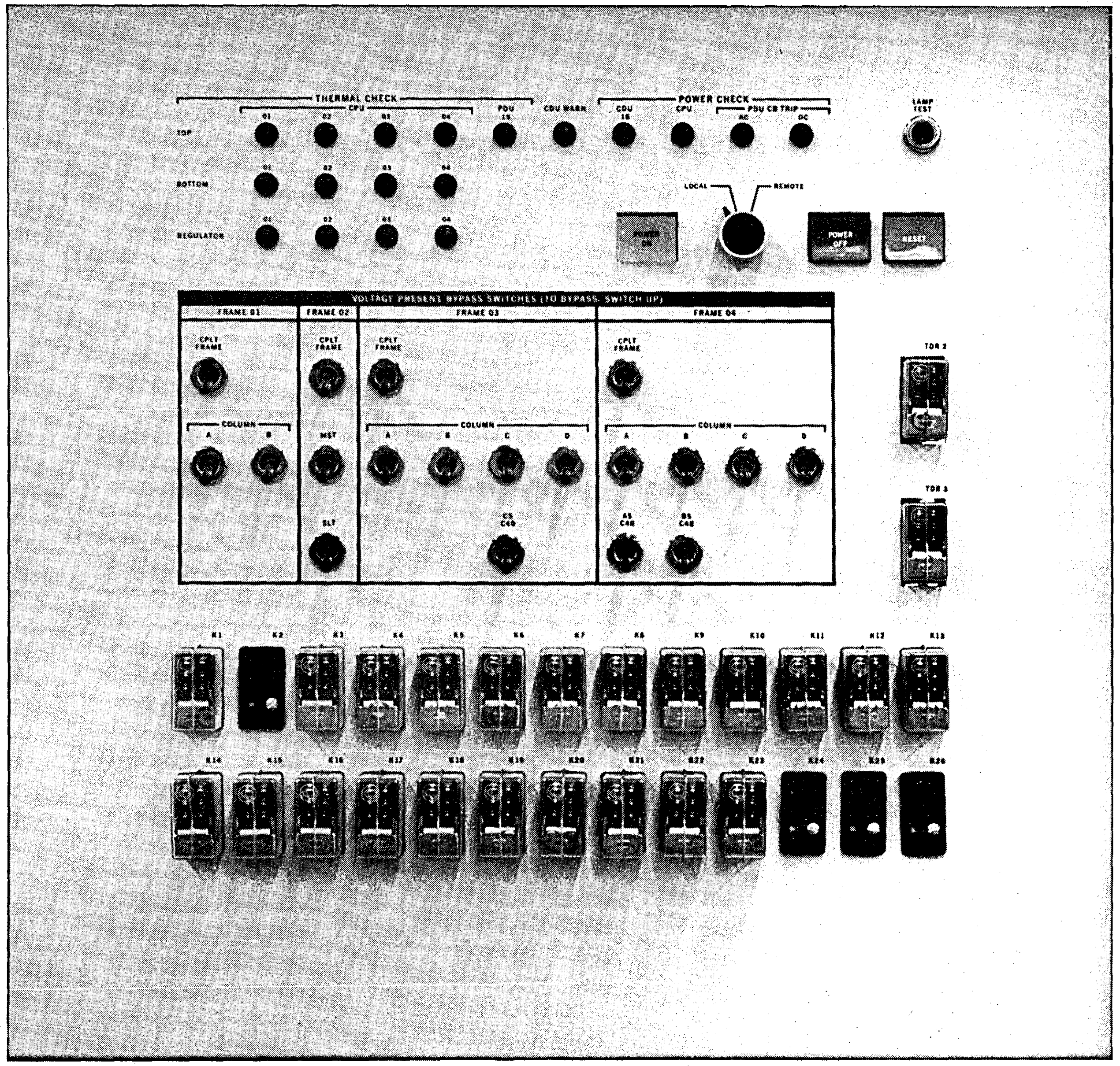


Figure 3-28 Frame 01 CE Panel

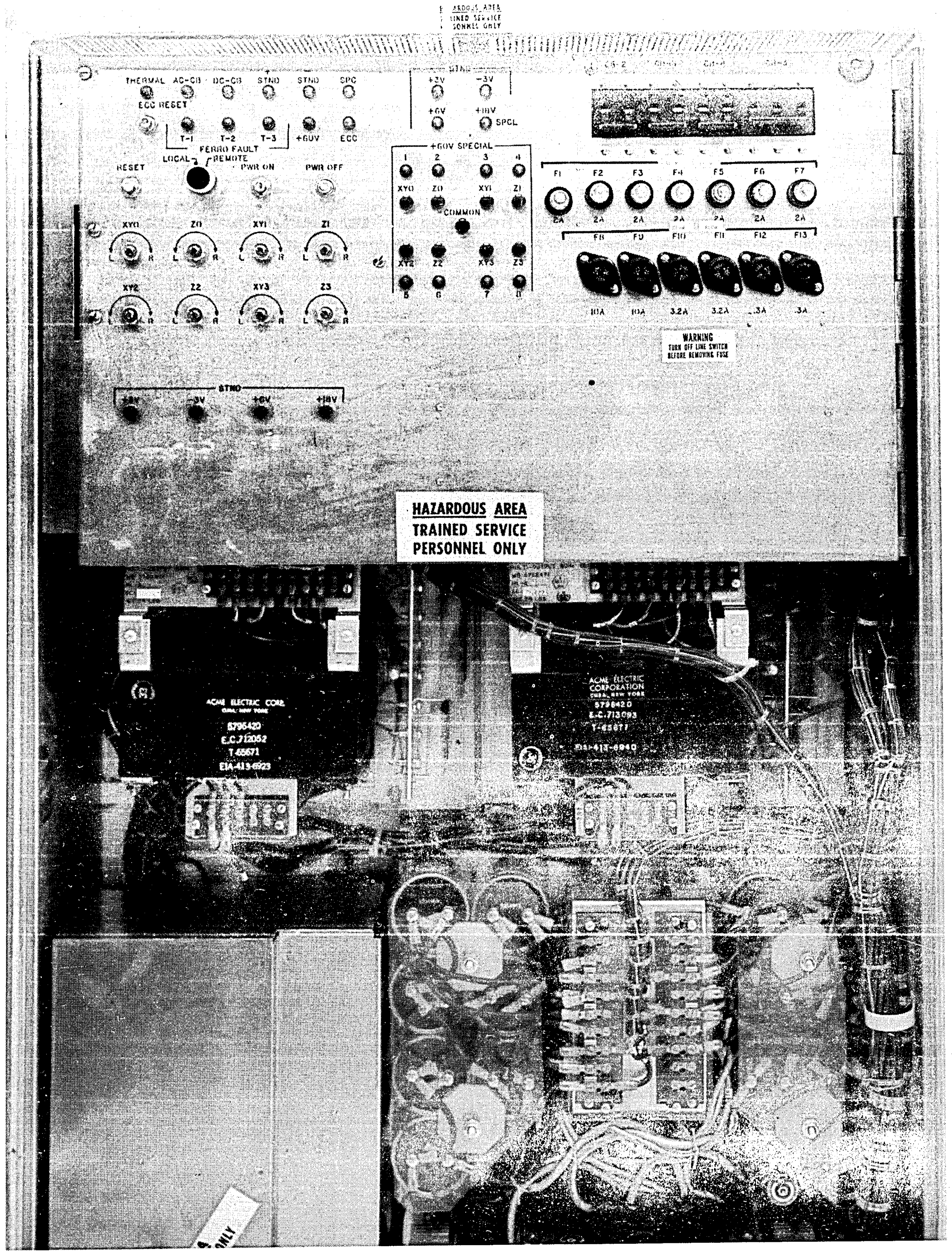


Figure 3-29 M9A CE Panel

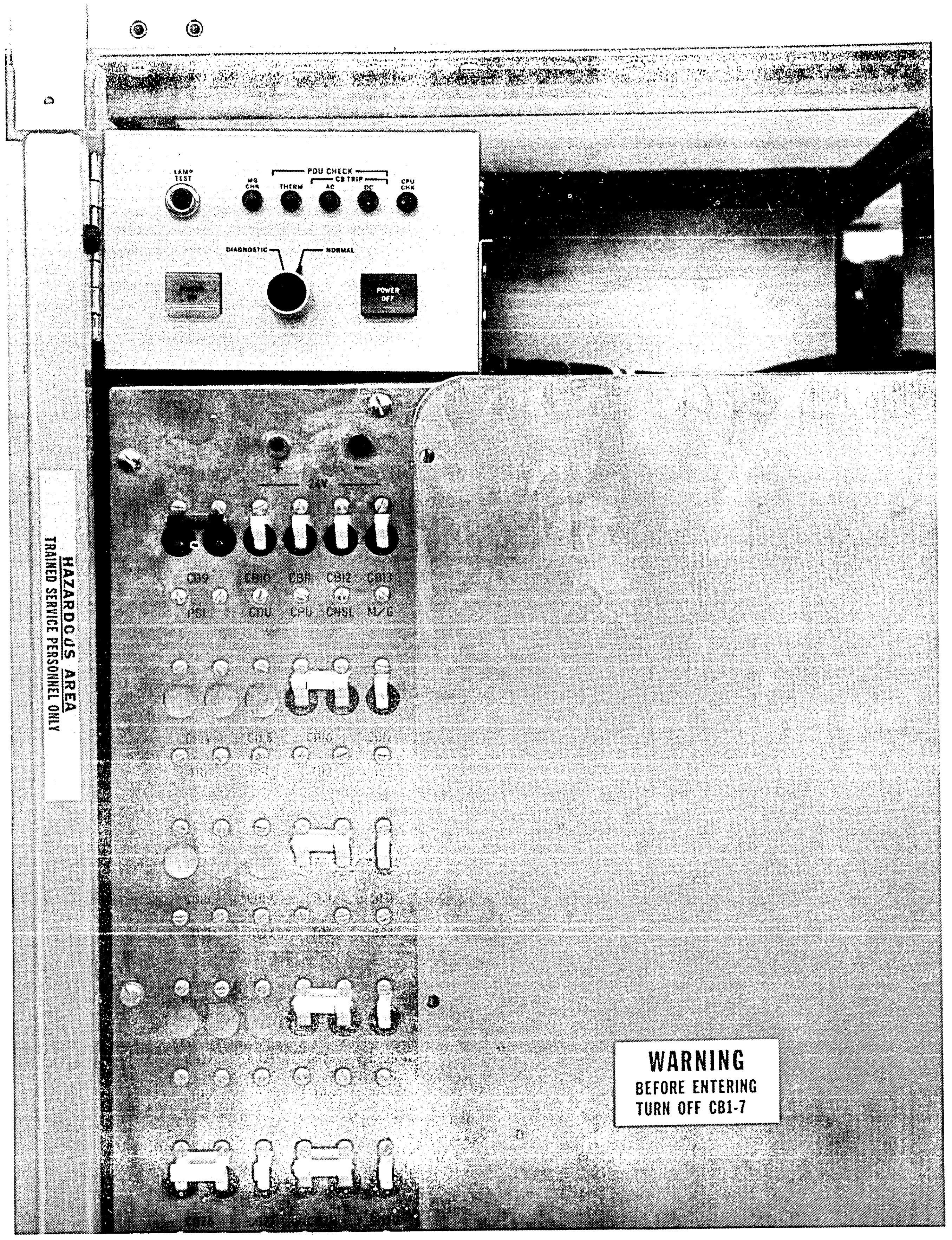


Figure 3-30 PDU CE Panel

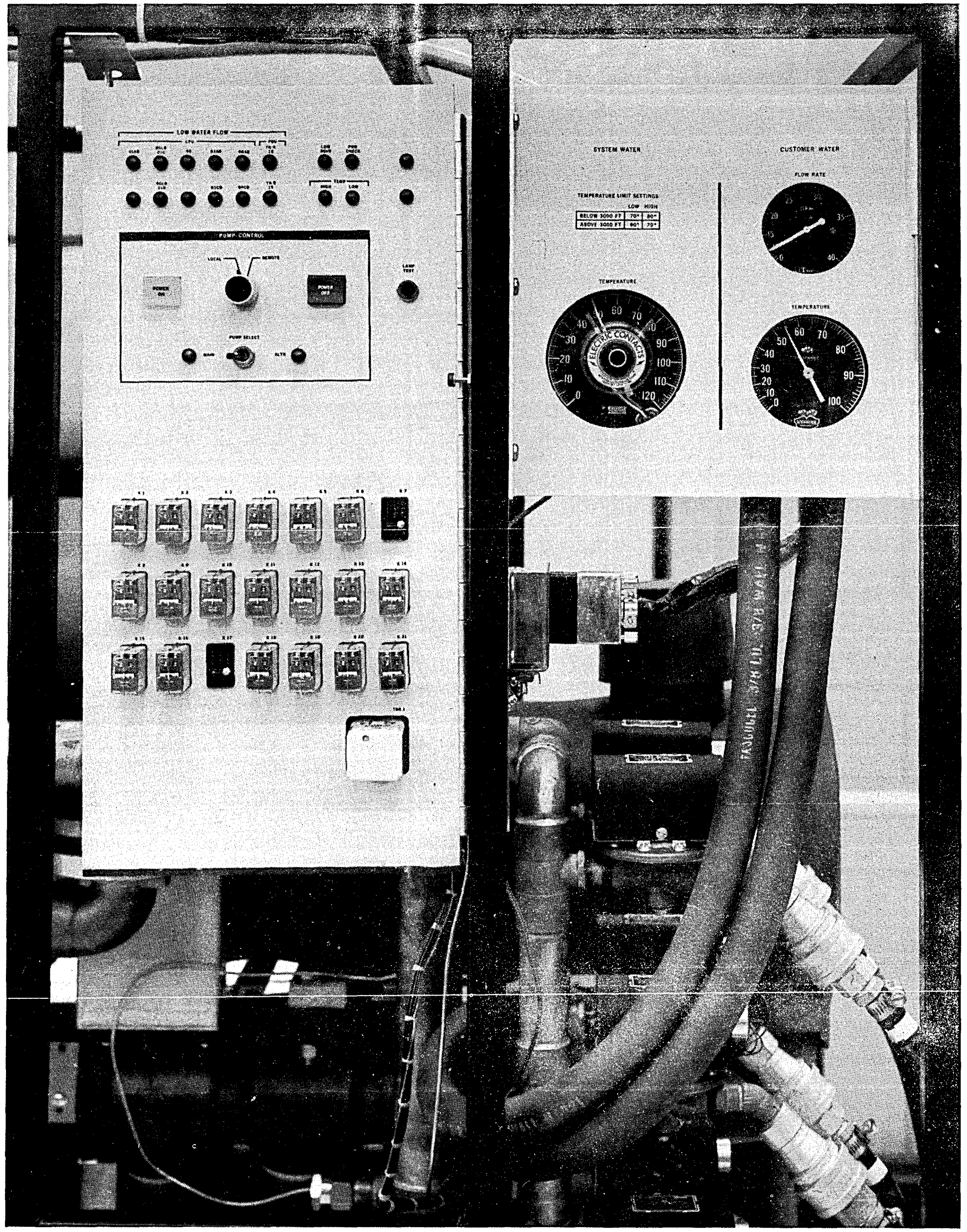


Figure 3-31 CDU CE Panel

10 FE Theory of Operation Manual (FETOM)
- Introduction and Timing
- I-Unit Operations
- E-Unit Operations, Volume 1
- E-Unit Operations, Volume 2
- SCU, SCLU, and Main Storage
- Features, 7094 Emulator
- Features, 7074 Emulator
- Features, 7080 Emulator
- Power and Cooling
- Console and Maintenance Facilities

6 FE Maintenance Diagrams - Manuals (FEMDMs)

*Volume 1	Base Machine - Diagnostic Diagrams
Volume 2	Base Machine - Clocks, I-Unit Operations, E-Unit Operations
Volume 3	Base Machine - Storage and Storage Control, Power and Cooling, Console
Volume 4	Features - 7094 Emulator
Volume 5	Features - 7074 Emulator
Volume 6	Features - 7080 Emulator

FE Maintenance Manual (FEMM)
FE Installation Manual (FEIM)
*Microdiagnostic Program User's Guide
FE Automated Logic Diagrams (FEALDs)
*DME User's Guide
*Model 165 Handbook

Figure 3-32 Manuals

Machine Check Interrupts (MCI)
Program Interrupt
Supervisor Call Interrupt
External Interrupt
Input/Output Interrupt

Figure 3-33 System/360 Interrupts

1. Hard Machine Check Interrupt
2. Program Interrupt
3. Supervisor Call Interrupt
4. Soft Machine Check Interrupt
5. External Interrupt
6. Input/Output

Figure 3-34 Model 165 Interrupt Priority

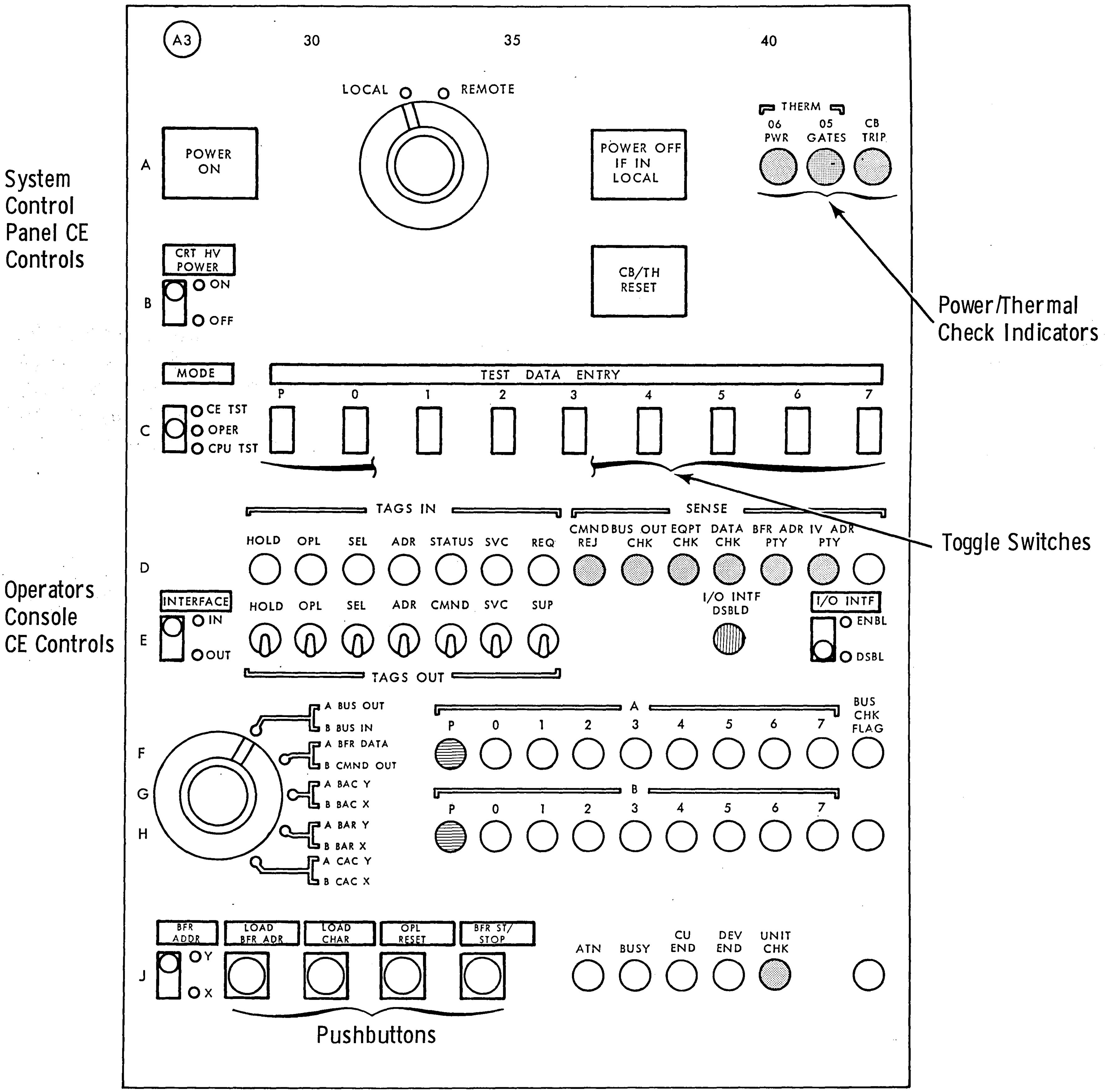


Figure 3-35 Section A3

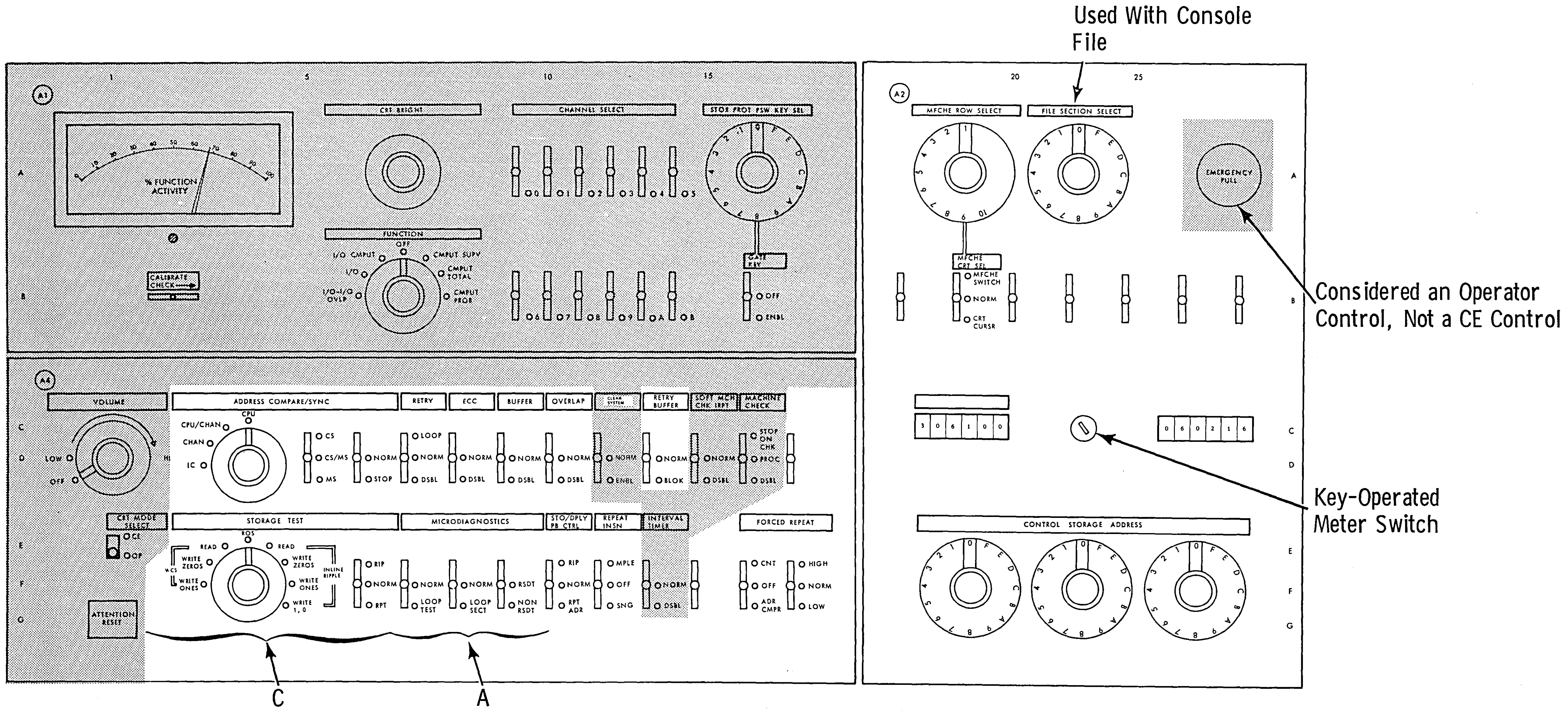


Figure 3-36 CE Controls

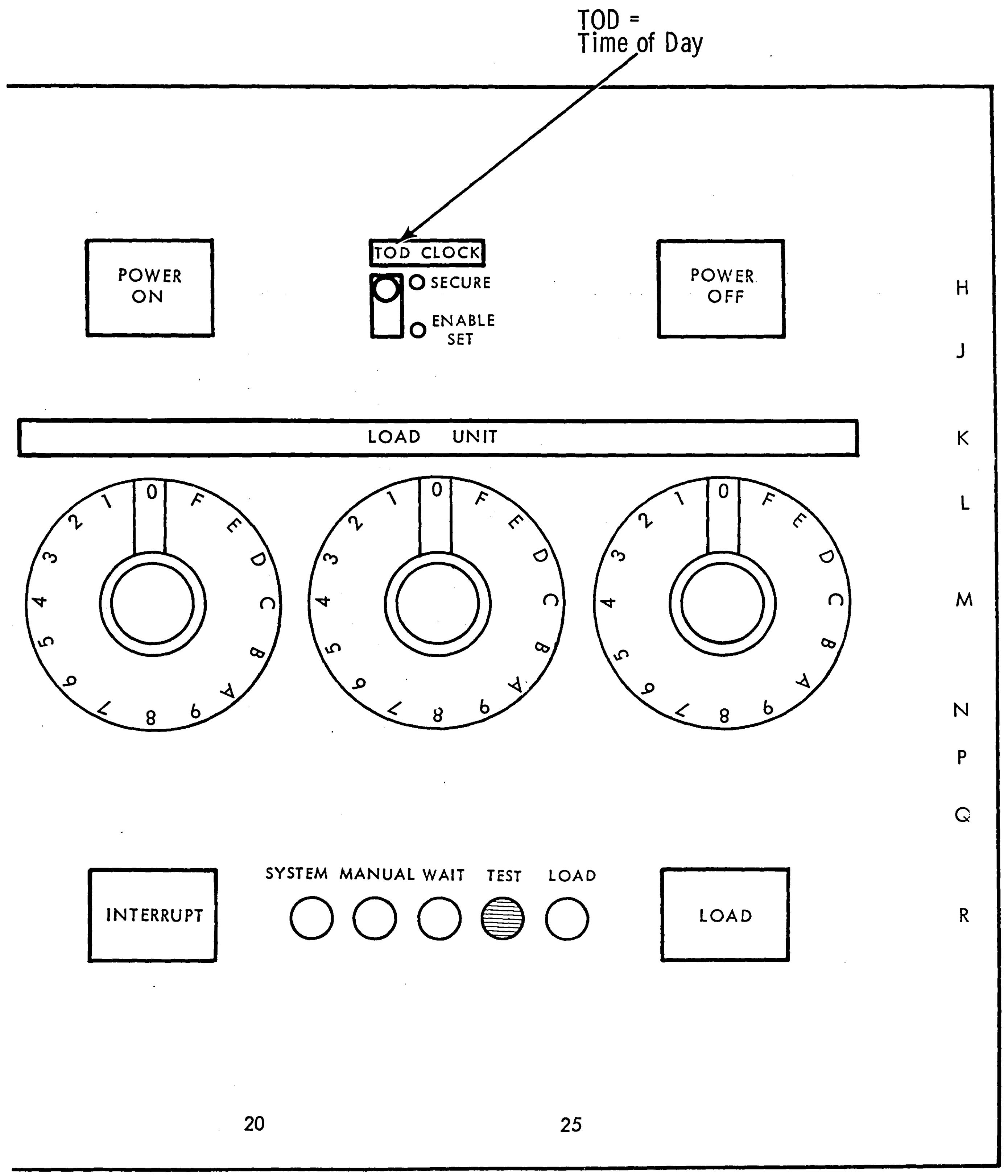


Figure 3-37 Operator Controls

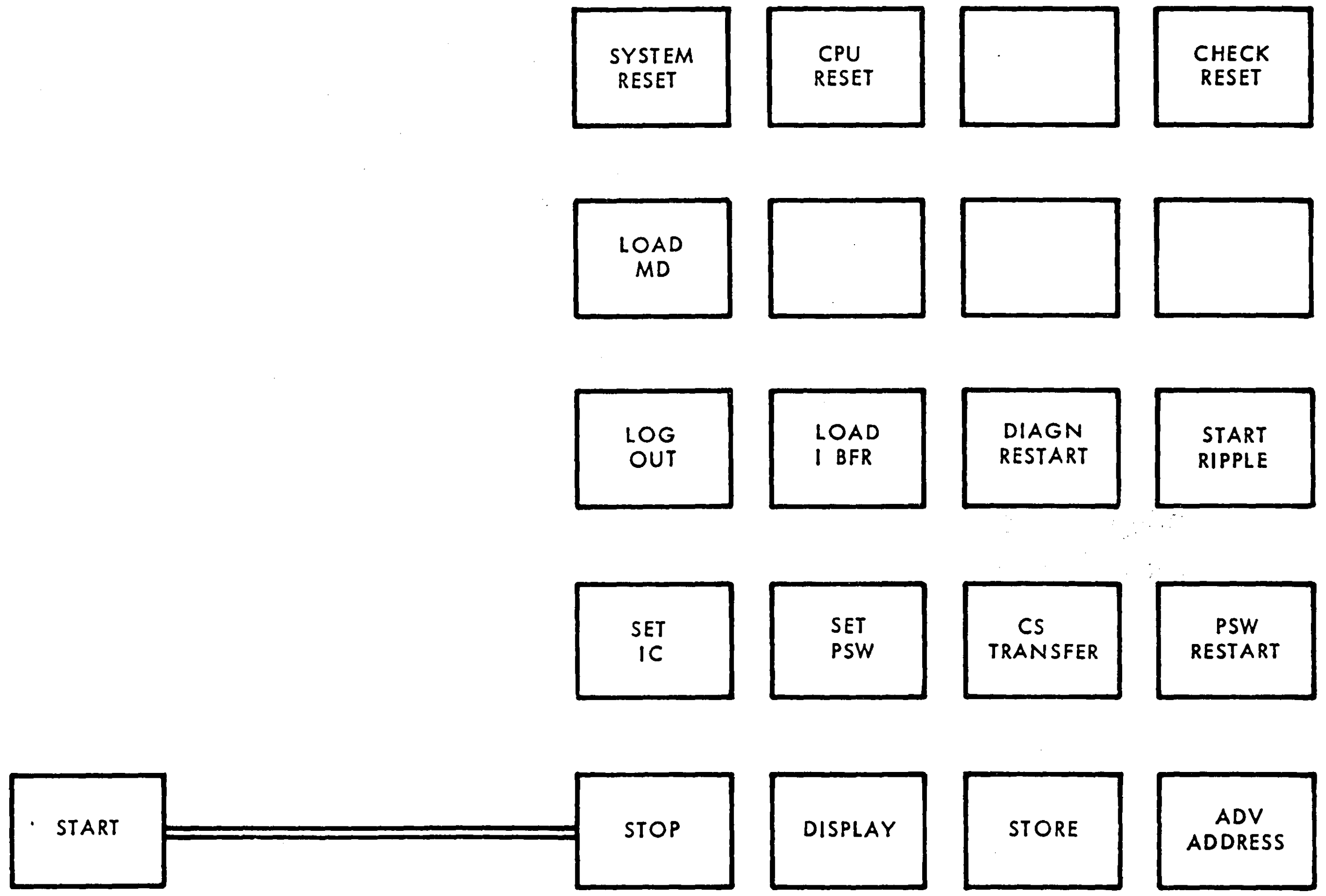


Figure 3-38 Pushbuttons

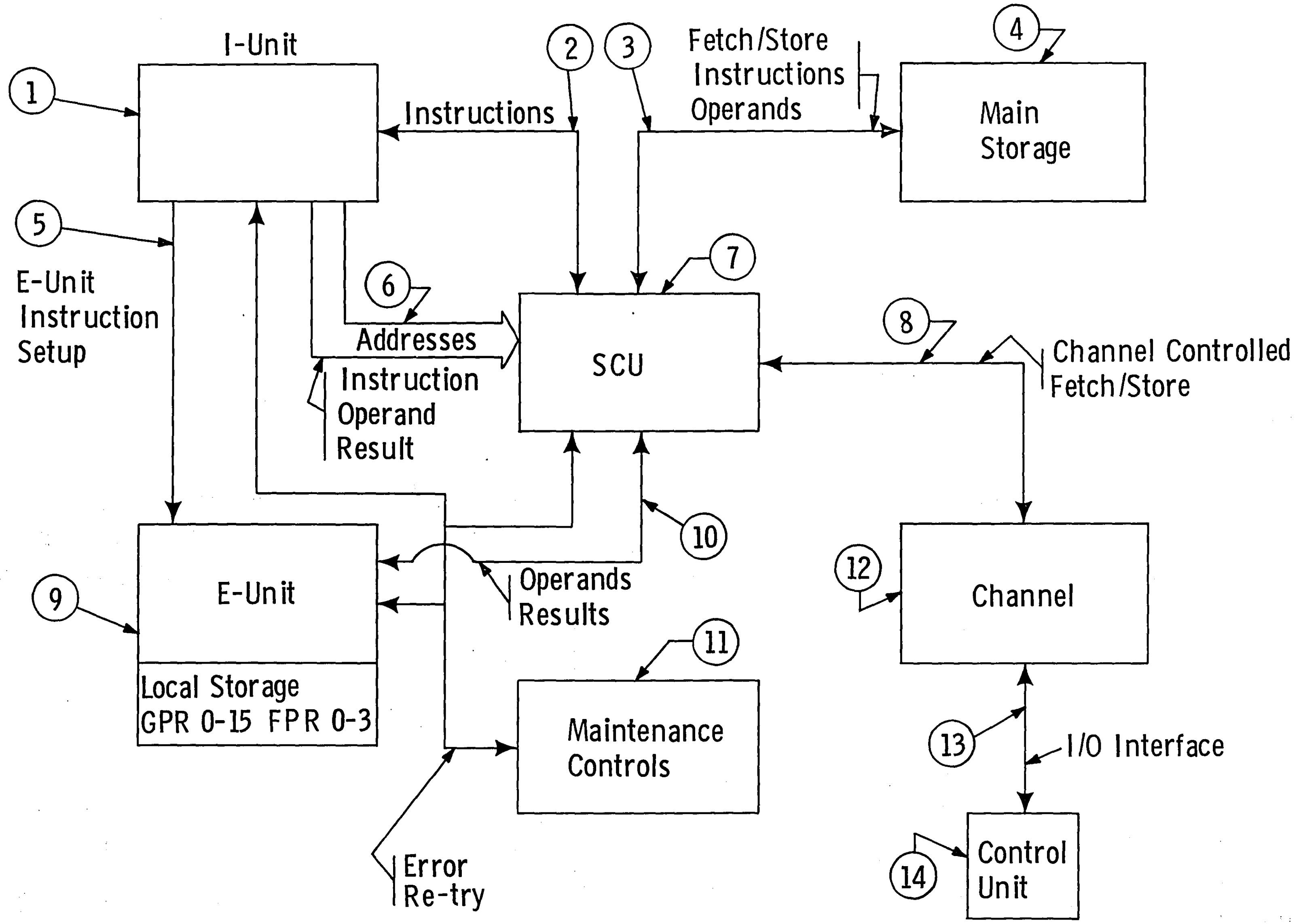


Figure 4-1 Questions 9-12

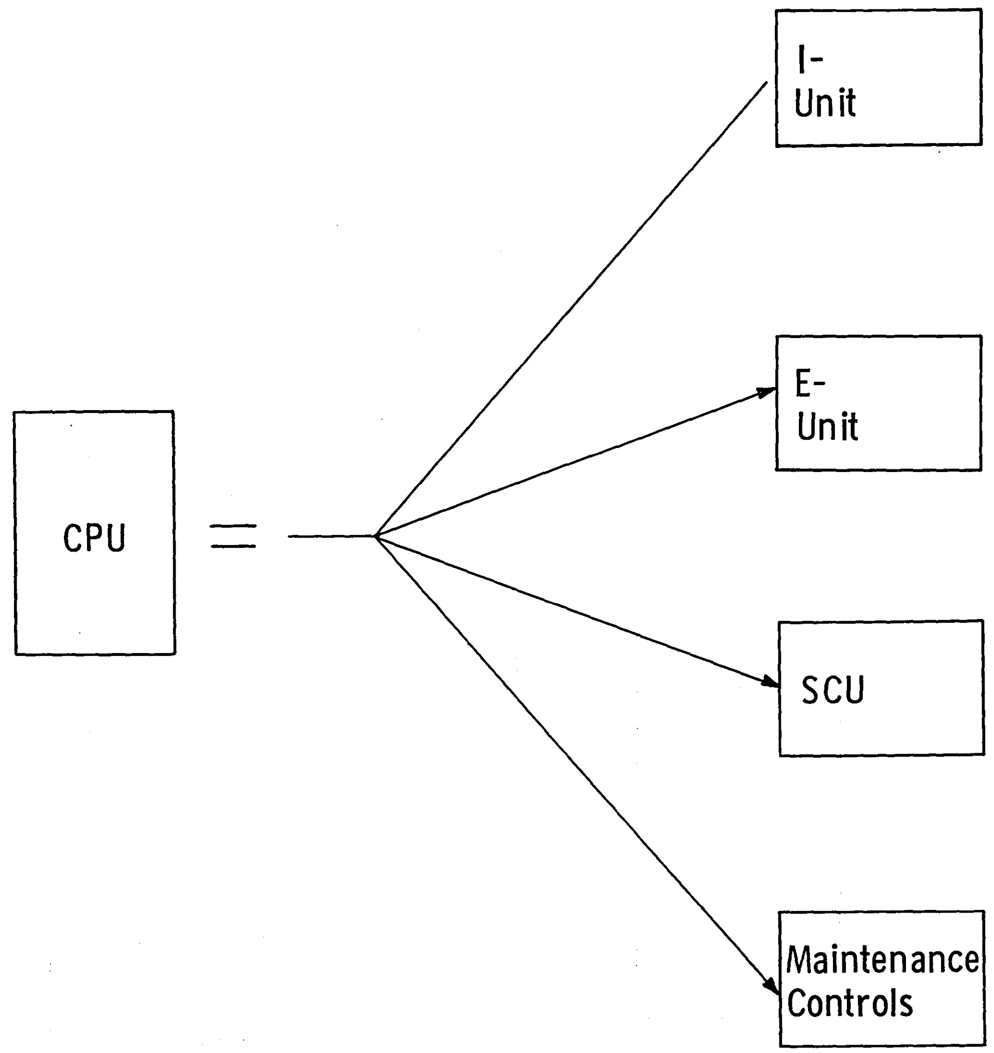
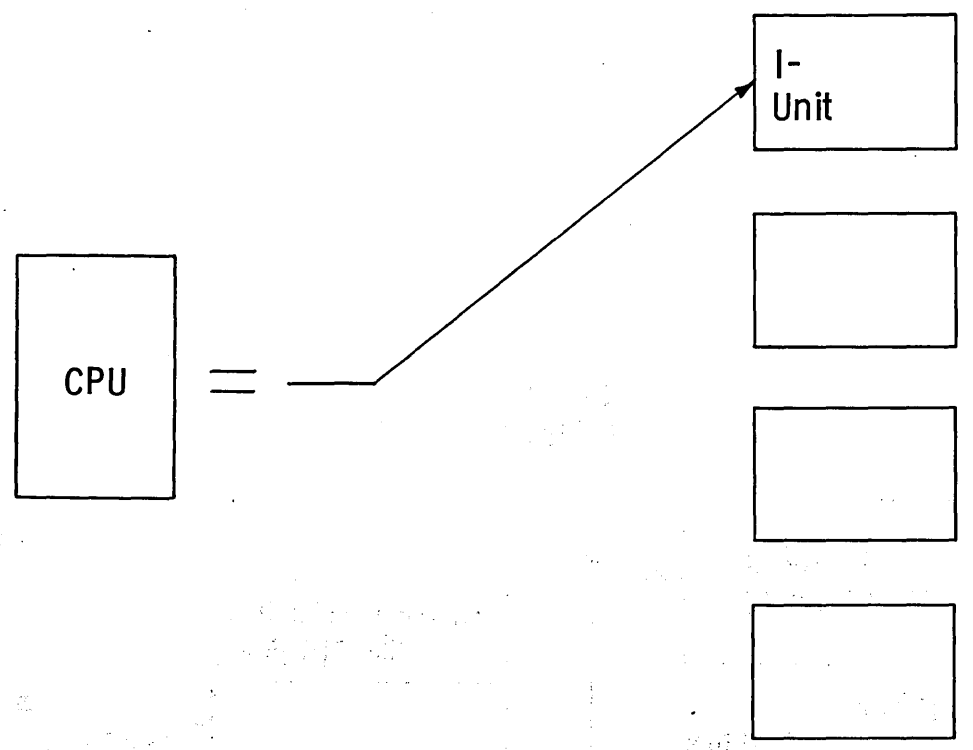


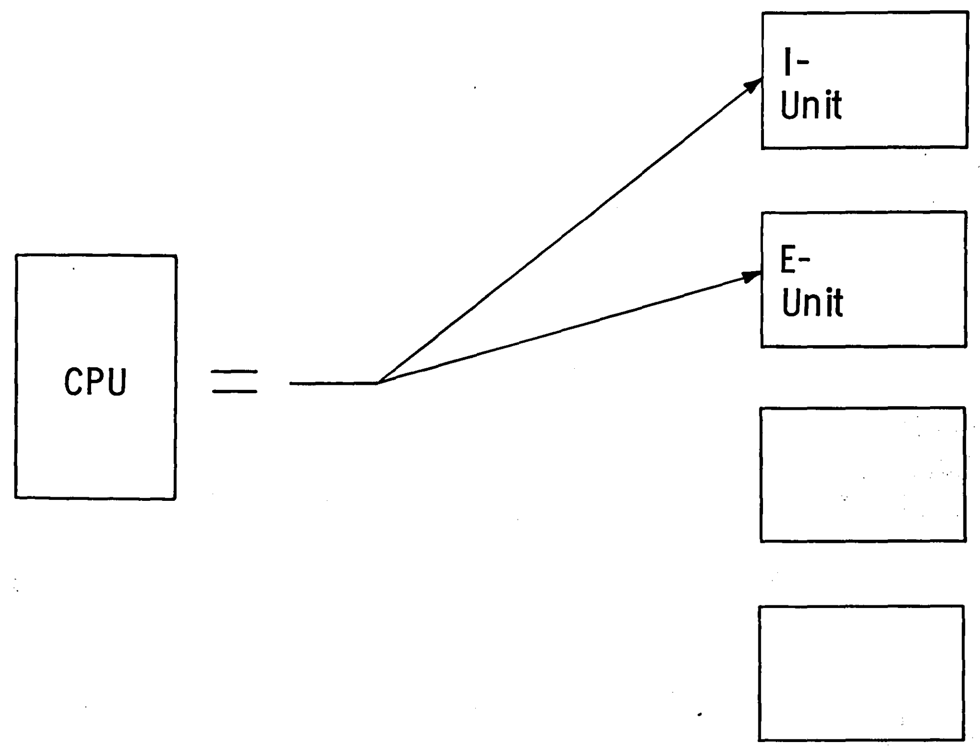
Figure 4-2 CPU Operational Units



Instruction Unit

* Fetch Instructions
  Requests "blocks" of instructions from storage via the storage control unit.
* Buffer Instructions
  Has buffers/registers for the temporary storage of instructions both as they enter and after they are decoded and are ready to be transferred to the E-unit.
* Decodes Instructions
  Instruction is identified and the availability of the facilities needed for further processing is checked.
* Calculates effective storage addresses
  A three input adder for determining address for branches, stores, fetches, I/O operations, and shift instructions.
* Fetches Operands
  Operands (data the program is operating upon) to be used by the E-unit are prefetched from storage by the I-unit, and loaded into the buffers in the E-unit.
* E-Unit Setup
  Establishes the initial conditions for instruction execution.

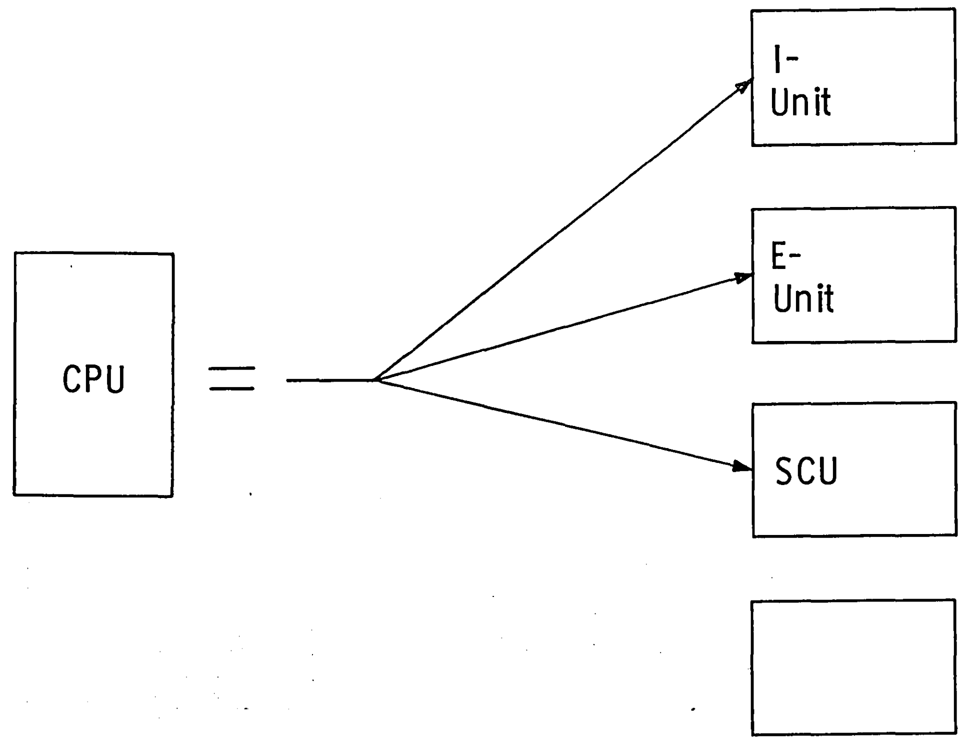
Figure 4-3 I-Unit Operations



Execution Unit

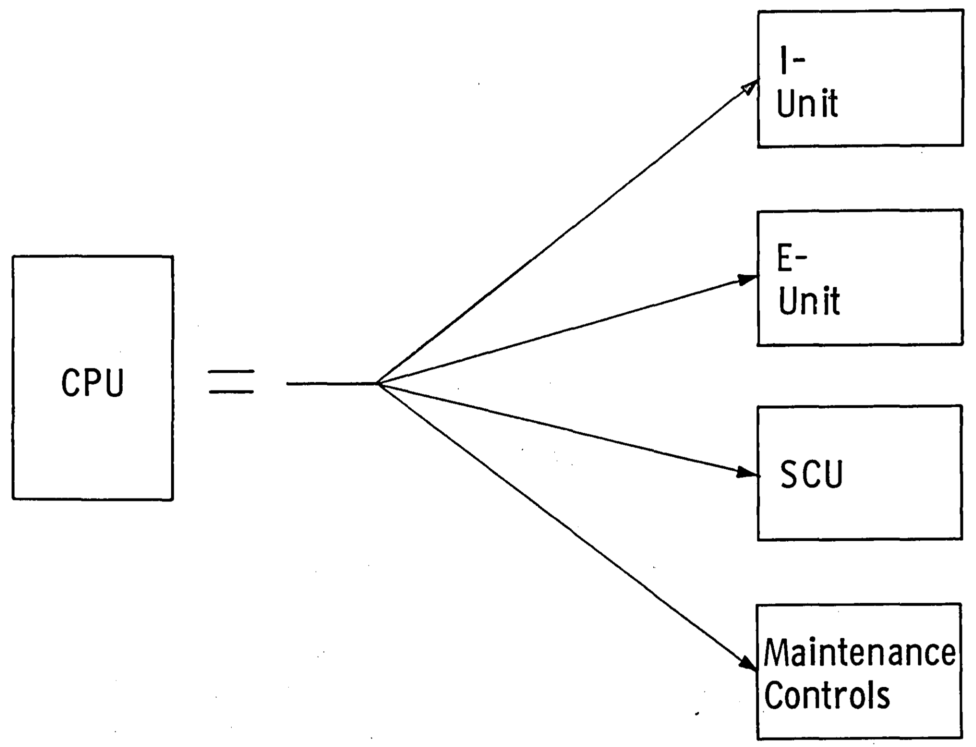
* Processes the instructions fetched and readied by the I-unit.
* Except for the portions needed by the I-unit to set up initial conditions, the E-unit is controlled by a microprogram.
* Local storage, consisting of 20-latch registers and associated address registers, are located in the E-unit.
  a. The 20 latch registers are used for the 16 GPRs and four FPRs that make up local storage.
  b. Having the GPRs and FPRs in the E-unit reduces the number of processor storage references required by the CPU during each operation.

Figure 4-4 E-Unit Operations



Storage Control Unit
*Functionally connects all system units to processor storage.
*Controls all references to storage from the CPU (I-and E-units), channels, and the maintenance controls.
a. Resolves priority between requests.
b. Checks the validity of each request.
*Controls the operation of the high-speed buffer located in the SCU.
*Contains the Error Checking and Correction (ECC) logic.

Figure 4-5 SCU Operations



Maintenance Controls
* Executes the functions provided by the system control panel controls, provides diagnostic capabilities for the system, and contains logic required to buffer the needed information to execute the system portion of instruction re-try.

Figure 4-6 Maintenance Control Operation

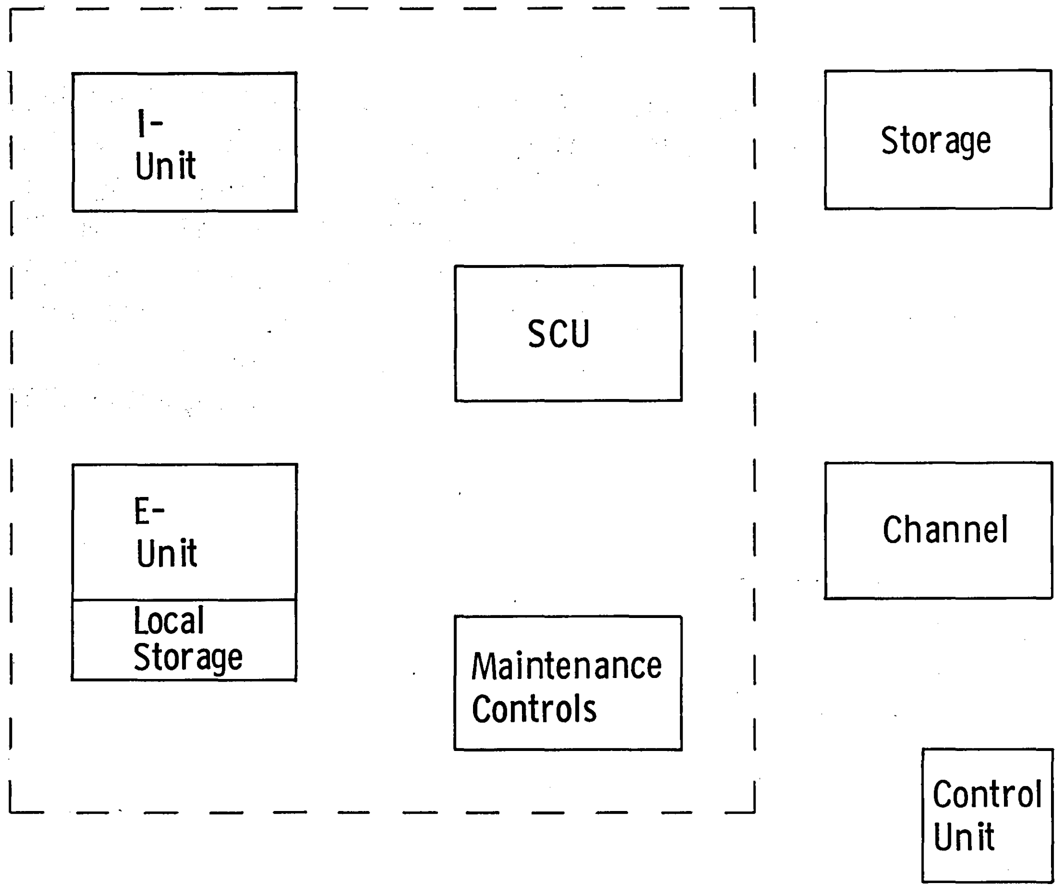


Figure 4-7 Basic Logic Flow

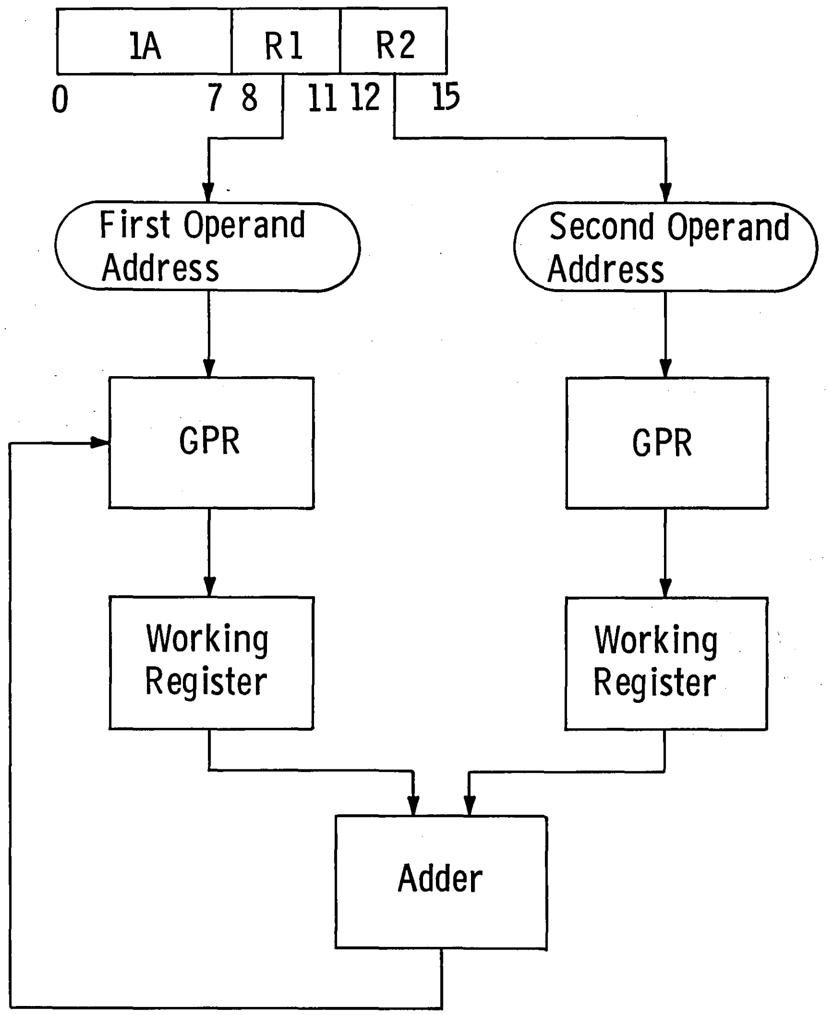


Figure 4-8 RR Instruction

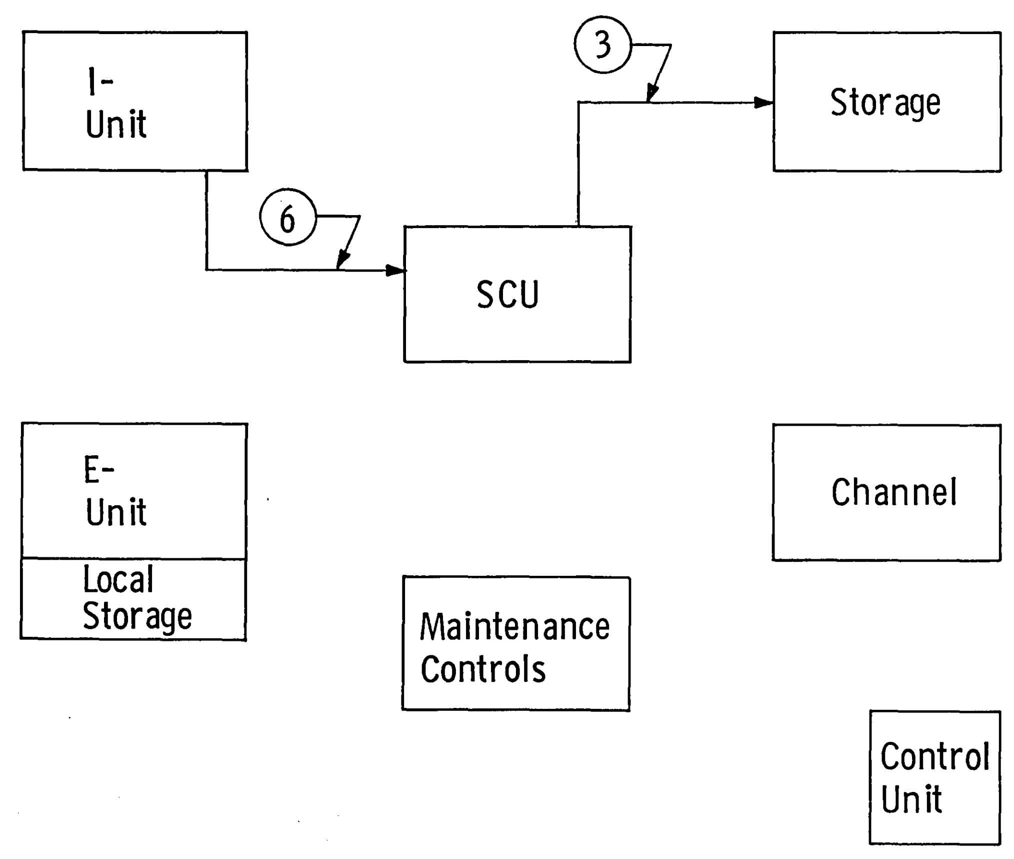


Figure 4-9 RR Basic Logic Flow 1

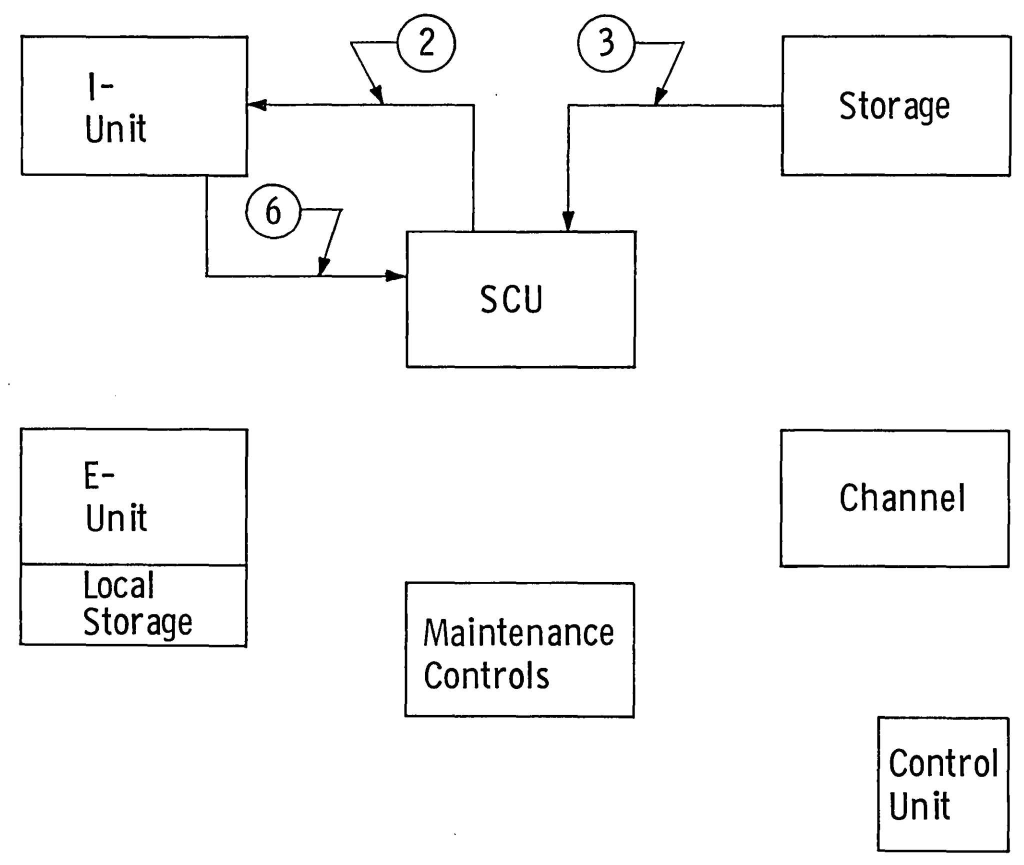


Figure 4-10 RR Basic Logic Flow 2

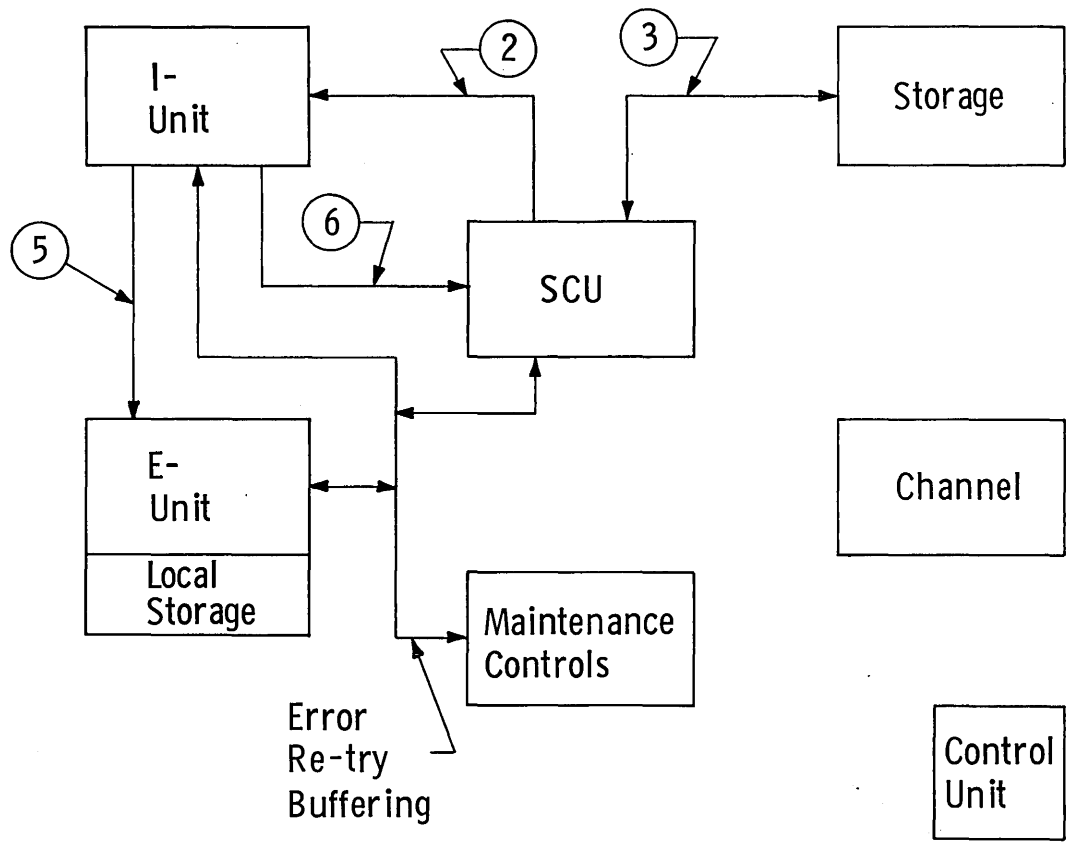


Figure 4-11 RR Basic Logic Flow 3

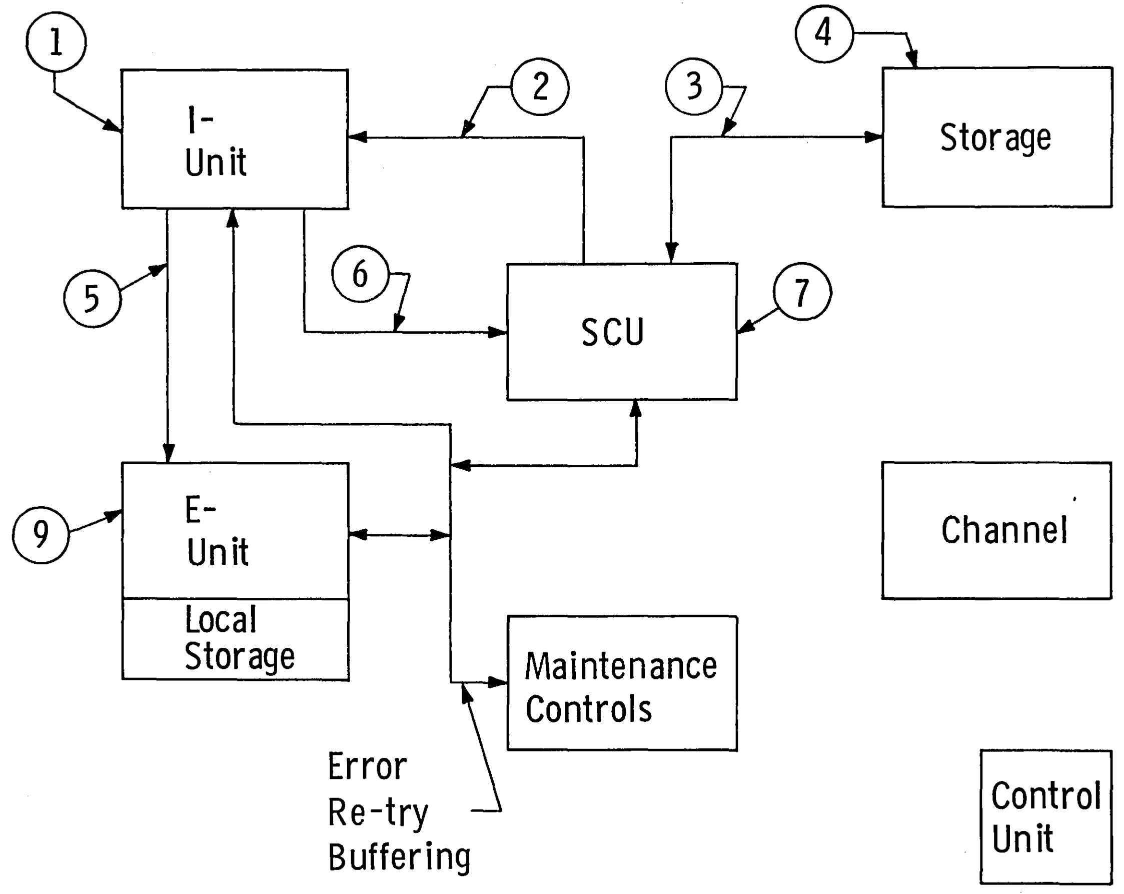


Figure 4-12 RR Question on Basic Logic Flow

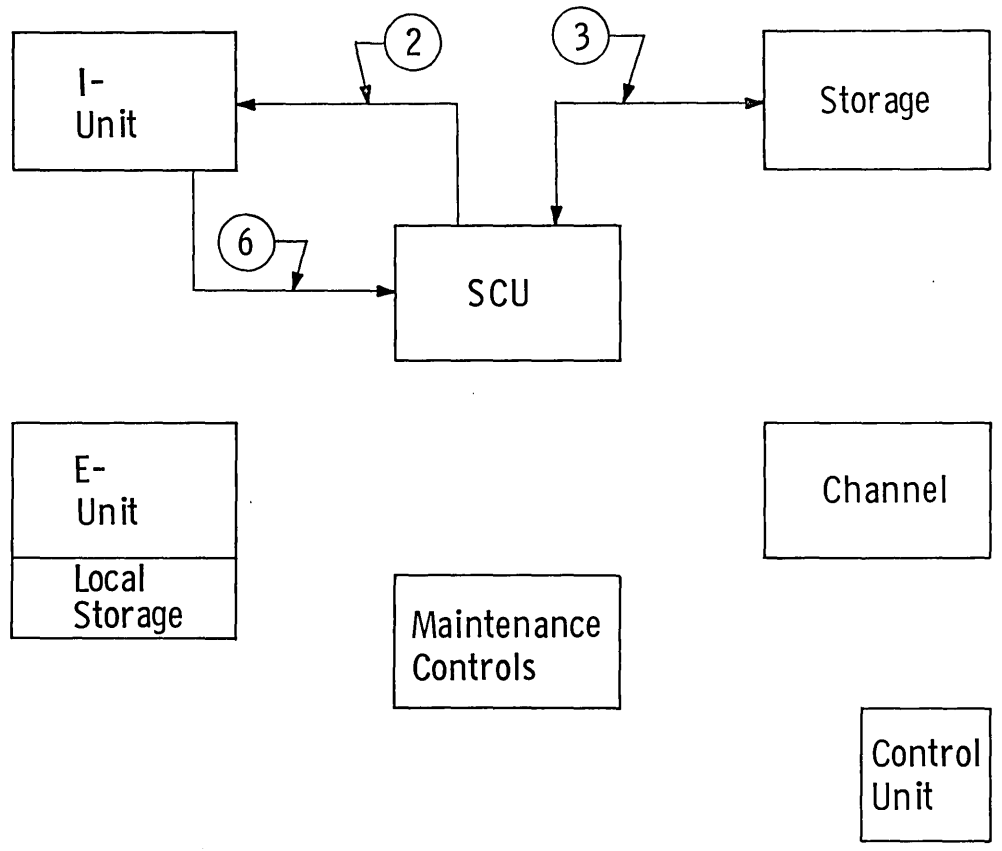


Figure 4-13 RX Basic Logic Flow 1

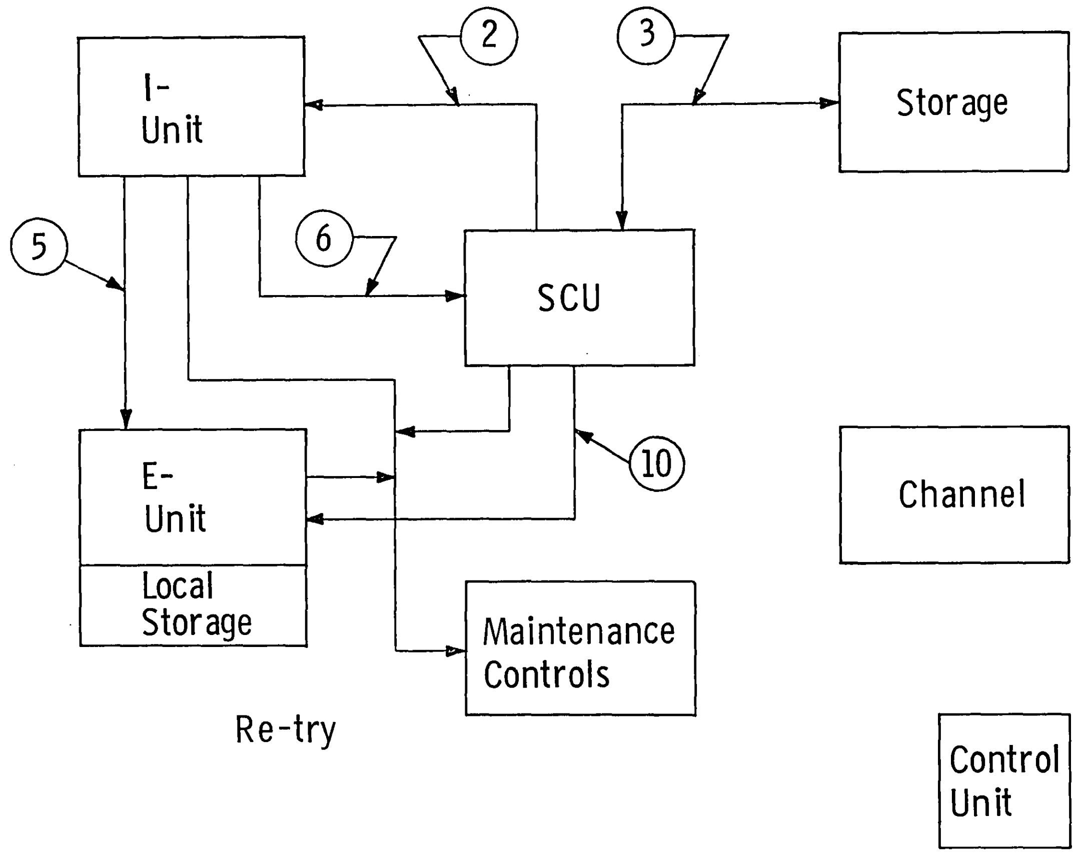


Figure 4-14 RX Basic Logic Flow 2

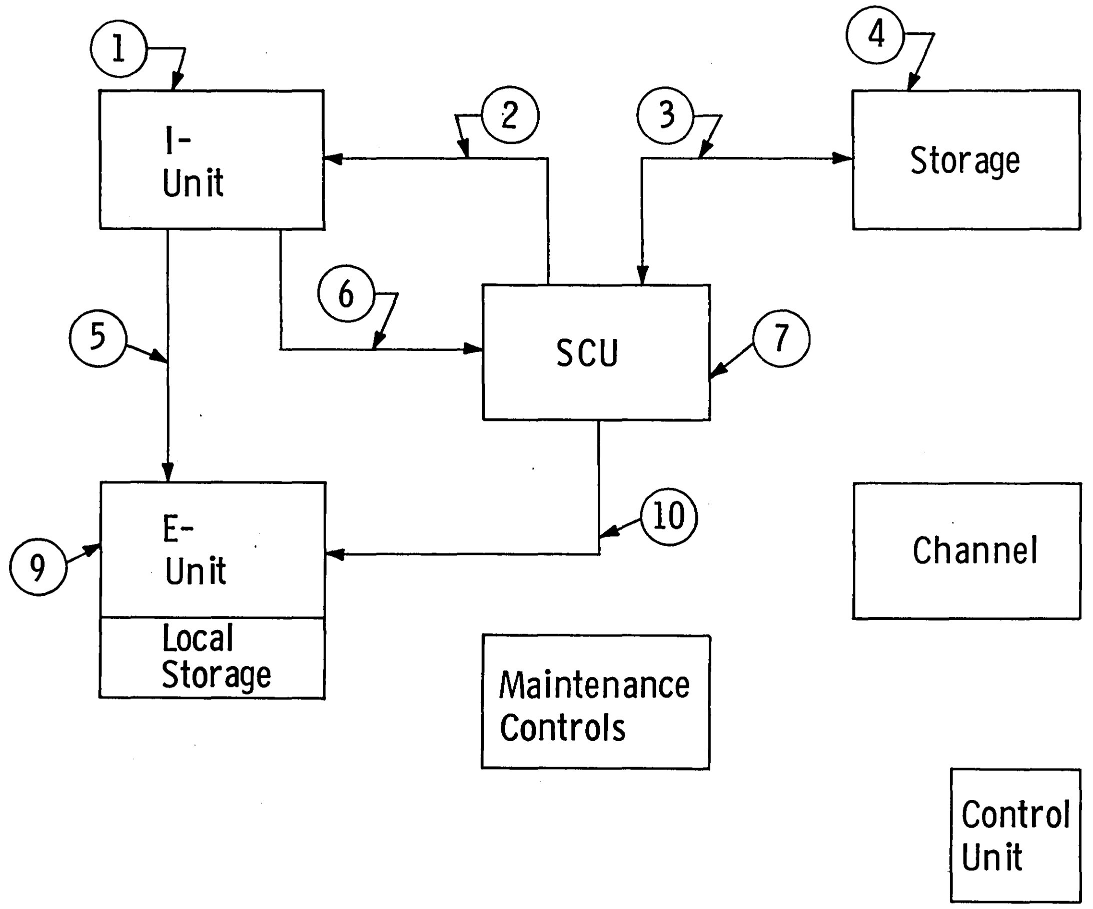


Figure 4-15 RX Question on Basic Logic Flow

Op Code	R1	R2

1A	Add
1E	Add Logical
19	Compare
15	Compare Logical
1B	Subtract
1F	Subtract Logical

Op Code	R1	X2	B2	D2

5A	Add
4A	Add Halfword
5E	Add Logical
59	Compare
49	Compare Halfword
55	Compare Logical
5B	Subtract
4B	Subtract Halfword
5F	Subtract Logical

Figure 4-16 RR-RX Instructions

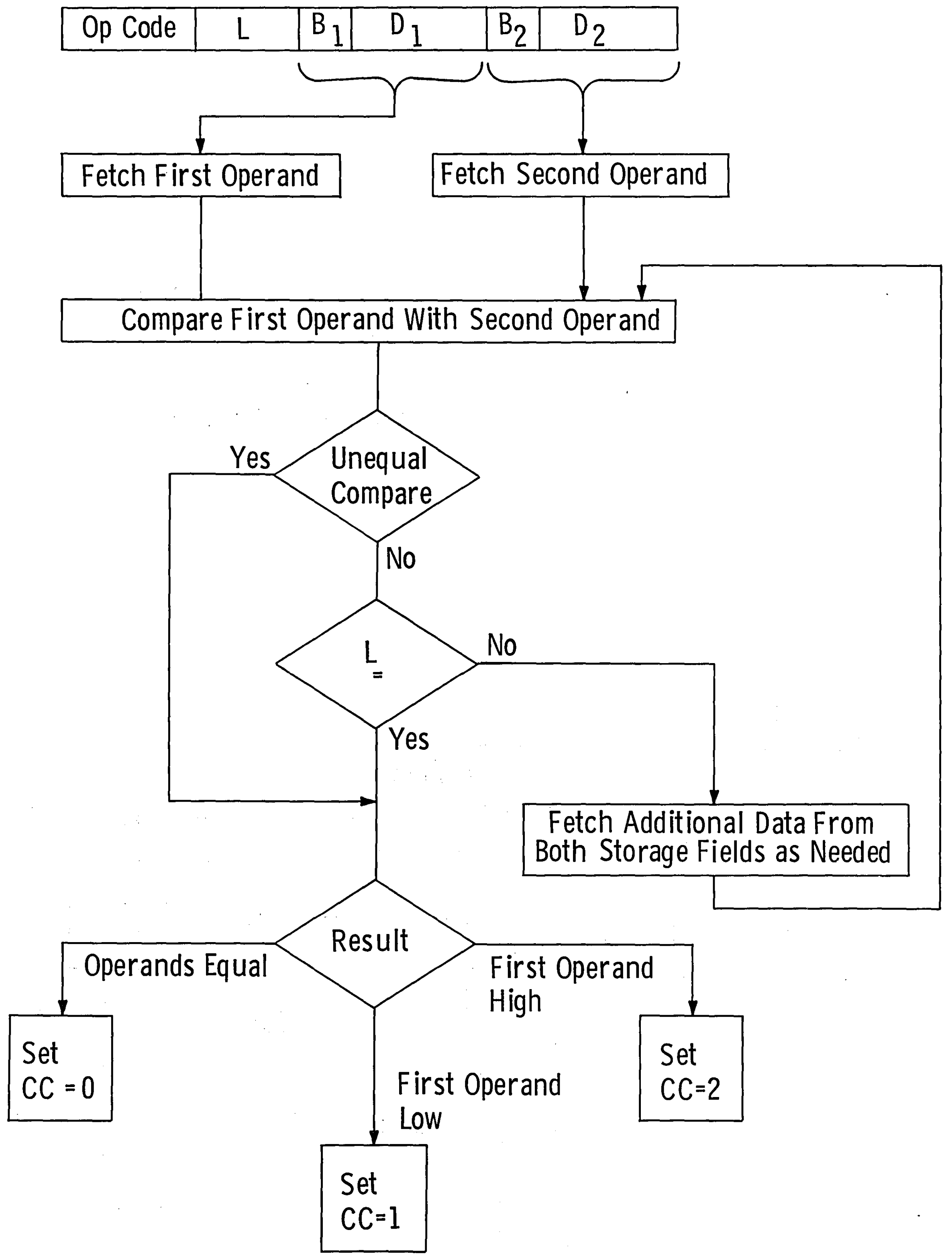


Figure 4-17 SS Compare Flowchart

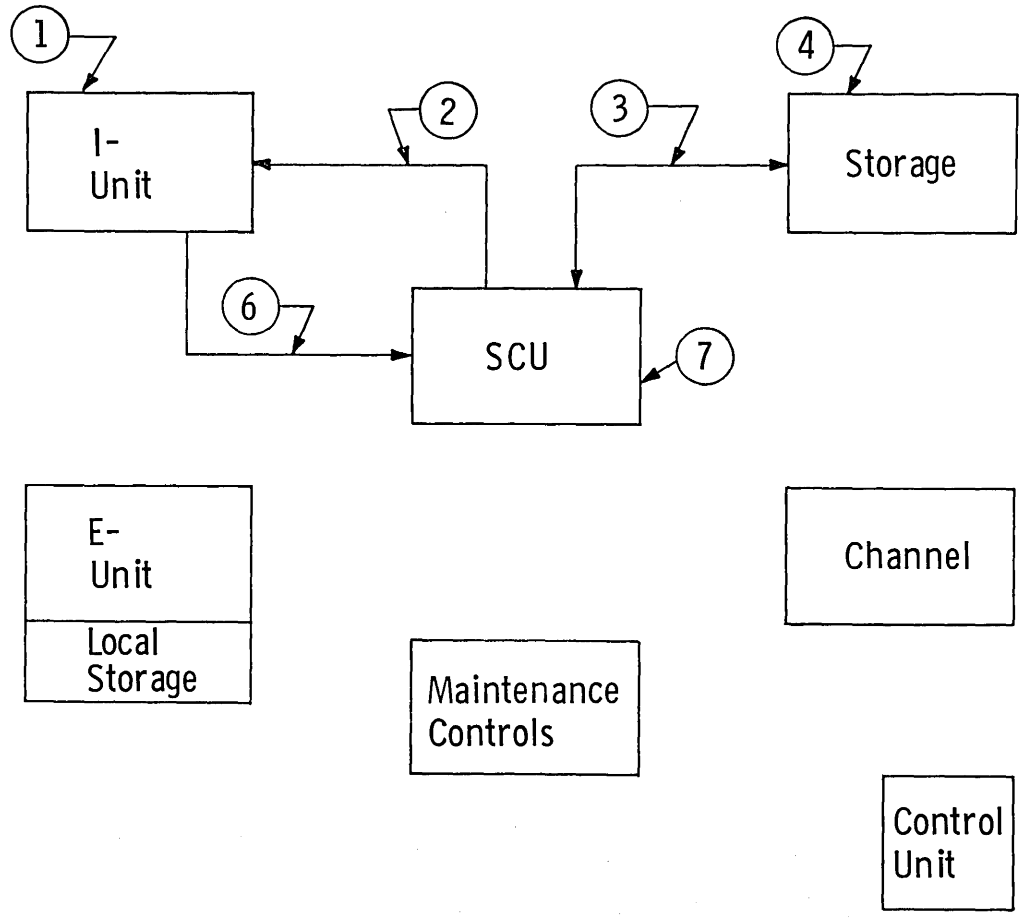
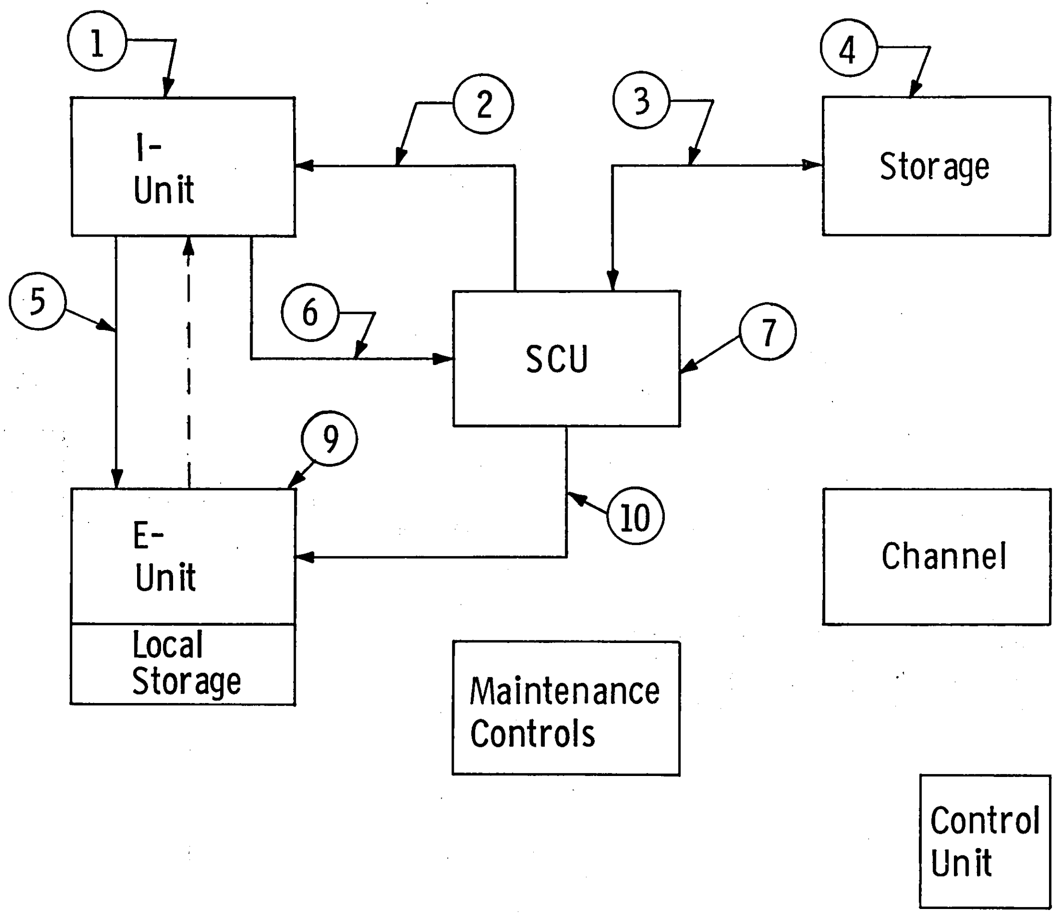


Figure 4-18 SS Basic Logic Flow 1



----- E-Unit Control of I-Unit For Operand Fetches

Note: Re-try Paths Omitted for Clarity

Figure 4-19 SS Basic Logic Flow 2

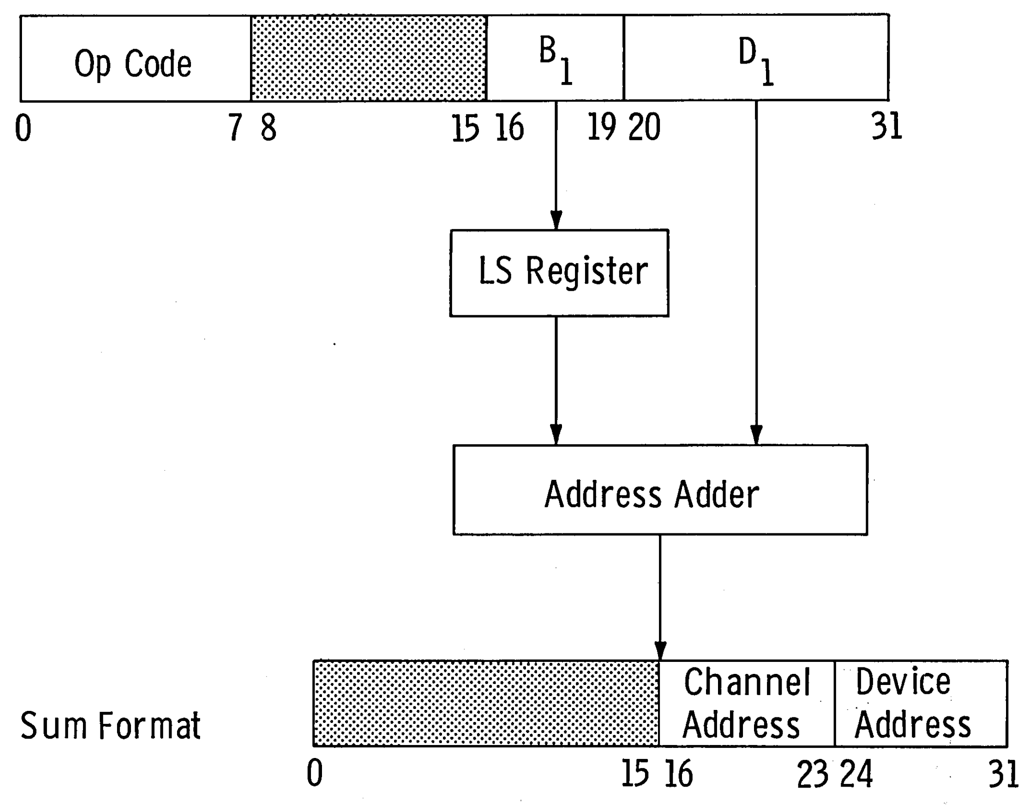


Figure 4-20 I/O Instructions

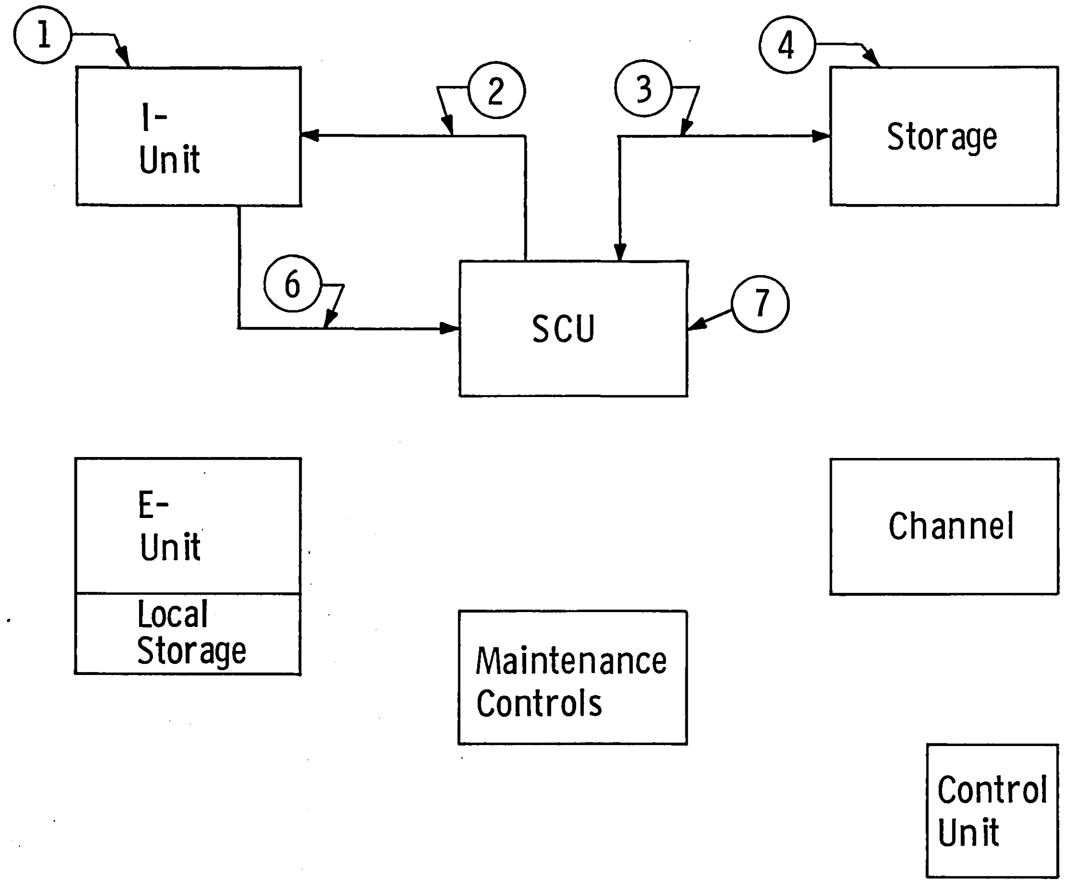
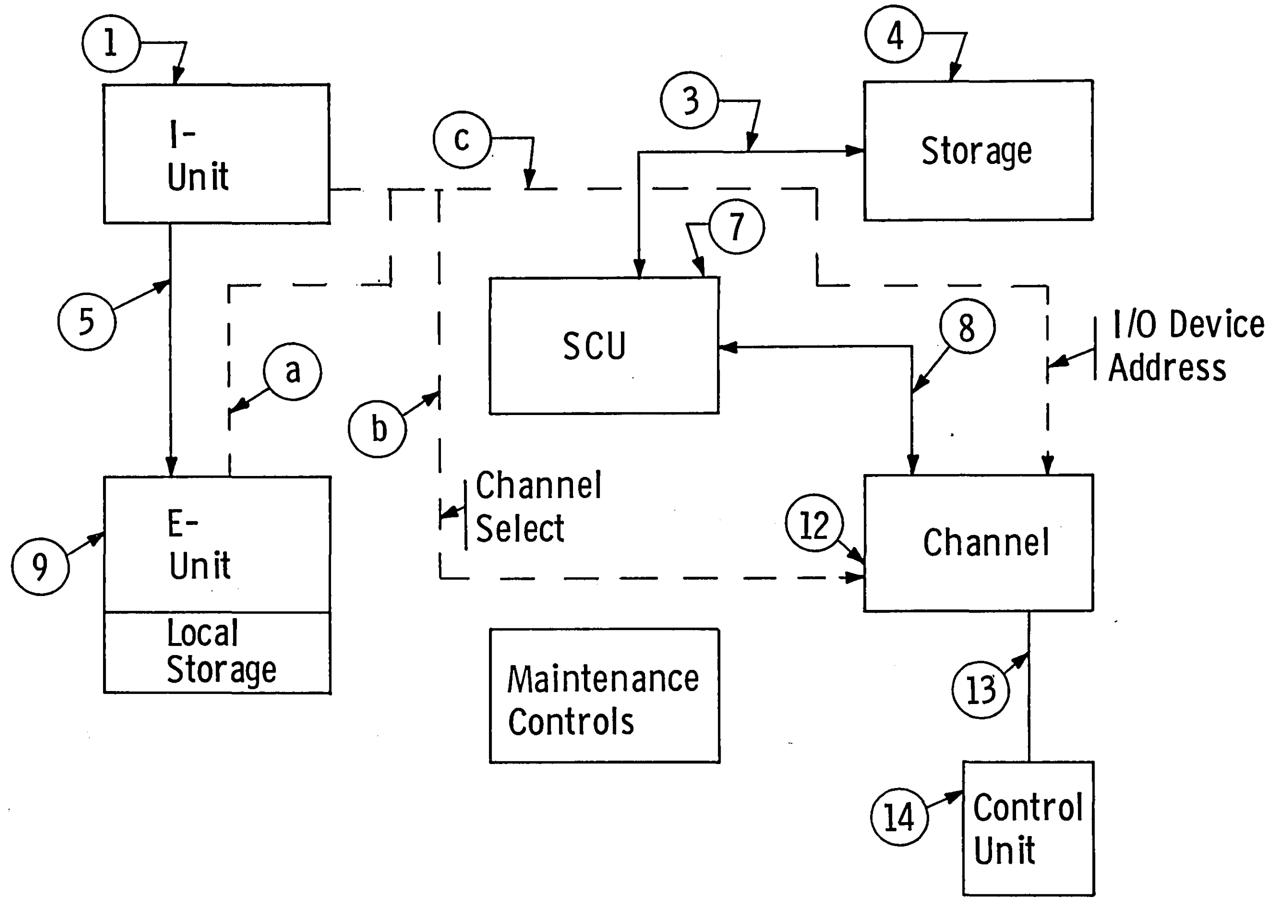


Figure 4-21 Start I/O Basic Logic Flow 1



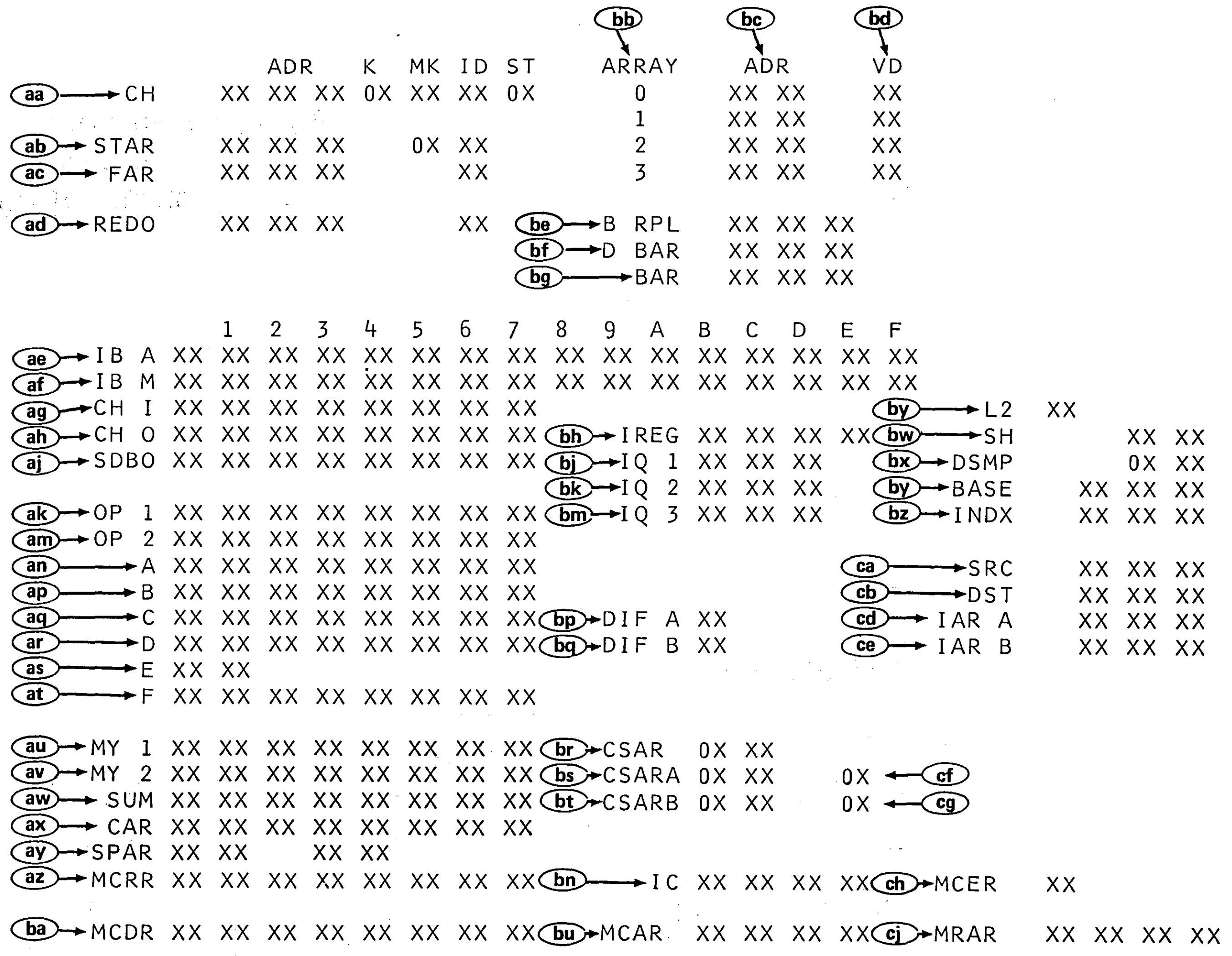
Note: Instruction Fetch Omitted
Error Re-try Omitted

Figure 4-22 Start I/O Basic Logic Flow 2

Partial List of System Logic Units

1. D BAR
2. Address Adder
3. LAR Buffer Latch
4. ACAL
5. MCDR
6. Buffer Bypass Latch
7. L2 Register
8. Parallel Adder
9. Channel Data-Out Buffer
10. A-Register
11. Shift Control Triggers
12. Destination Register
13. Main I-Buffer
14. A-Pointer
15. C-Register
16. STAR
17. FAR
18. Source Register
19. LAL 5
20. F-Register
21. Instruction Queues
22. Serial Adder
23. MCWR
24. Difference Register B
25. CSAR
26. B-Register

Figure 5-1 Partial List of System Logic Units



```
                                                         (bb)        (bc)       (bd)
                                                           ↓           ↓          ↓
                    ADR        K   MK  ID  ST          ARRAY        ADR         VD
(aa)───→CH      XX  XX  XX  0X  XX  XX  0X              0         XX  XX       XX
                                                        1         XX  XX       XX
(ab)→ STAR      XX  XX  XX      0X  XX                  2         XX  XX       XX
(ac)→  FAR      XX  XX  XX          XX                  3         XX  XX       XX

(ad)→REDO       XX  XX  XX          XX   (be)→B RPL      XX  XX  XX
                                         (bf)→D BAR      XX  XX  XX
                                         (bg)───→BAR     XX  XX  XX

                 1   2   3   4   5   6   7   8   9   A   B   C   D   E   F
(ae)→IB A  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX
(af)→IB M  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX  XX
(ag)→CH I  XX  XX  XX  XX  XX  XX  XX  XX                     (by)───→L2    XX
(ah)→CH O  XX  XX  XX  XX  XX  XX  XX  XX(bh)→IREG XX XX XX XX(bw)───→SH            XX  XX
(aj)→SDBO  XX  XX  XX  XX  XX  XX  XX  XX(bj)→IQ 1 XX XX XX   (bx)→DSMP             0X  XX
                                         (bk)→IQ 2 XX XX XX   (by)→BASE         XX  XX  XX
(ak)→OP 1  XX  XX  XX  XX  XX  XX  XX  XX(bm)→IQ 3 XX XX XX   (bz)→INDX         XX  XX  XX
(am)→OP 2  XX  XX  XX  XX  XX  XX  XX  XX
(an)────→A XX  XX  XX  XX  XX  XX  XX  XX                     (ca)───→SRC       XX  XX  XX
(ap)────→B XX  XX  XX  XX  XX  XX  XX  XX                     (cb)───→DST       XX  XX  XX
(aq)────→C XX  XX  XX  XX  XX  XX  XX  XX(bp)→DIF A XX        (cd)→ IAR A       XX  XX  XX
(ar)────→D XX  XX  XX  XX  XX  XX  XX  XX(bq)→DIF B XX        (ce)→ IAR B       XX  XX  XX
(as)────→E XX  XX
(at)────→F XX  XX  XX  XX  XX  XX  XX  XX

(au)→MY 1  XX  XX  XX  XX  XX  XX  XX  XX(br)→CSAR  0X XX
(av)→MY 2  XX  XX  XX  XX  XX  XX  XX  XX(bs)→CSARA 0X XX      0X ←(cf)
(aw)→ SUM  XX  XX  XX  XX  XX  XX  XX  XX(bt)→CSARB 0X XX      0X ←(cg)
(ax)→ CAR  XX  XX  XX  XX  XX  XX  XX  XX
(ay)→SPAR  XX  XX      XX  XX
(az)→MCRR  XX  XX  XX  XX  XX  XX  XX  XX(bn)───→IC XX XX XX XX(ch)→MCER     XX

(ba)→MCDR  XX  XX  XX  XX  XX  XX  XX  XX(bu)→MCAR  XX XX XX XX(cj)→MRAR     XX  XX  XX  XX
```

Figure 5-2 Session 5 Question 12-17

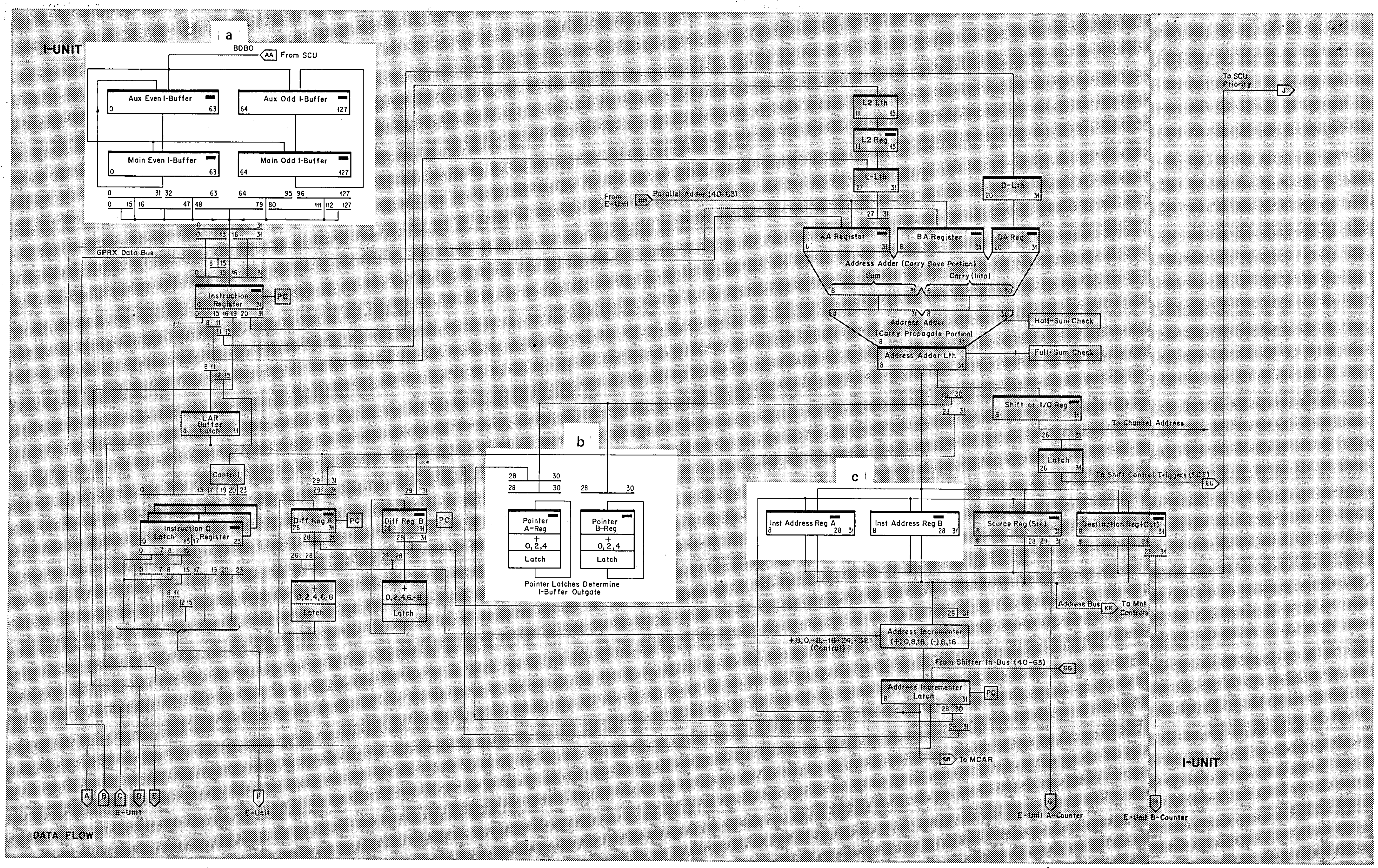


Figure 5-3 I Buffers

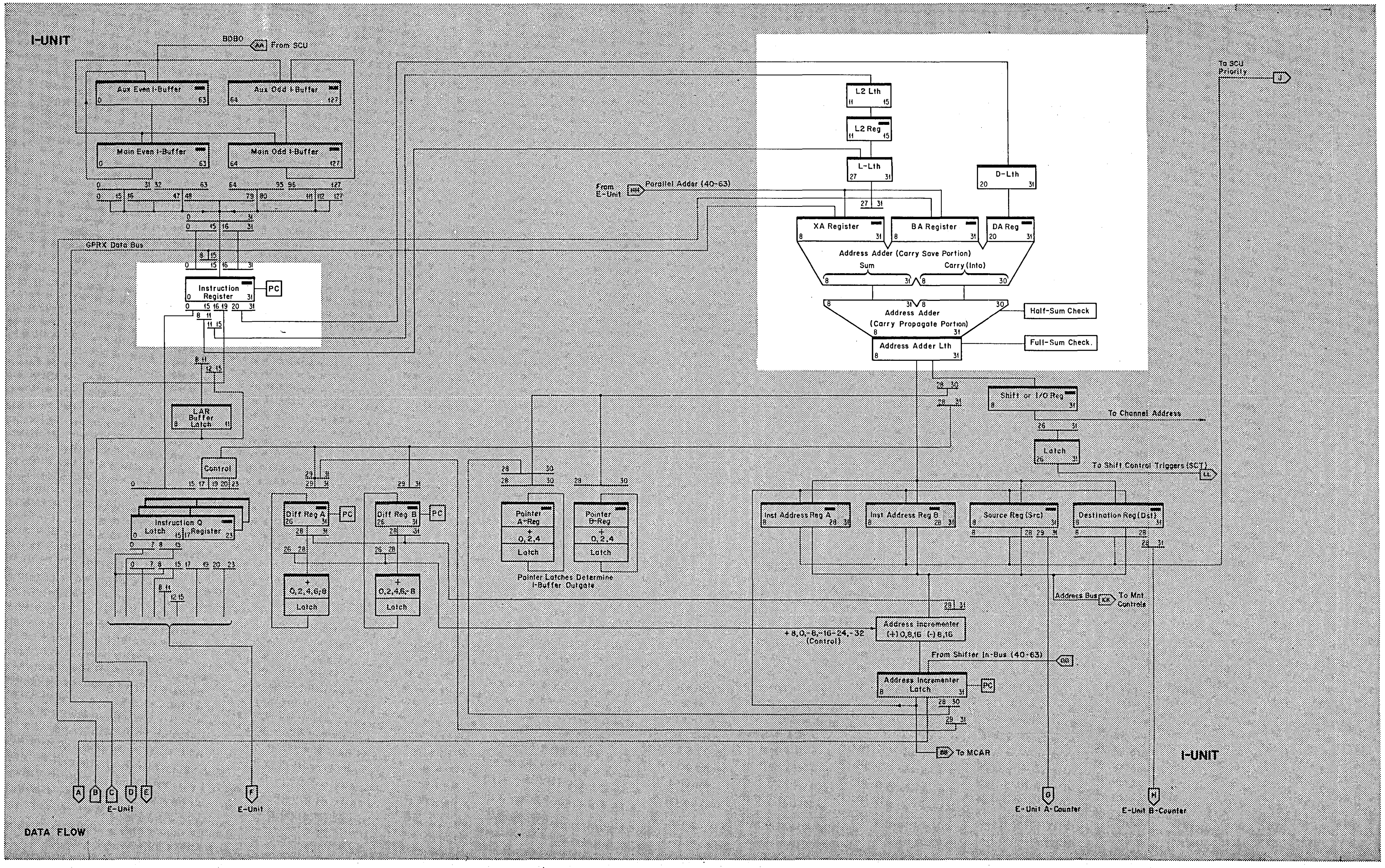


Figure 5-4 I-Register

Address Adder

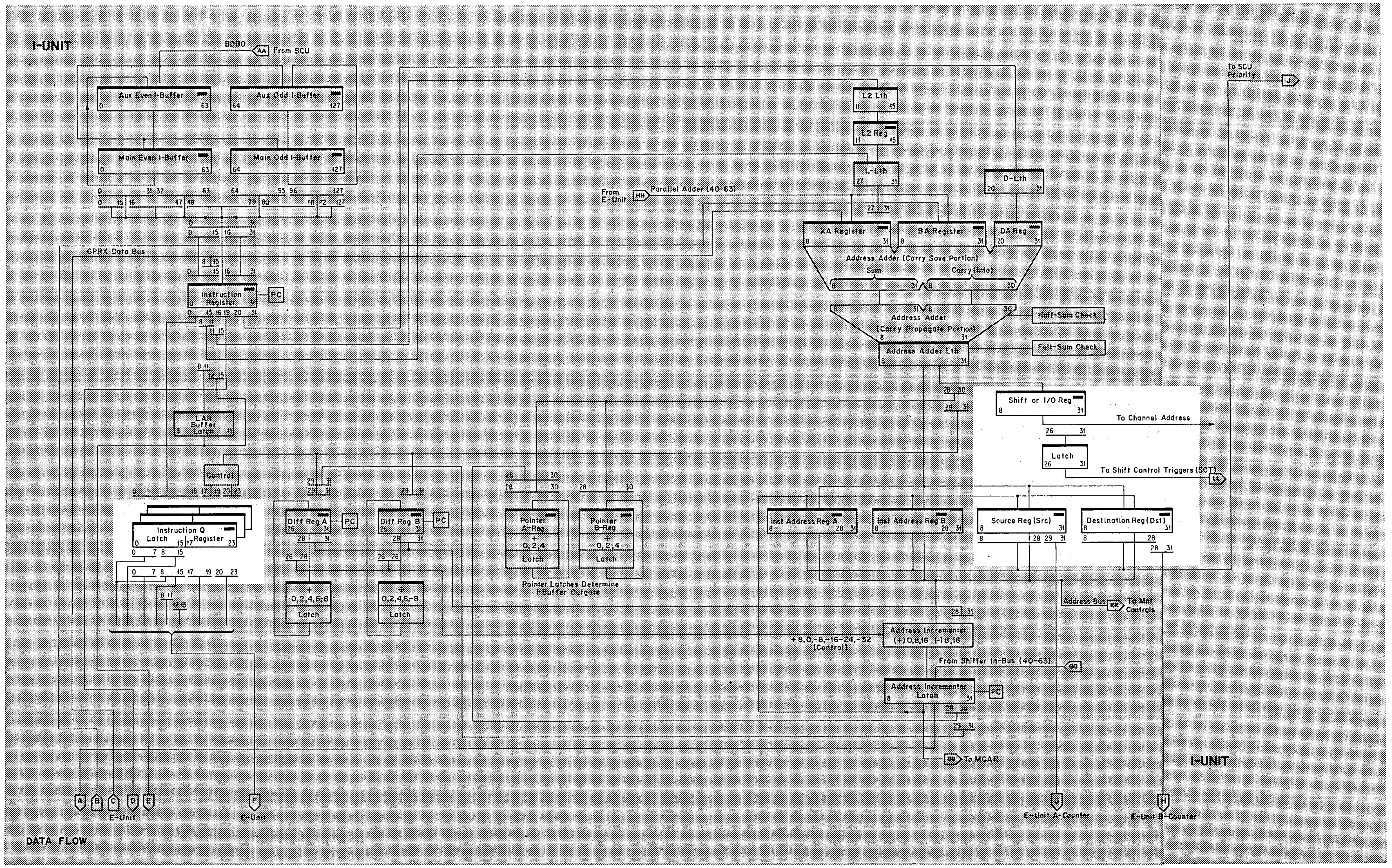


Figure 5-5 Instruction Queues Address Registers

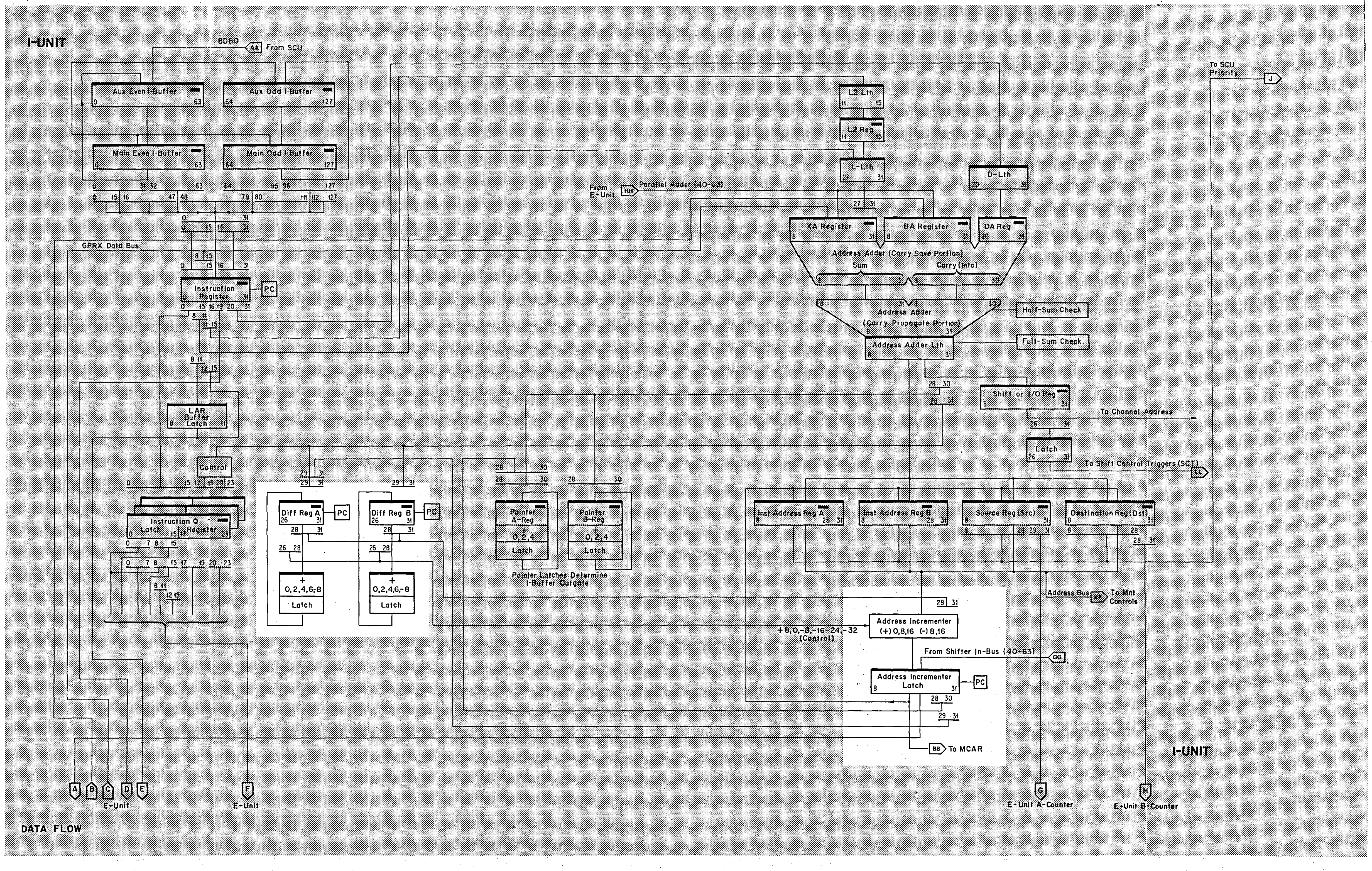


Figure 5-6 Difference Register

Address Incrementer

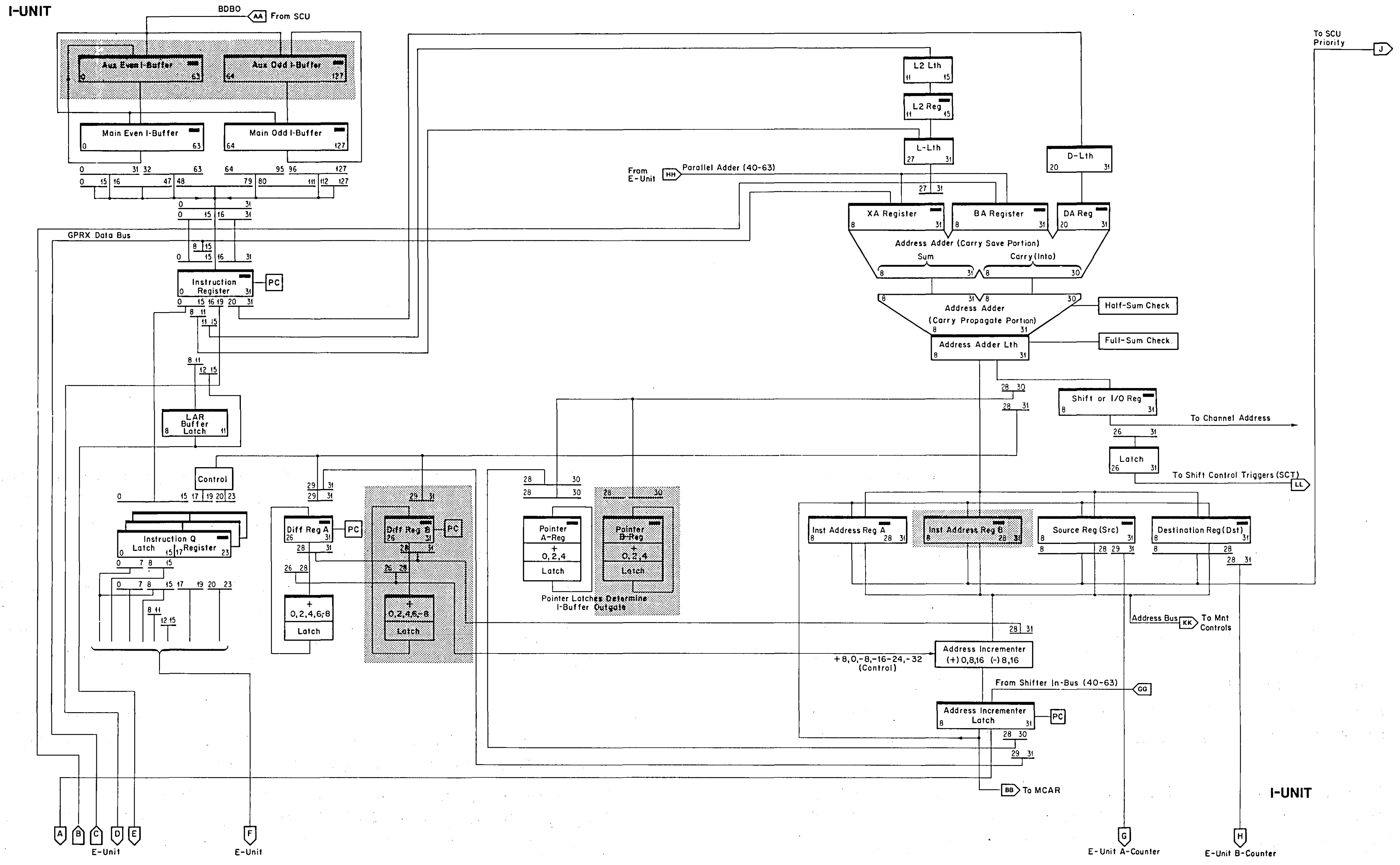


Figure 5-7 Revised Data Flow

INSTRUCTION BUFFER

To accelerate preprocessing of instructions, the main instruction buffer stores up to one quadword of the current instruction stream in the I-unit to provide extremely rapid access to consecutive instructions for preprocessing. A doubleword of the "instruction stream" is fetched from main storage and placed on a bus called "Buffer Data Bus-Out" (BDBO) by the Storage Control Unit (SCU). From BDBO the instructions can then be ingated to either half of the instruction buffer. A fetch to main storage is initiated immediately when either half of the I-buffer is depleted, to keep the contents of the I-buffer as current and as far ahead of the I-unit processing as possible.

Instructions are outgated one-at-a-time from the main instruction buffer to the instruction register under the control of the pointer A-register. Four bytes, starting with an even byte, are sent to the instruction register. If the left two bytes contain an RR instruction, the right-most two bytes are ignored. After an instruction is outgated from the I-buffer, the pointer is updated by the length of the instruction to keep track of which byte is the beginning of the next instruction. The pointer is also instrumental in determining when a new main storage fetch must be initiated to refill each half of the instruction buffer as soon as it is emptied.

Figure 5-8 I-Buffer

ADDRESS REGISTERS

The four 24-bit address registers used to make storage requests are: instruction address register A (IARA), instruction address register B (IARB), source (src) address register, and destination (dst) address register. Instruction address registers A and B are used exclusively for fetching instructions. The source address register is used only for fetching operands from main storage. The destination address register is used for storing operands into main storage.

For instructions requiring operands of eight bytes or less, the source address register is controlled only by the I-unit. For instructions requiring longer operands (load multiple and SS format), the initial fetch request is made by the I-unit, and control then passes to the E-unit for subsequent requests. The destination address register is used to hold the address during store requests, which are always made by the E-unit.

Any of the four registers can be ingated from the address-adder latch or the address-incrementer latch. The contents of any of the four registers can be transferred to the address-incrementer for incrementing or the SCU address latches for accessing main storage.

Figure 5-9 Address Registers

POINTERS

A special type of functional unit associated with the I-unit is the "pointer." Pointers keep track of individual instructions in the instruction buffers and the queue registers. Because the I-unit handles several instructions at a time, they are needed to ensure that instructions are processed in the proper sequence.

Address Register Pointers (Pointer A-Register and Pointer B-Register)

Two pointers, one associated with each of the two instruction address registers, select the word that is to be transferred from an instruction buffer to the instruction register. As instructions are transferred from the instruction register to the instruction queues, the contents of the pointer register are incremented according to the instruction length and are transferred to the pointer latch. The contents of the pointer latch then select the next word that is to be transferred from the instruction buffer to the instruction register.

Figure 5-10 Address Register Pointers

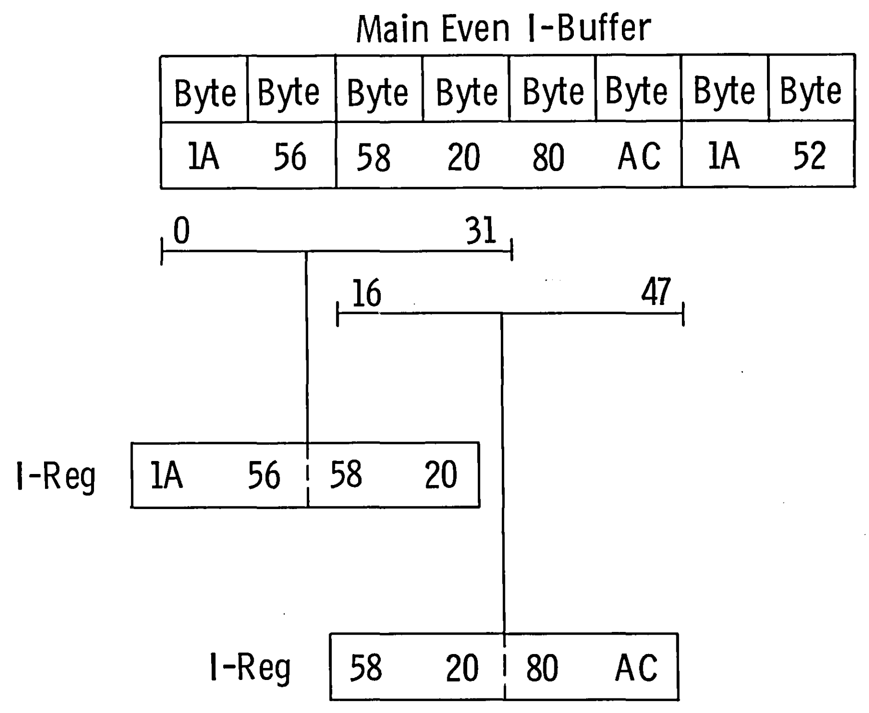


Figure 5-11 I-Reg Ingate

INSTRUCTION REGISTERS

The four-byte instruction register holds each instruction during a decode cycle. Ingating from the main instruction buffer is determined by pointer A-register, and by the format of the instruction itself (RR, RX, RS, SI, or SS). Each half of the instruction register can be ingated separately. Parity is checked on a byte basis.

Each instruction processed by the I-unit remains in the instruction register for at least one machine cycle for decoding. During this time, the instruction is decoded to determine all the information concerning the instruction that the I-unit requires to preprocess it. If all interlocking conditions are not satisfied, or if any I-unit facility (eg, the address adder or an address register) is busy with an earlier instruction, the decode cycle is delayed until all interlocking conflicts are resolved or the required facilities become available. The following conditions must be satisfied for an instruction to be decoded:

1. A queue register must be available (not busy) to receive the instruction.
2. Required general purpose registers in the E-unit, and the address adder must be available if an address calculation is required.
3. If an SS format instruction is being processed in the I-unit or the E-unit, only branch and RR format instructions are decoded.
4. If a defeat I-unit/E-unit overlap condition exists, no instruction can be decoded until all instructions preceding it complete their execution cycle. A defeat I-unit/E-unit overlap condition occurs when the overlap switch is set at the DSBL position, or when an instruction that required I-unit/ E-unit overlap to be defeated is decoded in the I-register.
5. If the instruction to be decoded is a branch, no other branch instruction can be in progress in the I-unit or E-unit.

Figure 5-12 Instruction Register

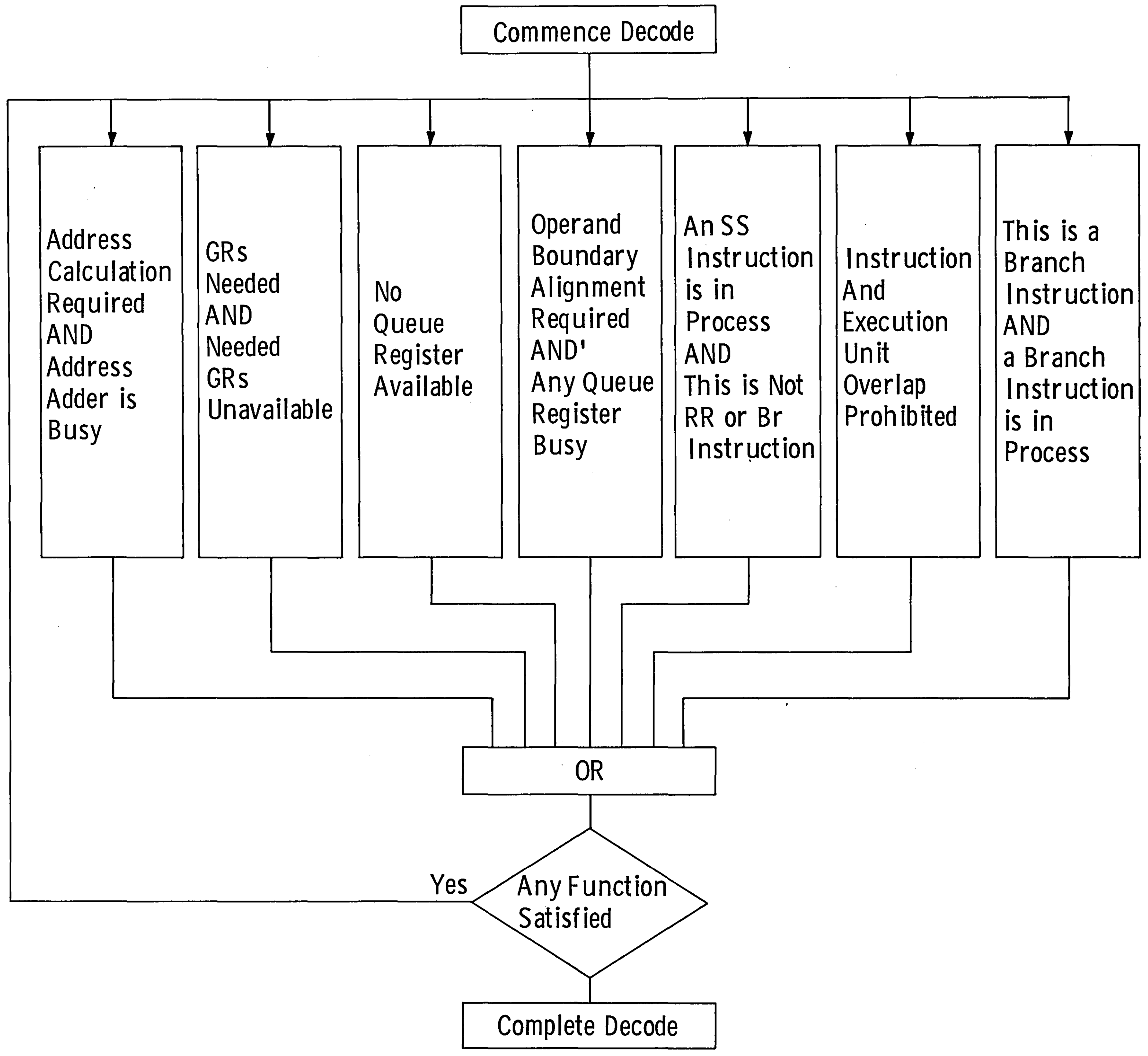

Commence Decode
Address Calculation Required AND Address Adder is Busy
GRs Needed AND Needed GRs Unavailable
No Queue Register Available
Operand Boundary Alignment Required AND' Any Queue Register Busy
An SS Instruction is in Process AND This is Not RR or Br Instruction
Instruction And Execution Unit Overlap Prohibited
This is a Branch Instruction AND a Branch Instruction is in Process
OR
Yes
Any Function Satisfied
Complete Decode

Figure 5-13 Instruction Decode

ADDRESS ADDER

Instructions that reference main storage for operands or that require address-type operands (shift and I/O instructions) must have addresses or shift amounts calculated for them using information supplied by the X, B, and D fields of the instruction format. These calculations are performed by the I-unit address adder prior to E-unit execution using base address (B), index (X), and displacement (D) components specified by the instruction.

To briefly review System/360 Principles of Operation on these address components they are defined as follows:

1. The base address is a 24-bit number in a GPR specified by the B-field.

2. The index is a 24-bit number in a GPR specified by the X-field.

3. The displacement is a 12-bit number in the instruction format.

In forming the address, the base address and index are treated as unsigned 24-bit positive binary integers (whole numbers). The displacement is treated as a 12-bit positive integer. The three are added as 24-bit binary numbers, ignoring overflow. Because every address includes a base, the sum is always 24-bits long. The address bits are numbered 8-31 corresponding to the numbering of the base address and index bits in the GPR. The instruction may have zeros in the base address, index, or displacement fields. A zero means that there is no corresponding address component. A base or index of zero implies that a zero quantity is to be used in forming the address, regardless of the contents of GPR0.

Because the 24-bit (plus 3 parity bits) address adder performs the additions involved in these address calculations, it has three input registers: DA, BA, and XA. Up to three address components (displacement, base address, and index) can be added and checked in one machine cycle.

Figure 5-14 Address Adder

The principal inputs to the adder input registers are two local storage buses (GPR X data bus and GPR B data bus) and the D-latches (which receives its input from the instruction register) During the instruction decode cycle, the required GPRs are read from local storage, and the D-field is set up. When the decode cycle is successful, the quantities are ingated to the adder input registers and then added in the next cycle.

When an instruction requires a store request, the result of the address calculation is put into the destination address register (dst). If dst is busy, the contents of the adder input registers are held until the destination register becomes not busy.

When an instruction requires a fetch request, the result of the address calculations is sent to the source address register (src) and a request is made during the same cycle in which the address is calculated. If src or both the operand buffers are busy, the contents of the adder input registers are held. The address components are retained in the XA, BA, and DA registers and re-added each cycle until the busy condition is cleared.

For shift and I/O instructions, the calculated address-type operand is put into the shift or I/O register. As in other instructions, the contents of the adder input registers are held until the receiving register is available.

Address Adder (continued)

INSTRUCTION QUEUE REGISTERS

The three instruction queue (IQ) registers allow up to three instructions to be preprocessed in the I-unit ahead of actual E-unit execution. The instructions are stacked in the IQ-registers until needed by the E-unit. Each IQ-register consists of two sections. Refer to the system data flow in Volume 1 of the FEMDM. Note that bits 0-15, which are timed as latches, hold the first two bytes of the instruction. Bits 17-23, which are timed as triggers, hold the initial values for the A-counter adder latches (ACAL) and B-counter adder latches (BCAL). (Note that there is no bit 16 in the queues.)

The IQ-registers are ingated and outgated in a 1, 2, 3, 1... cycle sequence. Two pointers are provided: The inpointer indicates which IQ-register is next to be loaded from the I-register, and the outpointer indicates which IQ-register contains the next instruction to be sent to the E-unit for execution. These pointers are not shown on the data flow.

Figure 5-15 Instruction Queue Registers

Instruction Queue Pointers

Two three-bit pointers are associated with the instruction queues: an inpointer that indicates which queue is next to be loaded from the instruction register, and an outpointer that indicates which queue's contents are to be used next by the E-unit. This pointer is not shown on the data flow. The IQ-inpointer steps 1, 2, 3, 1 ... etc, and works with busy triggers associated with each queue. Every cycle the contents of the instruction register are gated to all nonbusy queues. When conditions are present to allow the instruction to be released from the instruction register, the queue pointed to by the INPOINTER is loaded, made busy, and the inpointer is stepped to point to the next queue.

The instruction sent to the E-unit for execution is always from the queue pointed to by the outpointer. At the same time, the queue whose contents were outgated is made not busy and the outpointer is stepped to point to the next queue. The outpointer steps 1, 2, 3, 1... just as the inpointer does.

These pointers ensure that the instructions are kept in sequence as they are transferred to the E-unit.

Figure 5-16 IQ Pointers

Shift or I/O Register

The address adder generates shift counts and I/O addresses as if a main storage address were being generated. When operands for shift and I/O instructions are being prefetched, the I-unit transfers the contents of address-adder latches to the "shift I/O" register. The address adder can then be reused without waiting for execution of the shift or I/O instruction. Because shifting is performed by the E-unit's shifter, the shift-count bit positions of the "shift or I/O" register are transferred to a six-bit shift latch to preserve trigger-latch timing relationship This latch is shown under the shift or I/O regist er on the data flow. Its output goes to the shift control triggers, which are part of the shifter, located in the E-unit.

During the execution of I/O instructions, the shift or I/O register contents are gated to the channel-selection logic.

Figure 5-17 Shift or I/O Register

Difference Registers

Two difference registers, each associated with an instruction address register, permit calculations of the true instruction count (IC) when needed. Because the I-unit prefetches instructions up to two doublewords ahead of the instruction being executed, neither instruction address register contains the actual instruction counter value of the instruction in the E-unit. Two registers (difference registers A and B) contain in twos complement form the difference between the instruction counter (IC) value and the corresponding instruction address register. The IC value can be obtained when needed by ADDing the active IAR and its corresponding difference reg ister.

The difference register (A when using IAR A) keeps track of how far ahead I-unit instruction processing is with respect to the E-unit. An incrementer associated with each difference register can increment or decrement the contents of the difference register or can do both. When an instruction address register is set to a new value, the four low-order bits are transferred to the four low-order bits of the difference register; the remaining two bits of the difference register are reset to zeros. As instruction processing progresses, the contents of the difference register are changed such that at any point the sum of the instruction address and the difference equals the current IC. To keep the difference register value correct, the difference register is decremented by 8 each time the I-unit increments the instruction address by 8.

Address incrementer bits 29-31 are always transferred to difference register A after the E-unit or the I-unit calculates the IC transfer into instruction address register B because the I-unit always restarts with IARA active after an operation that requires IC to be calculated.

Figure 5-18 Difference Register

Address Incrementer

The 24-bit address incrementer can increment or decrement the contents of the four address registers and can be used as a path between the I-unit and the E-unit. The I-unit uses the incrementer for:

1. Adding 8 to the appropriate instruction address register A or B, when updating to the address of the next doubleword of instructions.

2. Calculating the instruction-counter (IC) value from the appropriate instruction address and difference register and transferring the result to instruction address register A This calculation is part of the program store compare recovery operation.

The E-unit also has access, and can use, the incrementer.

1. The E-unit uses the incrementer to add or subtract from the contents of the source register and destination registers.

2. Transfer main storage addresses from the shifter in-bus to the appropriate address register. The transferred address is neither shifted nor incremented.

3. Transfer the contents of an address register or the value of the IC to the maintenance controls address register.

Note: If a conflict exists between the I-unit and the E-unit for usage of the incrementer, the I-unit waits.

Figure 5-19 Address Incrementer

```
           ADR   K  MK ID ST     ARRAY    ADR     VD
  CH    XX XX XX 0X XX XX 0X       0     XX XX    XX
                                   1     XX XX    XX
STAR    XX XX XX    0X XX          2     XX XX    XX
 FAR    XX XX XX       XX          3     XX XX    XX

REDO    XX XX XX       XX       B RPL    XX XX XX
                                D BAR    XX XX XX
                                  BAR    XX XX XX

         1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
IB M XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
CH I XX XX XX XX XX XX XX XX                           L2   XX
CH O XX XX XX XX XX XX XX XX     IREG XX XX XX XX      SH          XX XX
SDBO XX XX XX XX XX XX XX XX     IQ 1 XX XX XX       DSPM          0X XX
                                 IQ 2 XX XX XX       BASE       XX XX XX
OP 1 XX XX XX XX XX XX XX XX     IQ 3 XX XX XX       INDX       XX XX XX
OP 2 XX XX XX XX XX XX XX XX
   A XX XX XX XX XX XX XX XX                          SRC       XX XX XX
   B XX XX XX XX XX XX XX XX                          DST       XX XX XX
   C XX XX XX XX XX XX XX XX    DIF A XX            IAR A       XX XX XX
   D XX XX XX XX XX XX XX XX    DIF B XX            IAR B       XX XX XX
   E XX XX
   F XX XX XX XX XX XX XX XX

                                CSAR  0X XX
                                CSARA 0X XX    0X
                                CSARB 0X XX    0X


MCRR XX XX XX XX XX XX XX XX       IC XX XX XX XX    MCER   XX

MCDR XX XX XX XX XX XX XX XX    MCAR  XX XX XX XX    MRAR   XX XX XX XX
```

Figure 5-20 CRT Display

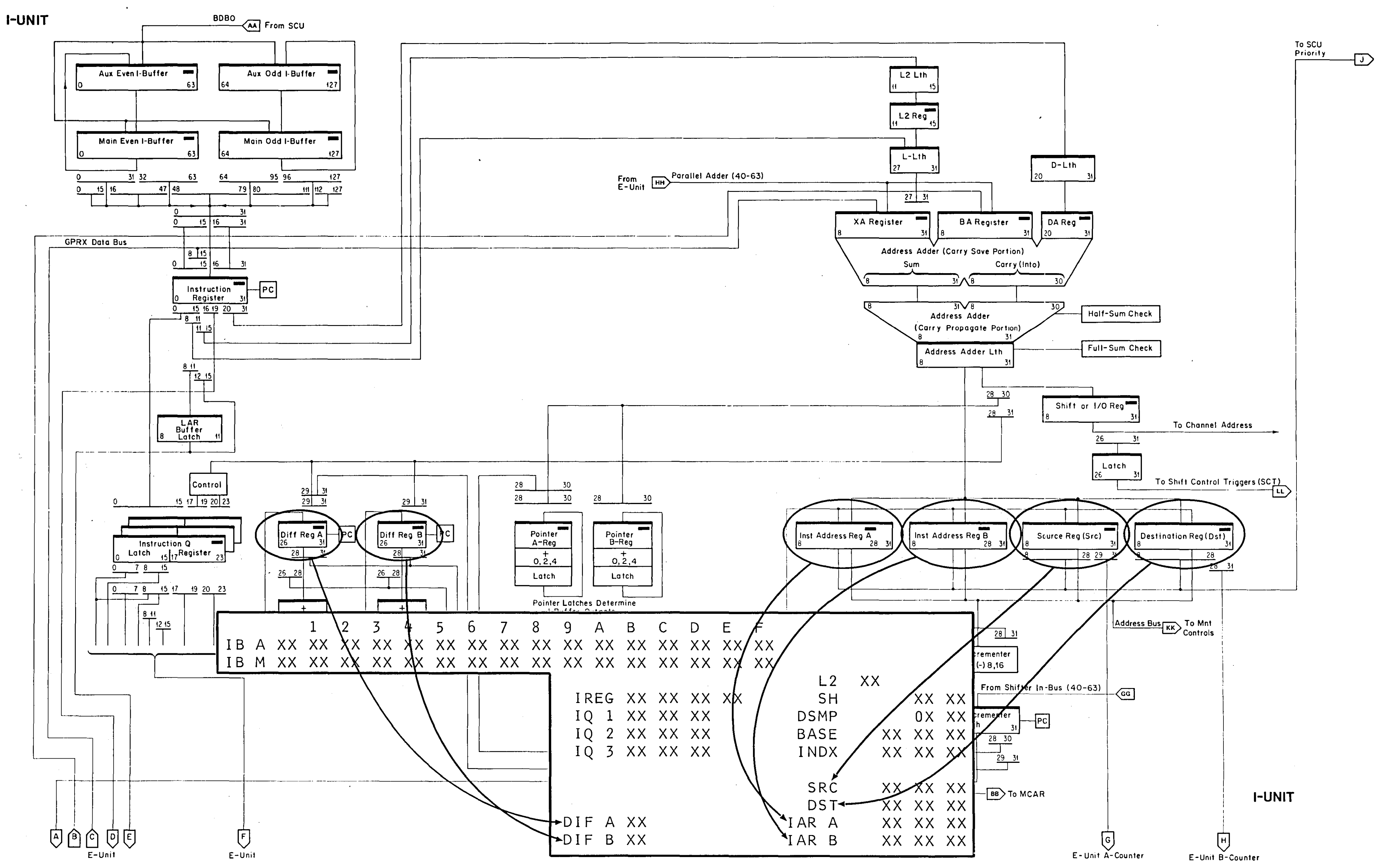


Figure 5-21 Difference Register Displays

Address Register Displays

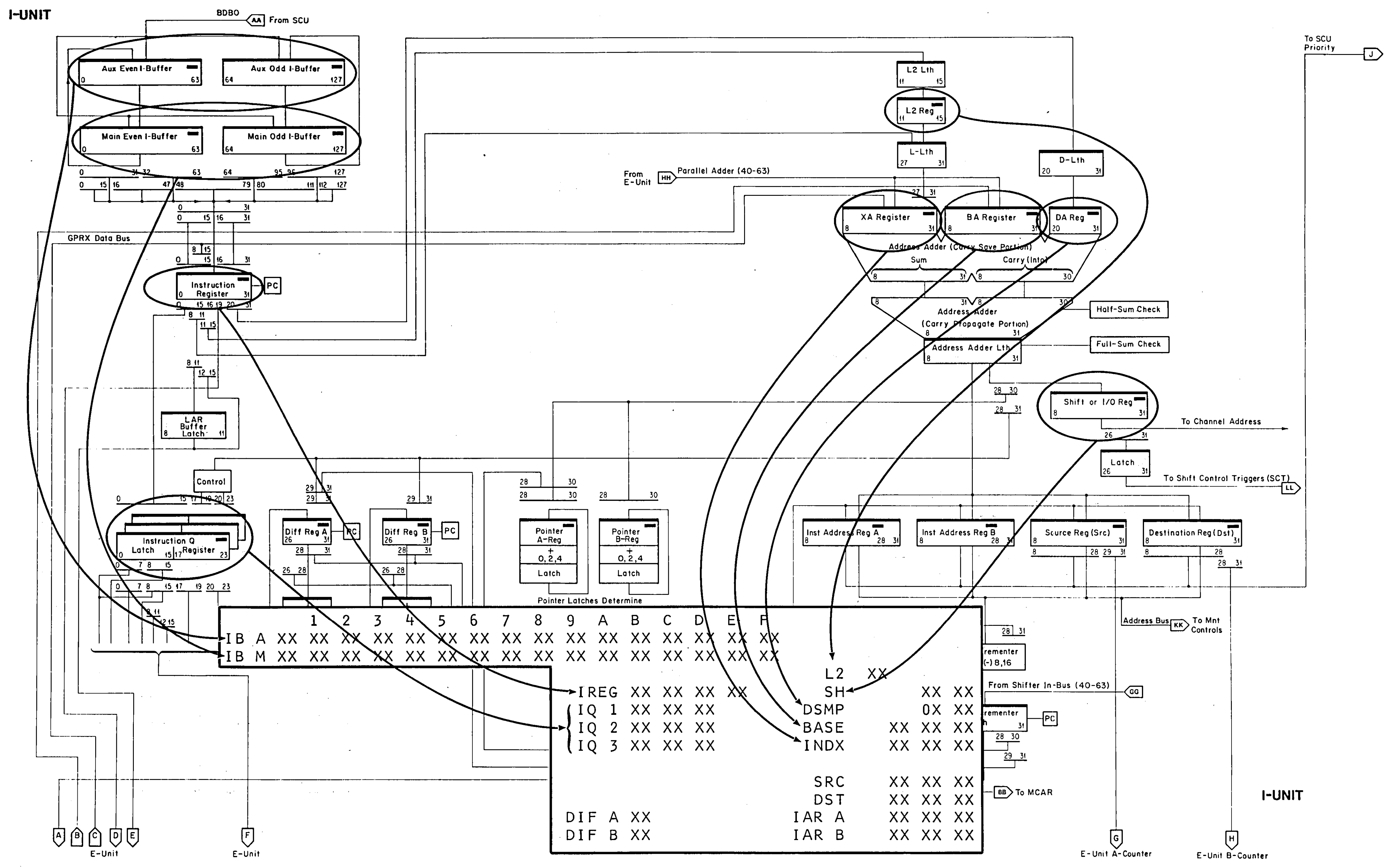


Figure 5-22 I-Buffer, I-Register, I-Q Display

Shifter Inputs and I/O Reg Display

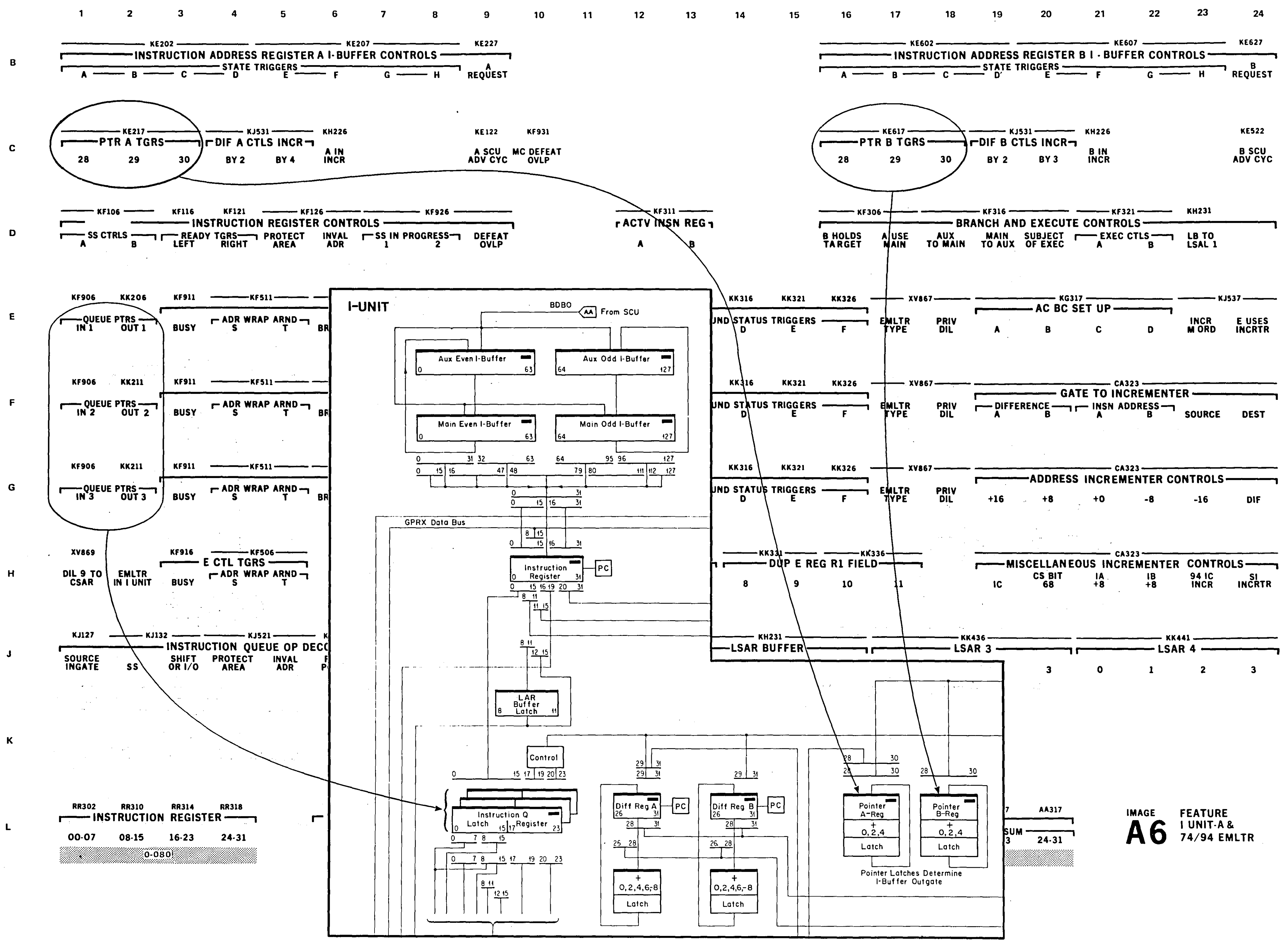


Figure 5-23 Pointer Displays

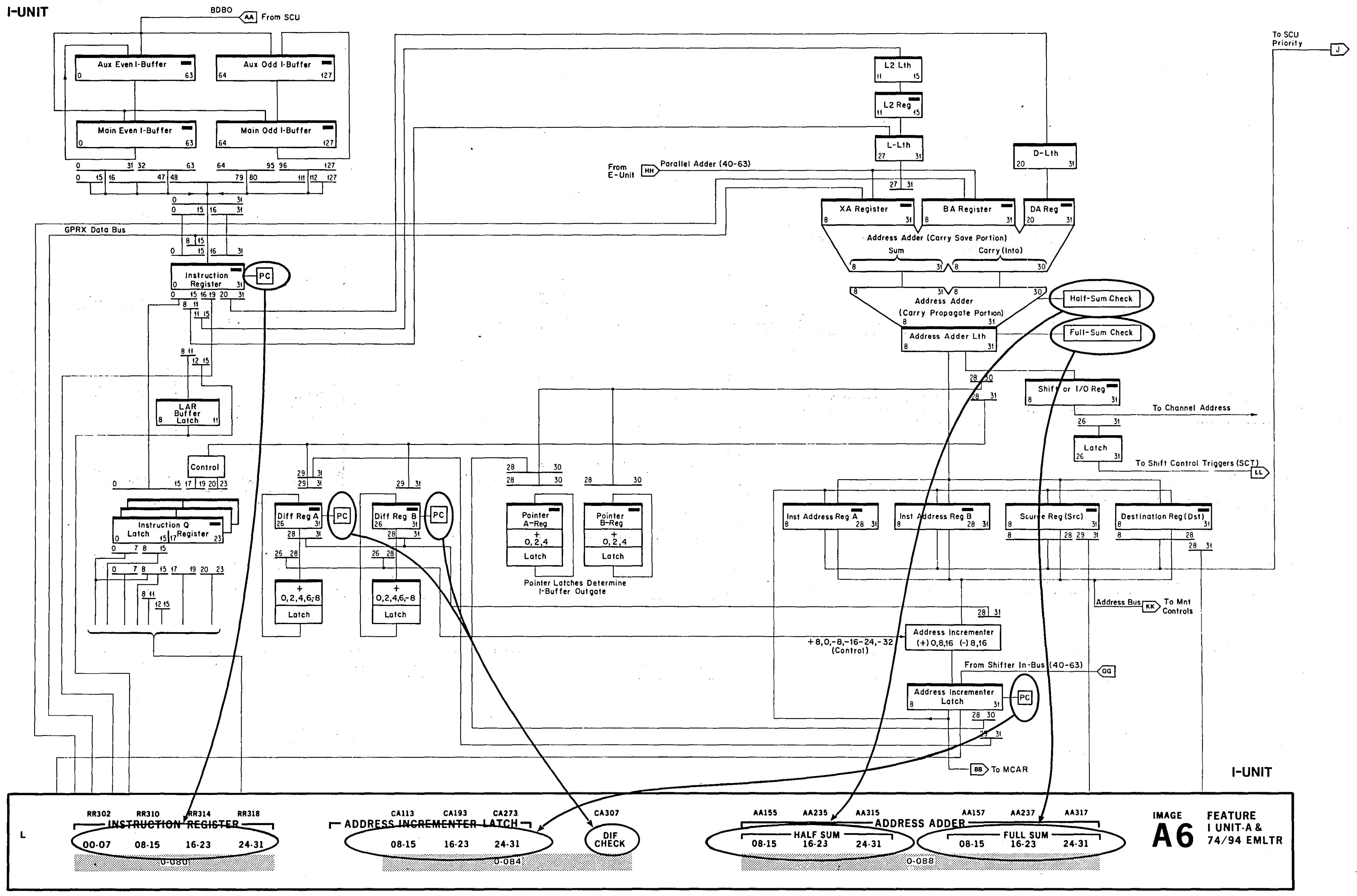


Figure 5-24 I-Unit Errors

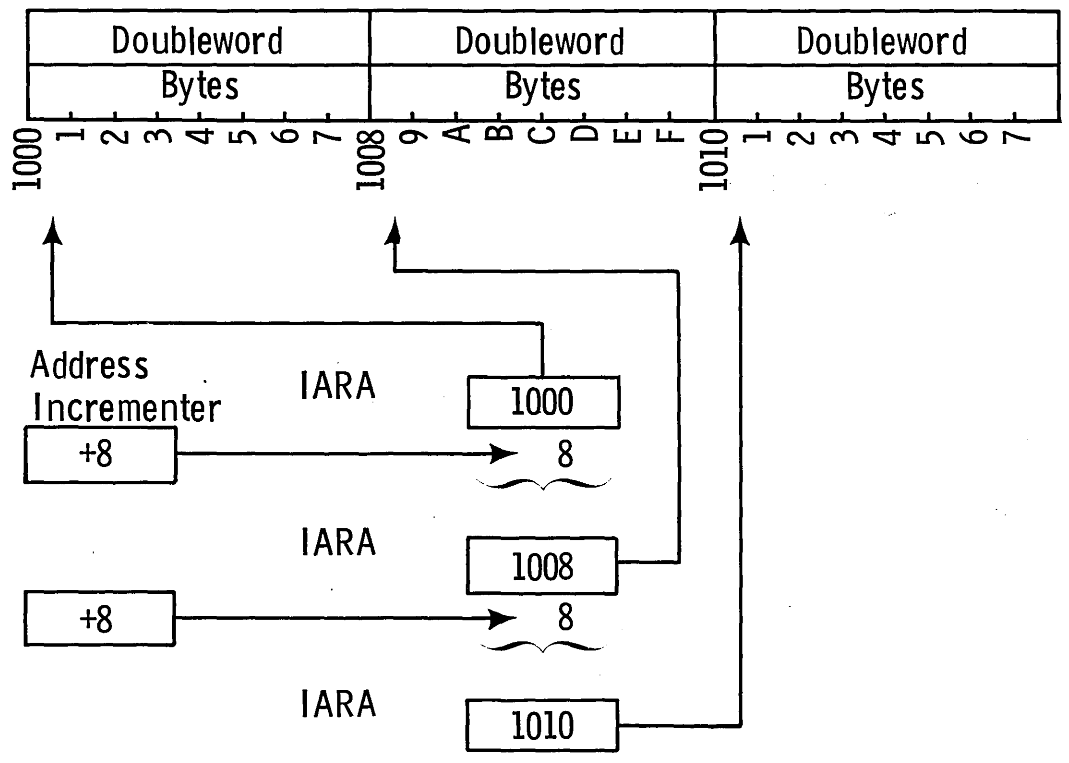


Figure 5-25. Address Update

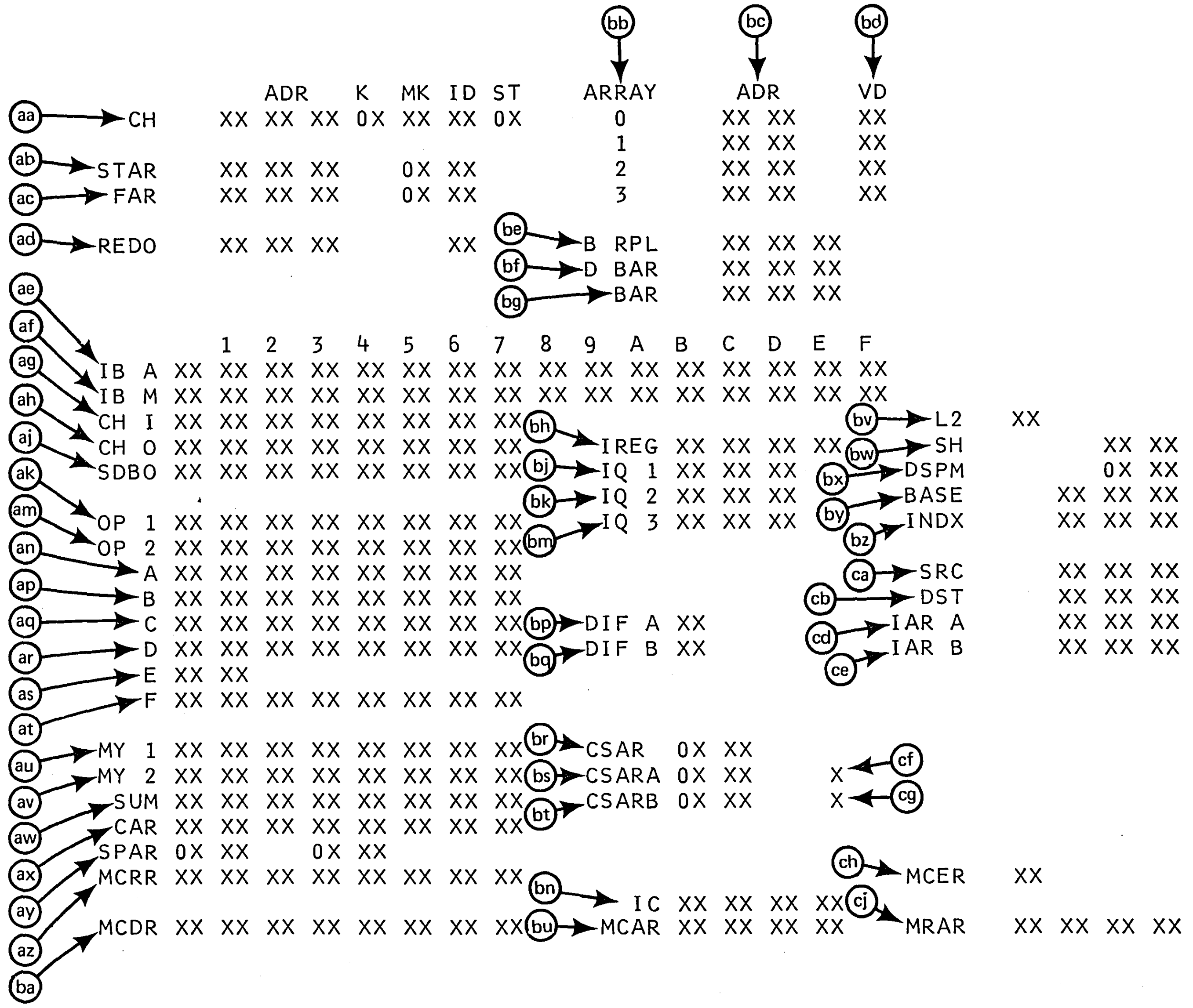


Figure 5-26. Questions 12-17 Remedial

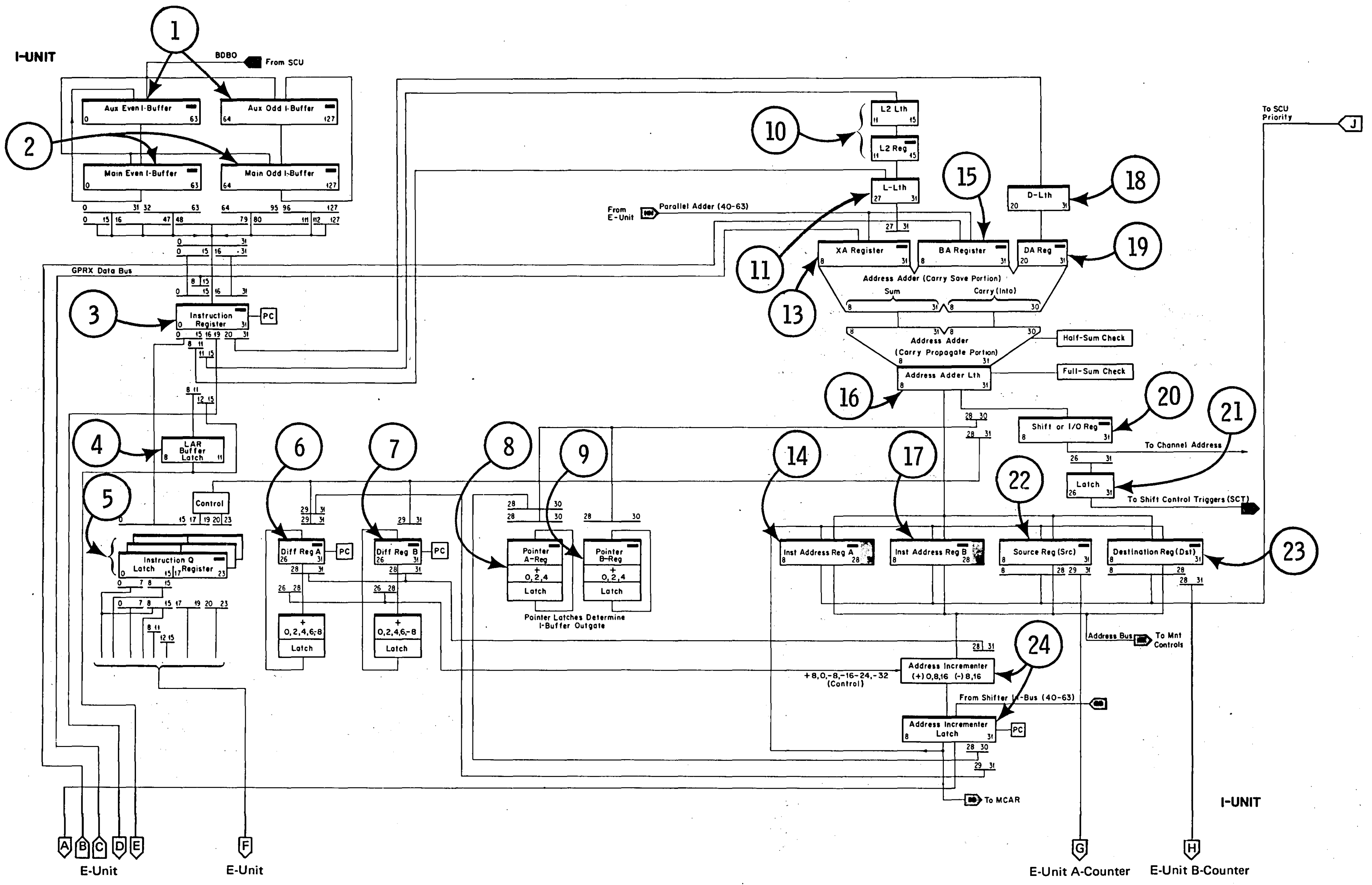


Figure 6-1. Session 6, Questions 10-15

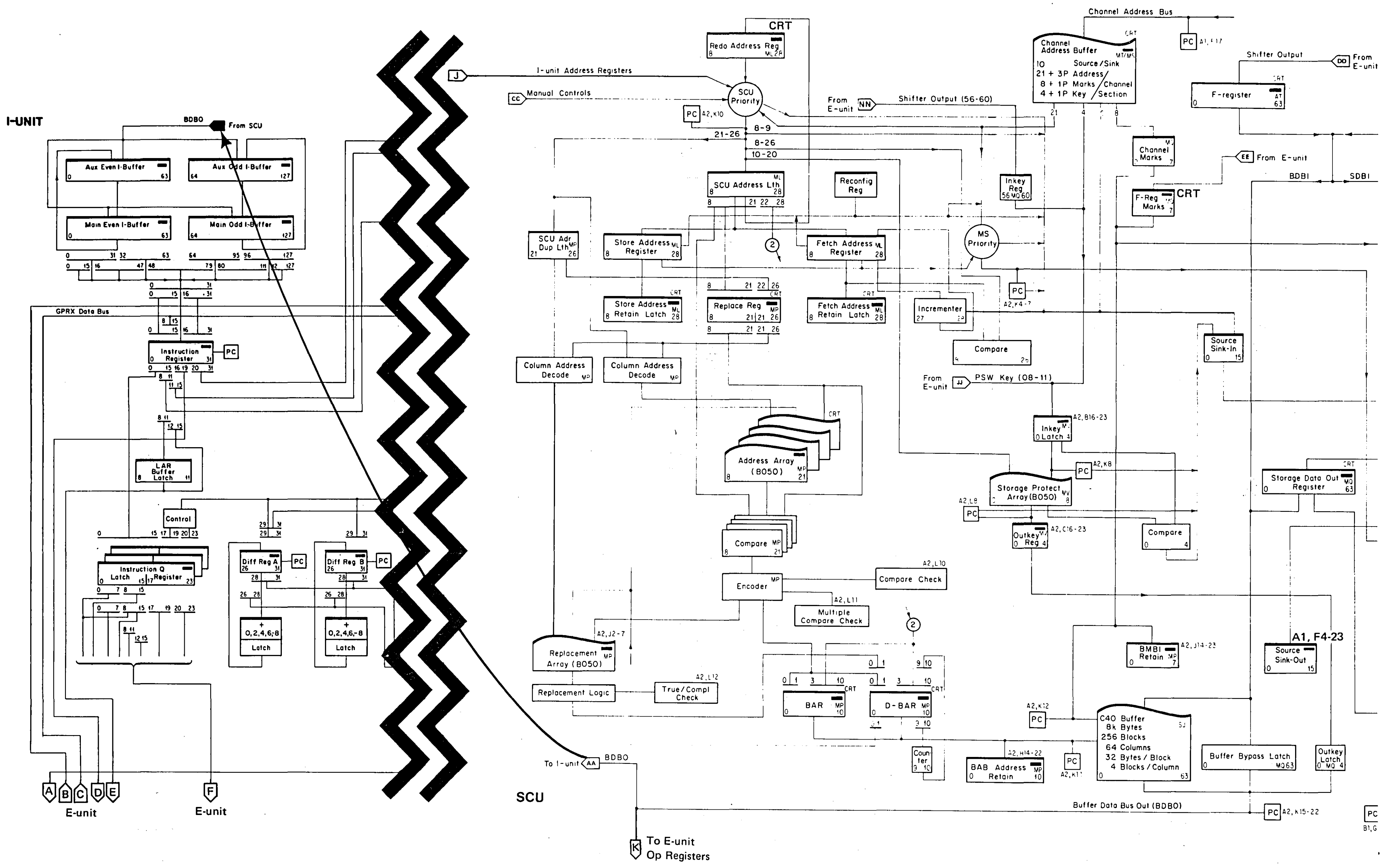


Figure 6-2. BDBO

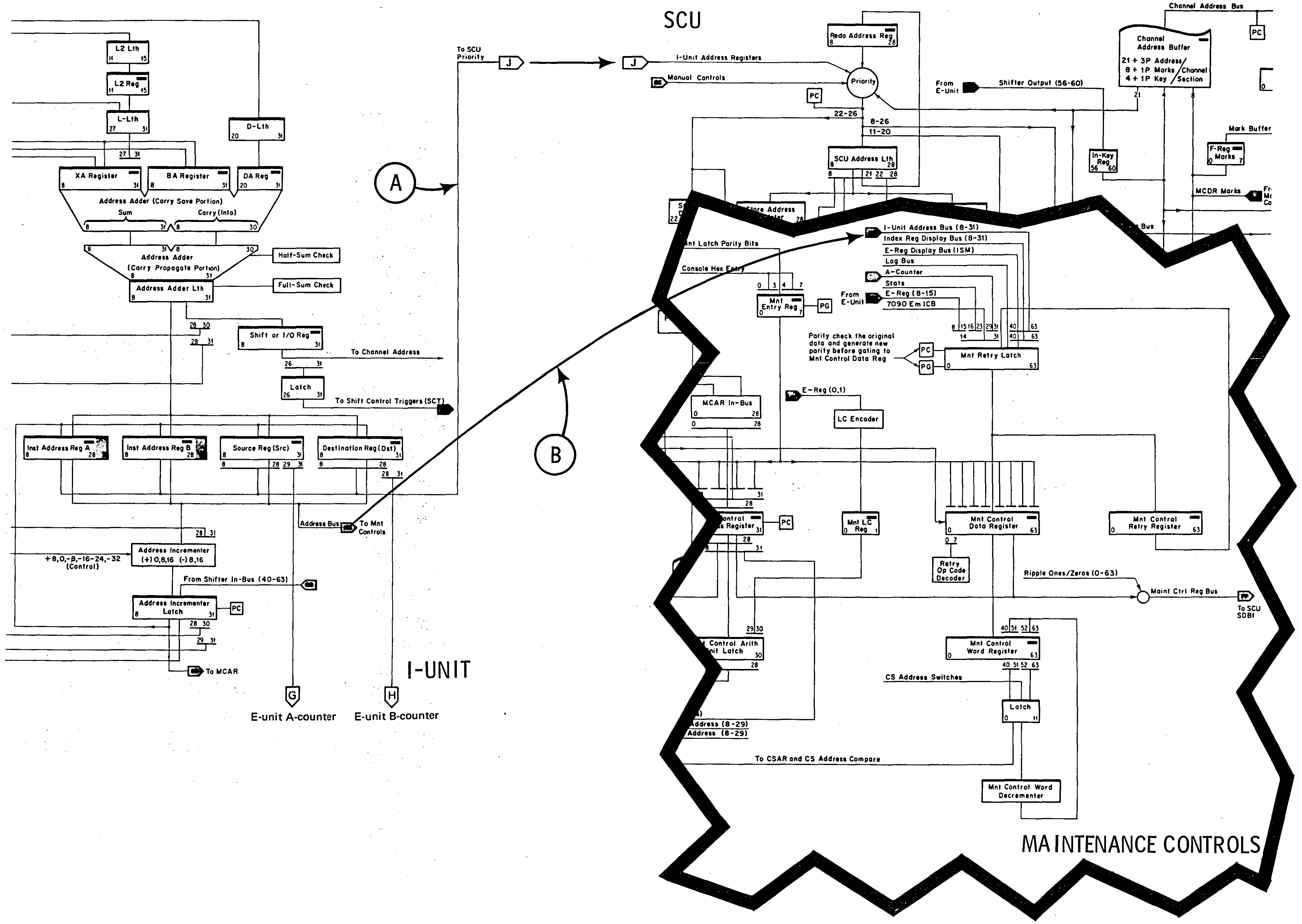


Figure 6-3. Address Register Bus

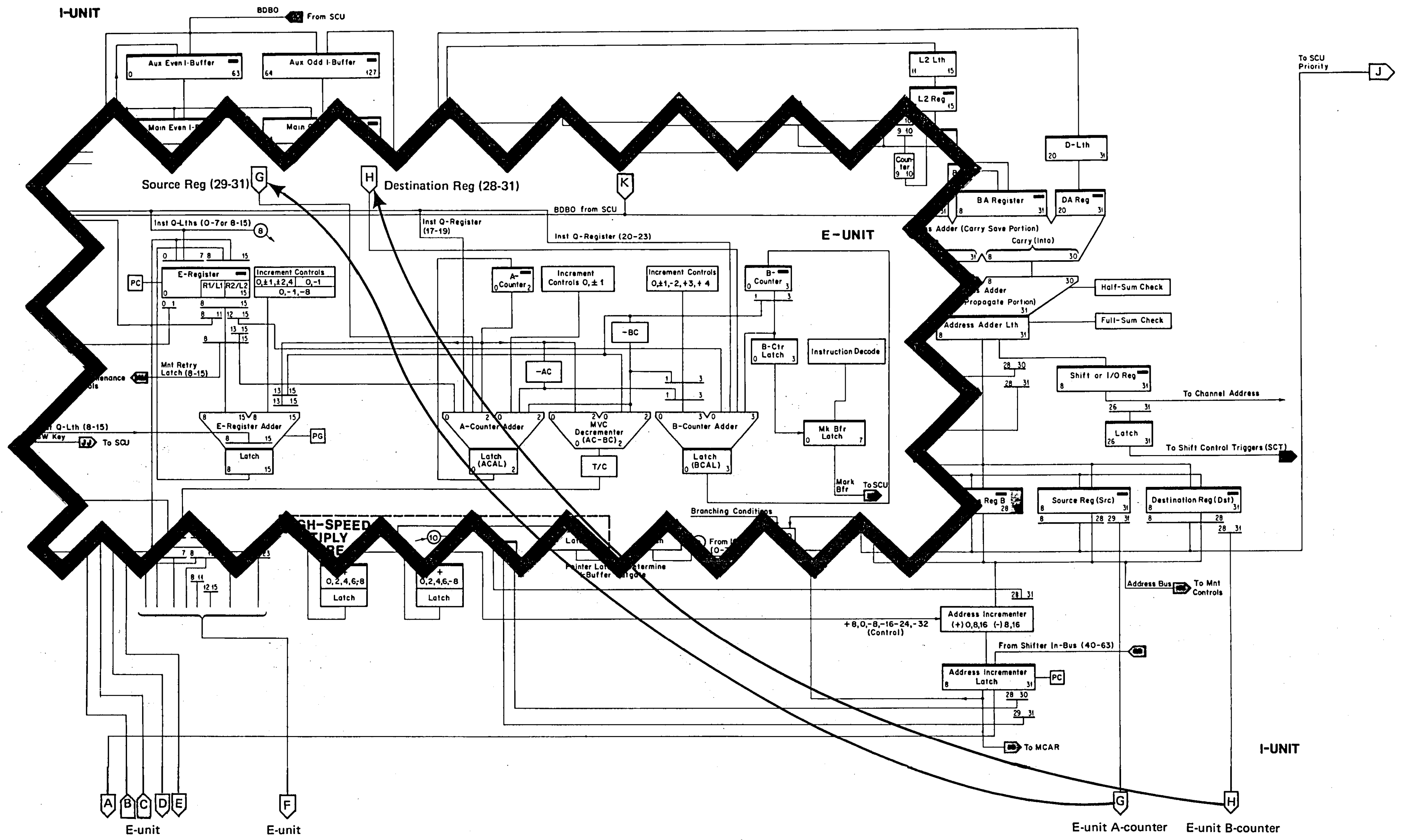


Figure 6-4. Address Register To E-Unit

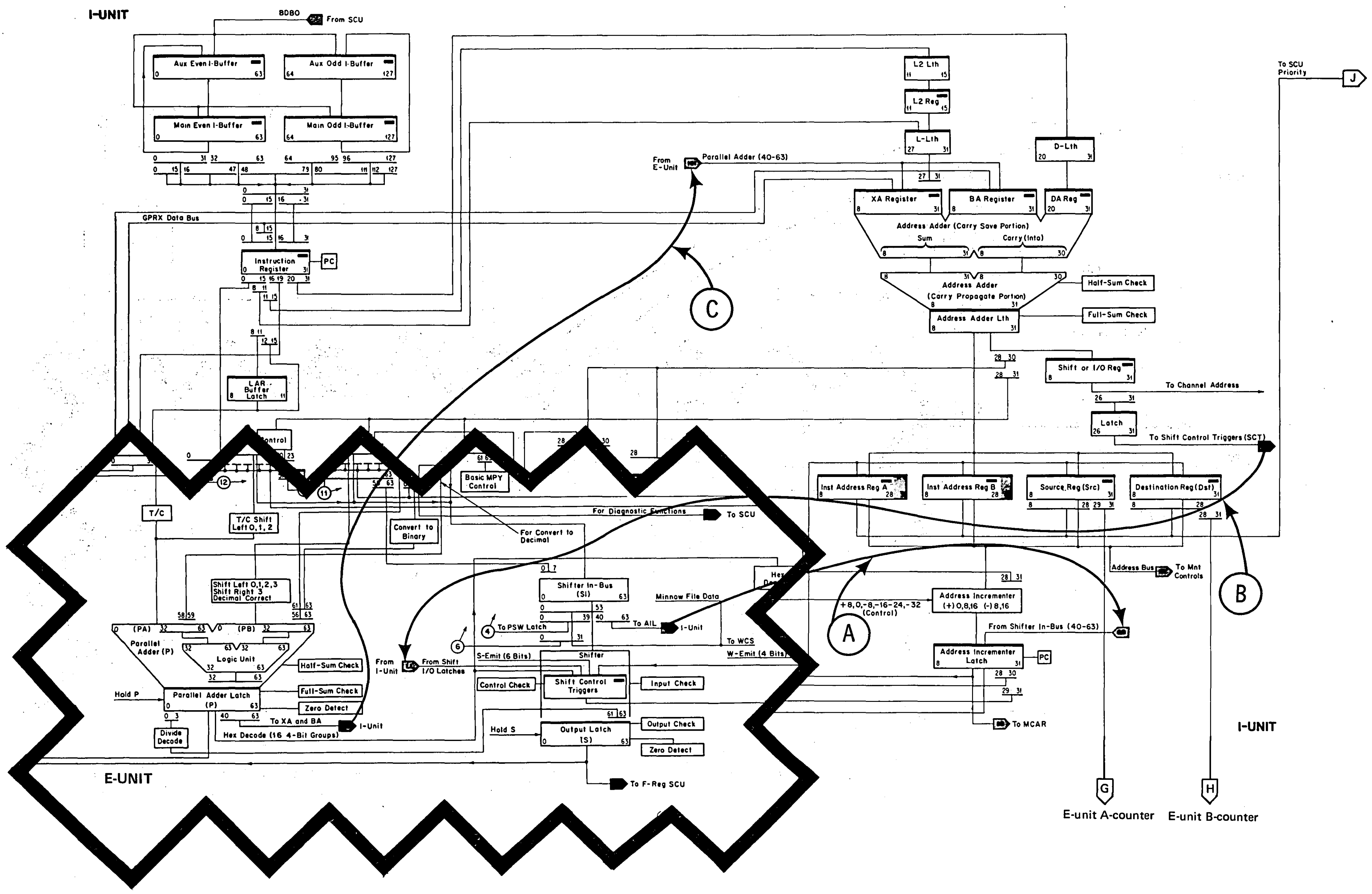


Figure 6-5. PAL, AIL, SCT

Because the I-unit preprocesses instructions, the need for a GPR in the I-unit can arise before that GPR is updated and available. For example, if the I-register contains an RX instruction, the address adder is not busy, the E-unit is executing an add instruction (RR) with an R1 field addressing GPR5, and the RX instruction X-field is also addressing GPR5, a GPR conflict condition exists. The RX instruction in this case must not be decoded until the needed GPR (5) has been changed by the E-unit. Otherwise, the wrong value will be obtained from GPR5. The needed delay is produced by blocking 'I is go'. The RX instruction remains in the I-register until the GPR is updated and a GPR conflict no longer exists.

When a GPR conflict delays the decoding of an instruction in the I-register, the total delay time can sometimes be reduced by one cycle by taking the address component from the parallel adder output during the GPR put-away cycle instead of waiting one more cycle for the information to be written in the GPR and read out again.

The normal sequence for the E-unit is to complete an execution, activate "end op," and then to write the result into the GPR for the specified R1 field on the following cycle. This sequence is not altered; but, by first detecting the GPR conflict and then waiting for "end op," the unavailable GPR contents will be available at the parallel adder during the "end op" cycle. Detection and obtaining the updated GPR contents from the parallel adder latch during the pre-fetching cycle is called address wraparound.

Figure 6-6. Address Wraparound

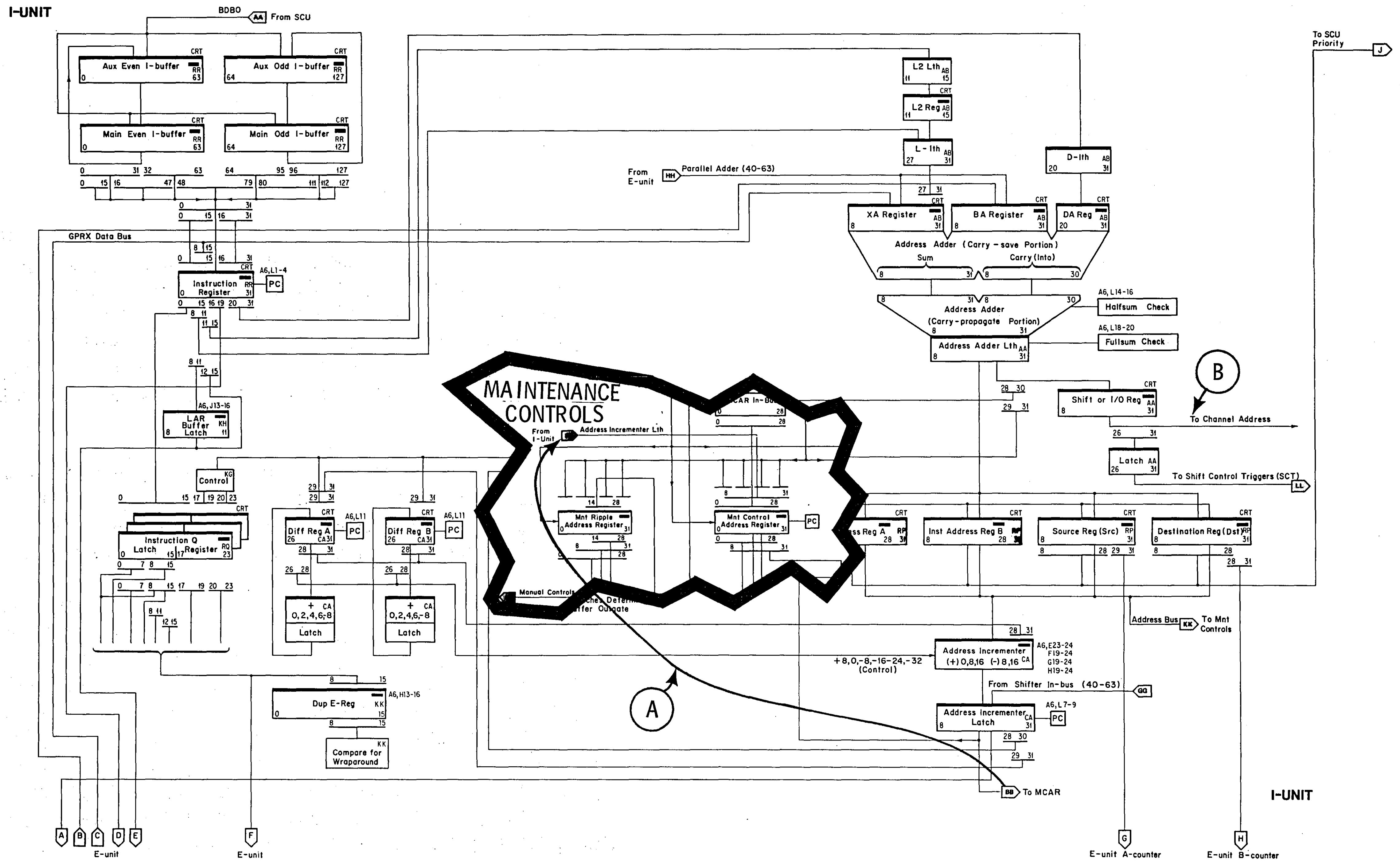


Figure 6-7. AIL To MCAR

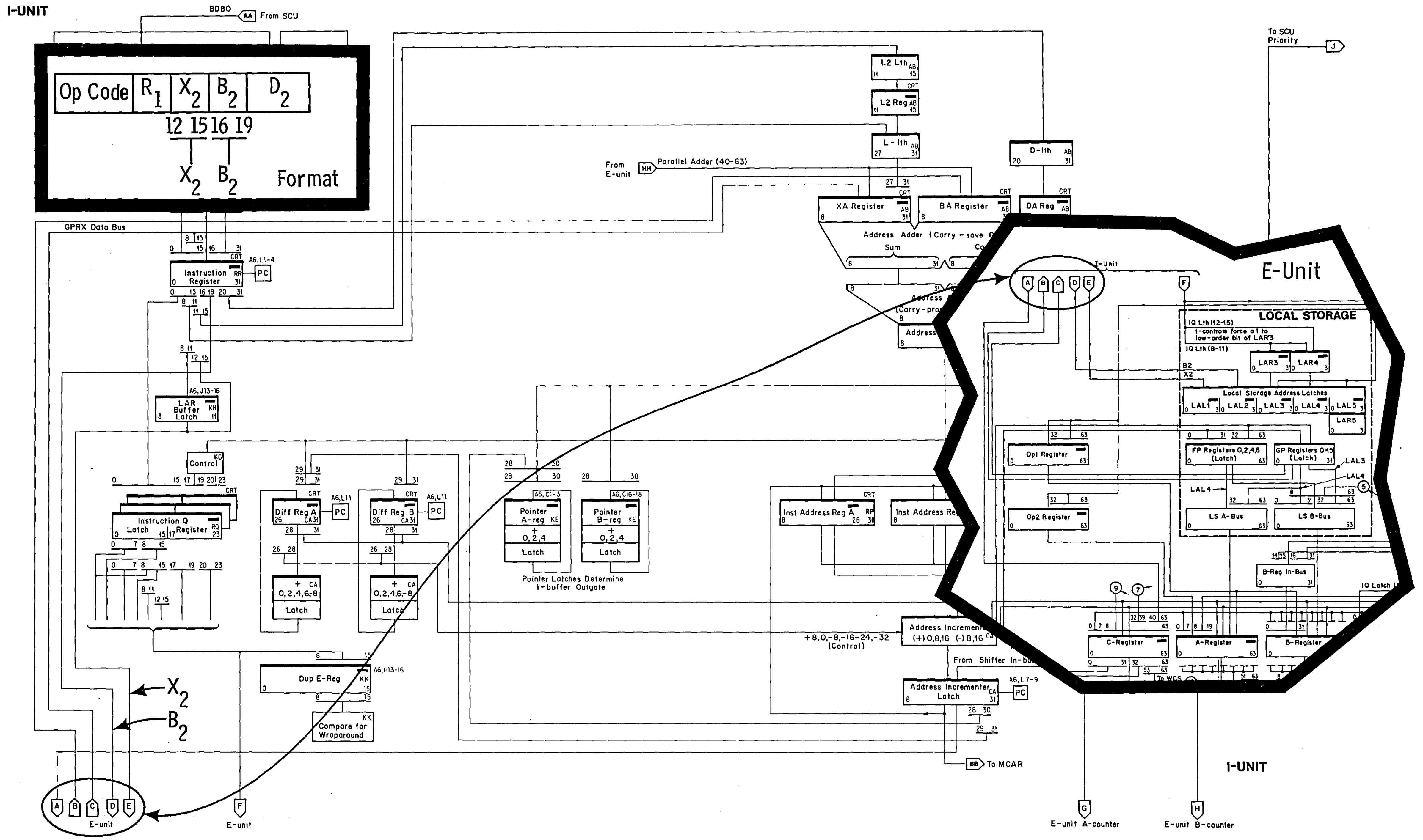


Figure 6-8. Local Store

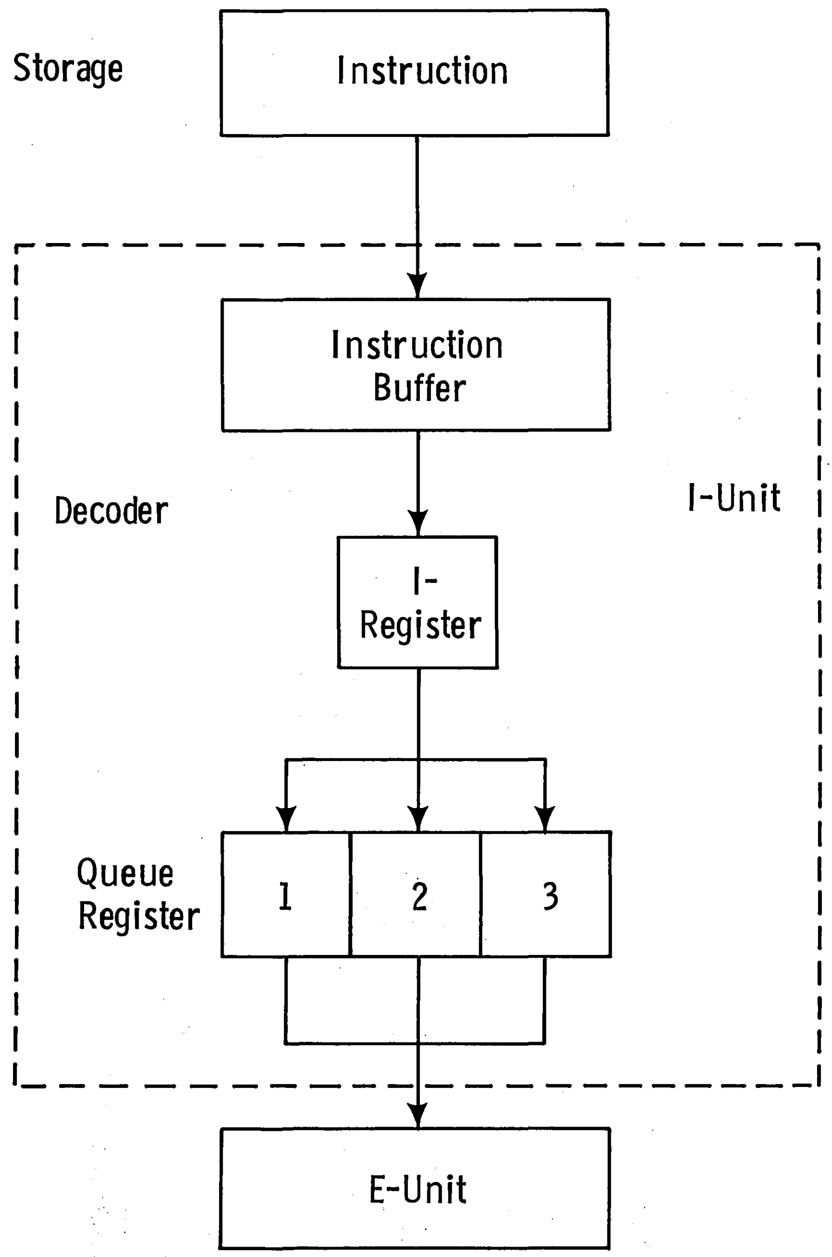
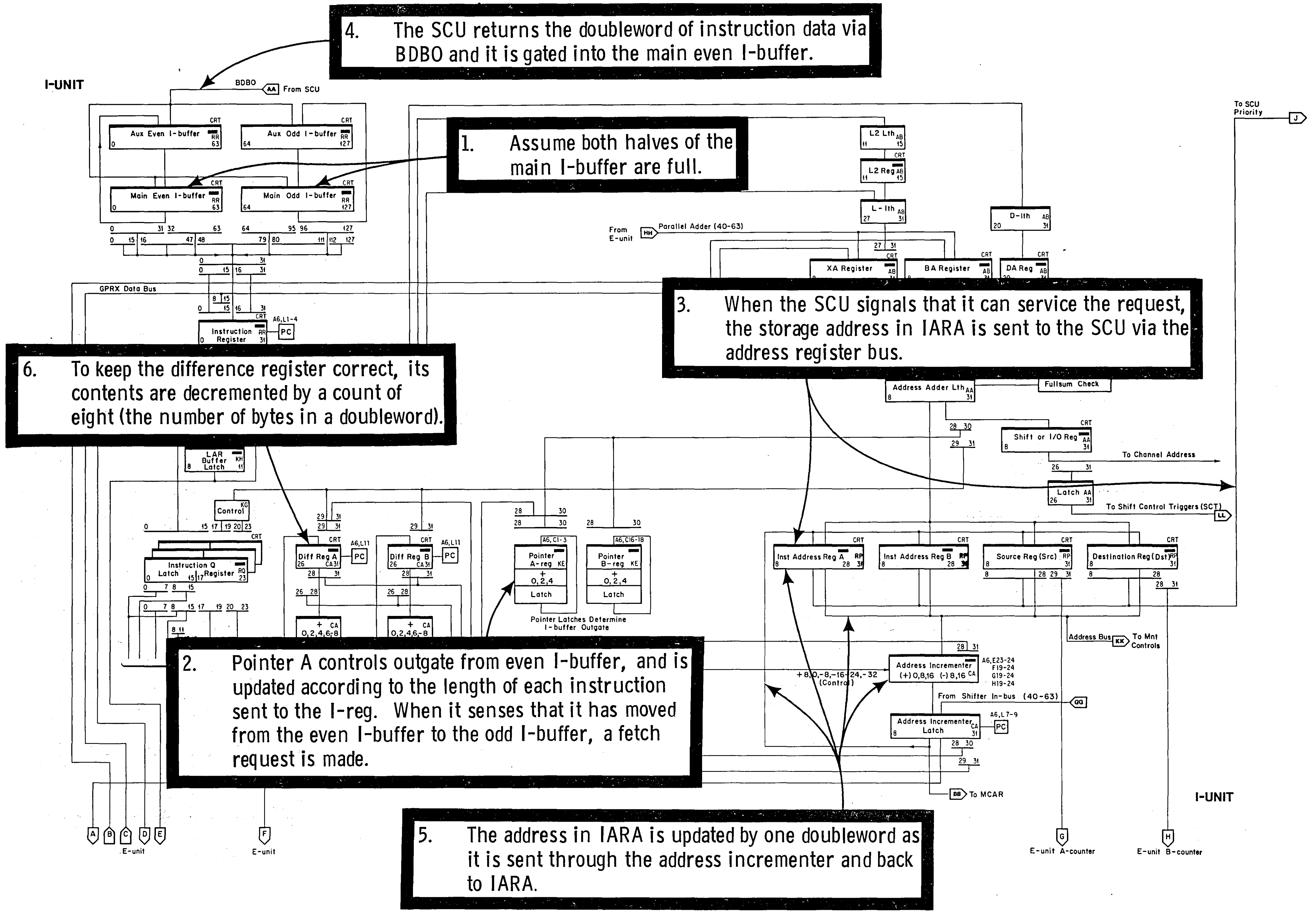


Figure 6-9. Simple I-Unit Data Flow

The 128-bit I-buffer temporarily stores instructions so that the I-register can have a supply of instructions to decode without having to wait for storage fetch operations. This is accomplished by fetching 16 bytes of instructions and gating them into the I-buffer. All requests are for doublewords (eight bytes) and the doubleword can be ingated to either half, even or odd, of the I-buffer. An attempt is made to keep the I-buffer ahead of instruction processing. Therefore, when half of an I-buffer is emptied by having all of its contents moved to the I-register, a storage request for more instruction stream data is initiated. A pointer register, in addition to controlling the gating of instructions from the I-buffer to the I-register, also indicates when a buffer half is empty.

The instruction address register contains the storage address of the instruction stream being fetched. The instruction address register is updated by the address incrementer.

Figure 6-10. I-Fetch Review



1. Assume both halves of the main I-buffer are full.

2. Pointer A controls outgate from even I-buffer, and is updated according to the length of each instruction sent to the I-reg. When it senses that it has moved from the even I-buffer to the odd I-buffer, a fetch request is made.

3. When the SCU signals that it can service the request, the storage address in IARA is sent to the SCU via the address register bus.

4. The SCU returns the doubleword of instruction data via BDBO and it is gated into the main even I-buffer.

5. The address in IARA is updated by one doubleword as it is sent through the address incrementer and back to IARA.

6. To keep the difference register correct, its contents are decremented by a count of eight (the number of bytes in a doubleword).

Figure 6-11. I-Fetch

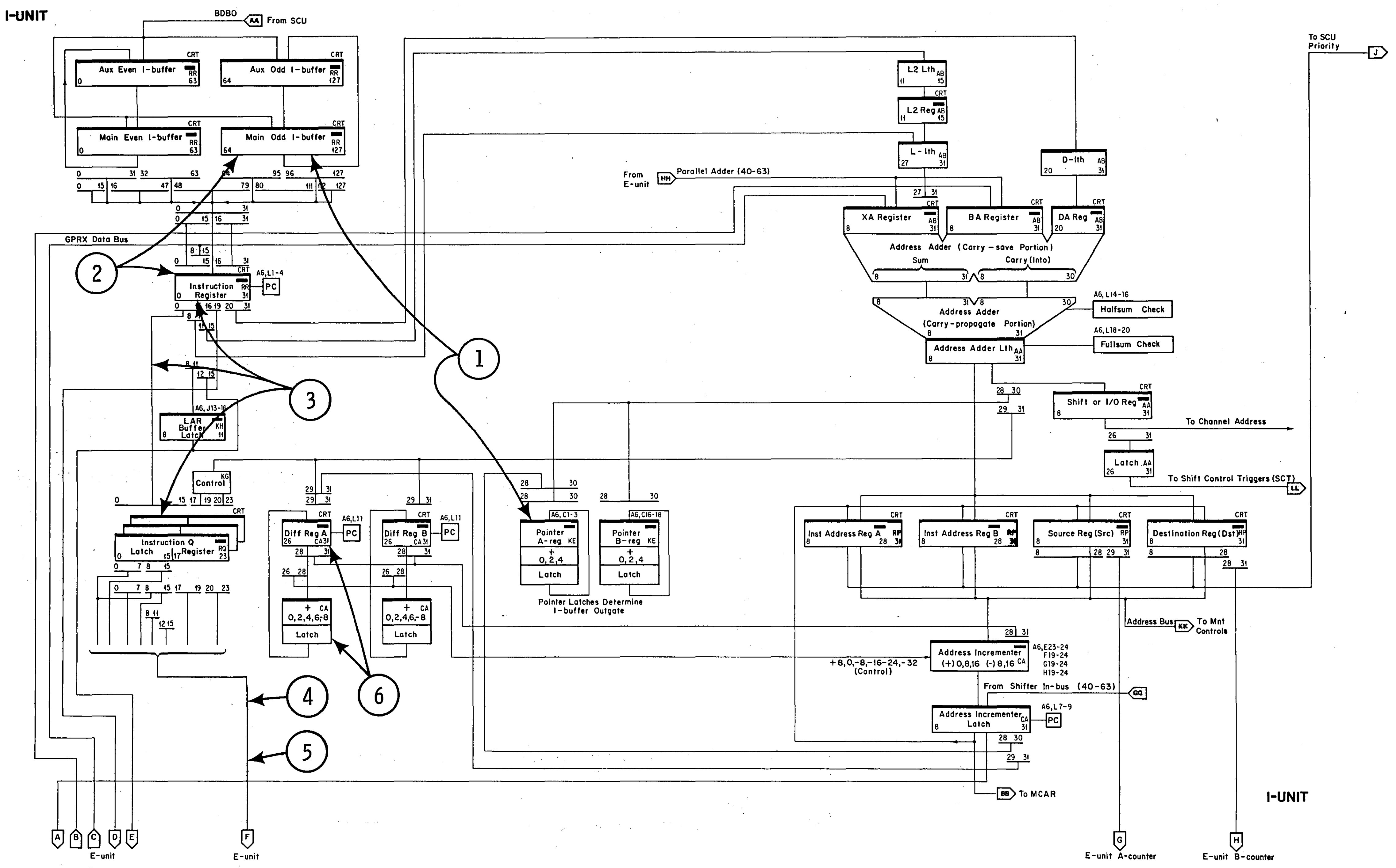


Figure 6-12. RR Add

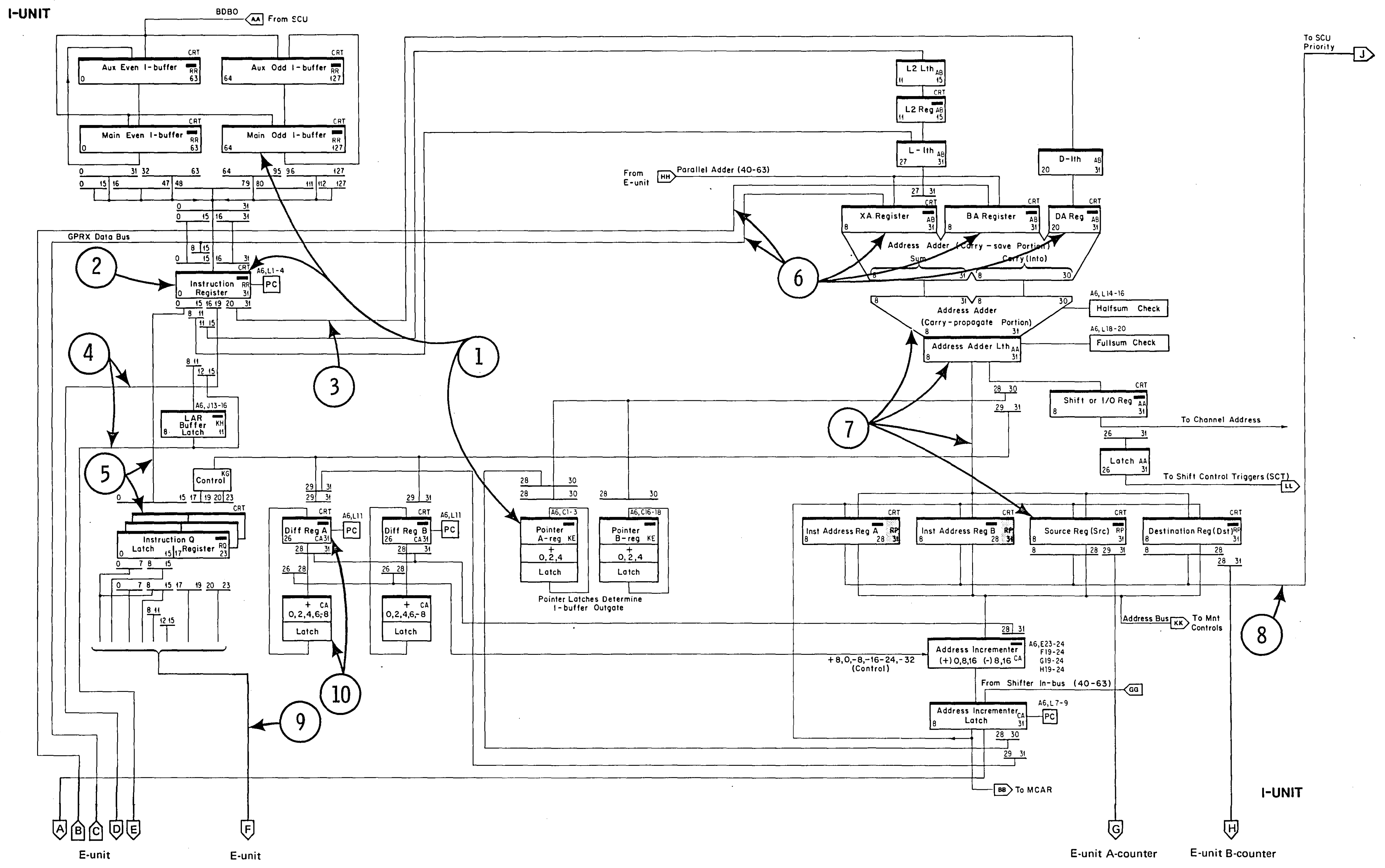


Figure 6-13. RX Add

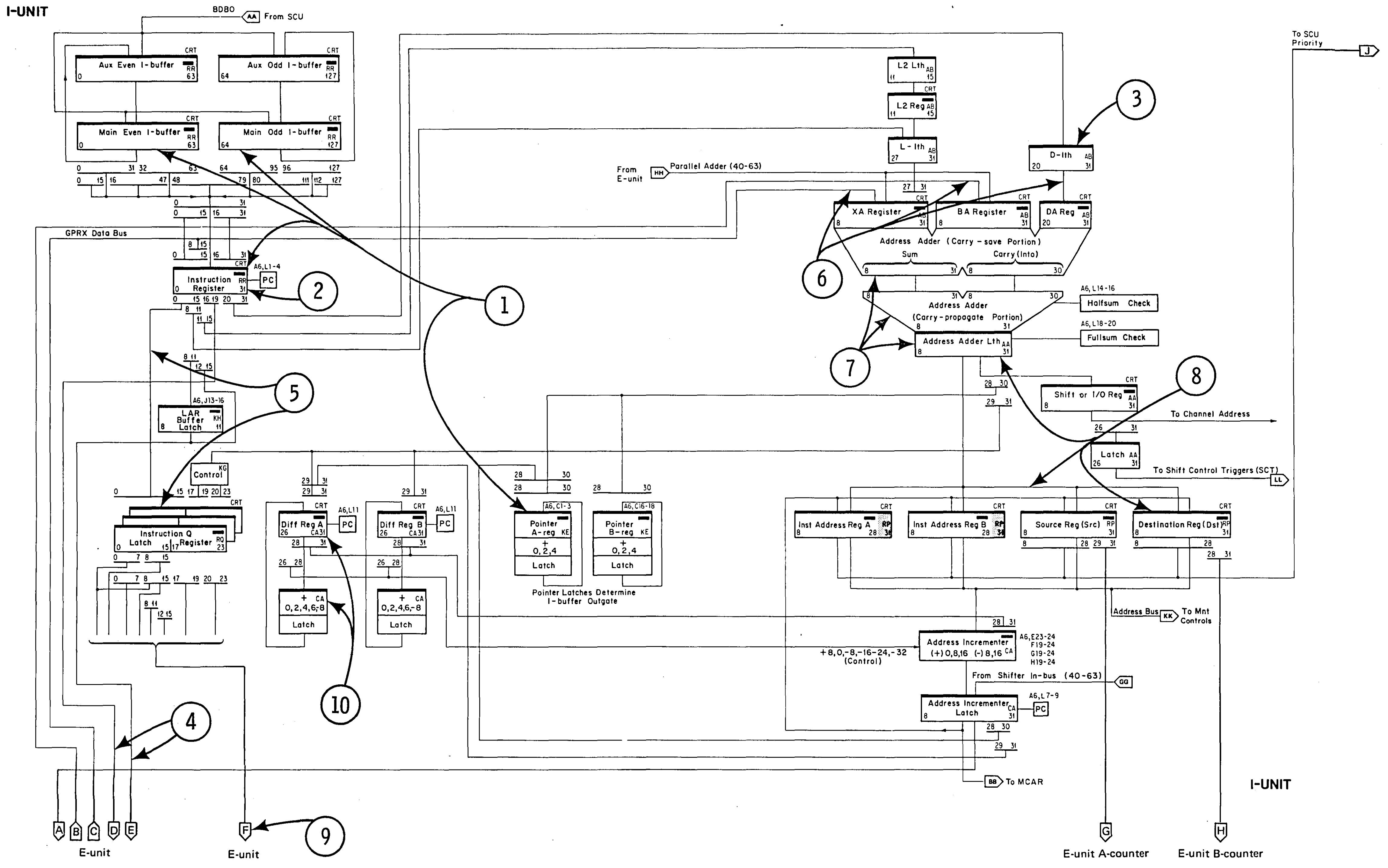


Figure 6-14. RX Store

Operand Store Compare

An operand store compare condition exists when a fetch instruction is decoded before a store instruction is completed. A store fetch sequence can cause a problem because of the way the I-unit processes fetch instructions and the way that the E-unit processes store instructions.

For example, if a fetch follows closely behind a store, it is possible for the fetch to be processed out of program sequence before the store. This can happen because the I-unit generates the address for a store but the E-unit initiates the request to SCU when the data is ready to be stored. While the E-unit is processing previously decoded instructions, the I-unit could decode a fetch instruction. Because the request to the SCU for a fetch instruction is made at the beginning of the address calculation cycle, the fetch request to the SCU could be processed before the store request. This does not cause any problem except when the fetch and store requests are made to the same main storage address. Then special action is needed.

Because this condition can exist, the source and destination registers are compared each time a fetch instruction is decoded before a store instruction is completed. If fetch address equals the store address, the fetch instruction is ignored until the store instruction has been completed. After the store is finished, a new request is made for the fetch.

The I-unit controls check for this condition and provide the control necessary to prevent a fetch instruction from receiving old information.

Figure 6-15. Operand Store Compare

PROGRAM STORE COMPARE

Because the I-unit preprocesses instructions, a store instruction could alter an instruction that the I-unit has already fetched. This situation could cause incorrect program execution. Therefore, to prevent the E-unit from receiving the wrong instruction, a test is made by the I-unit controls. The controls cause a compare circuit to compare the address in the destination register with the instruction address register (IARA). If the destination register equals IARA or IARA + 16 or IARA + 32, a program store compare condition is signaled.

A program store compare condition (PSC) causes all further processing of instructions to stop until the store instruction has finished. The I-unit is reset, the address of the instruction immediately following the store is calculated, and then instruction processing is restarted.

Figure 6-16. Program Store Compare

SS FORMAT INSTRUCTIONS

Multiple decode and address calculations cycles are needed. For decoding and address calculations, the I-unit handles an SS format instructions as two or three instructions with multiple decode cycles (three for logical and two for decimal) and multiple address calculation cycles. The I-unit generates both initial operand addresses and fetches a doubleword operand from each address. For logical instructions, base and displacement are added to produce an address that points to the leftmost byte of the field.

WORD OVERLAP TEST (WOT)

I-unit tests for word overlap condition for SS logical instruction. The I-unit also tests logical instructions to determine whether the E-unit is to use an alternate execution microprogram routine because of overlapping operand fields. An overlapped operand field exists when all or a portion of the first and second operands reside in the same address in main storage. For example, if an SS logical instruction specifies address 1005 for the first operand, a length field of five, and an address of 1009 for the second operand, there is an overlap of one byte in address 1009.

Figure 6-17. SS Instruction

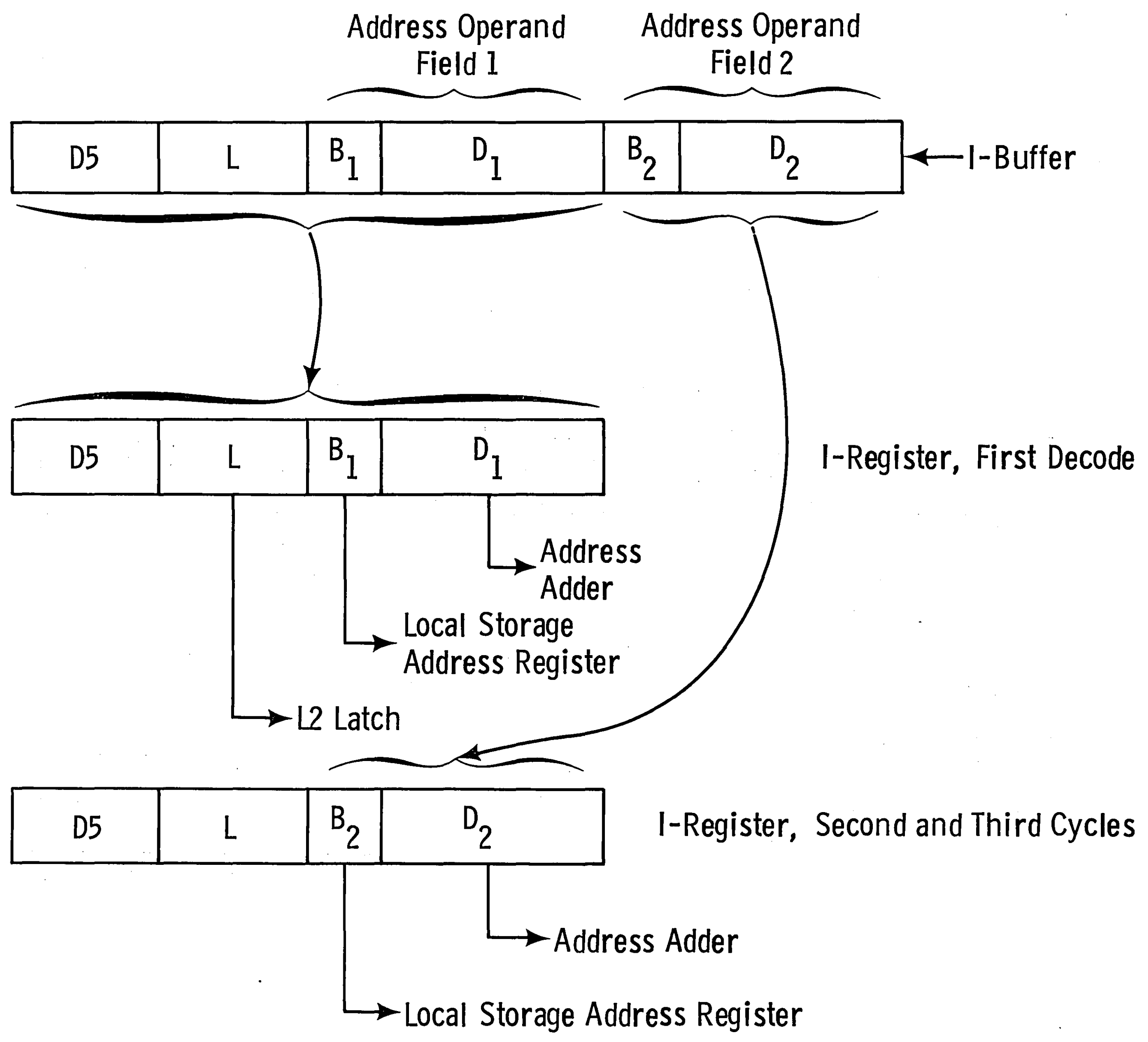


Figure 6-18. I-Register

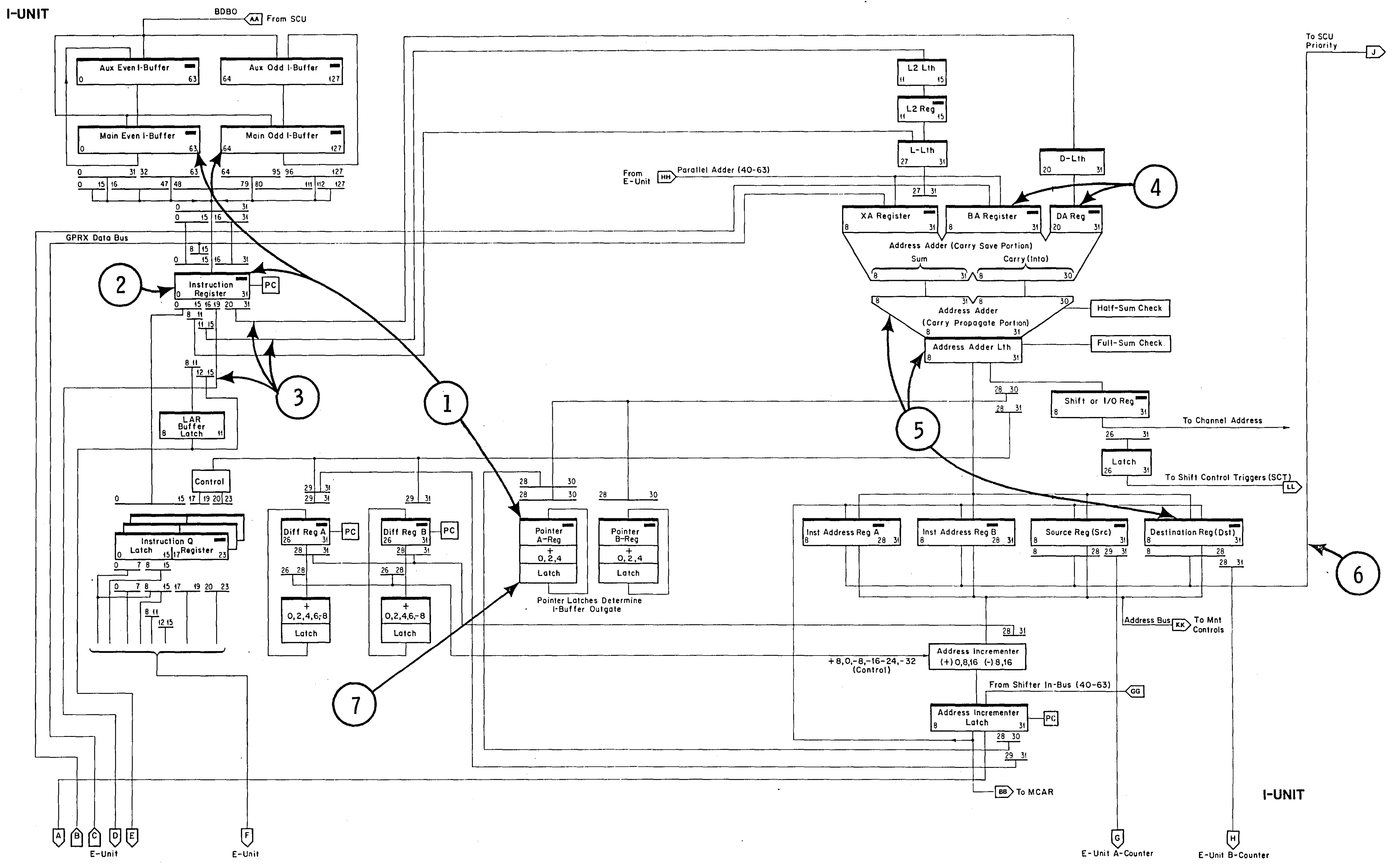


Figure 6-19. First Cycle, SS Instruction

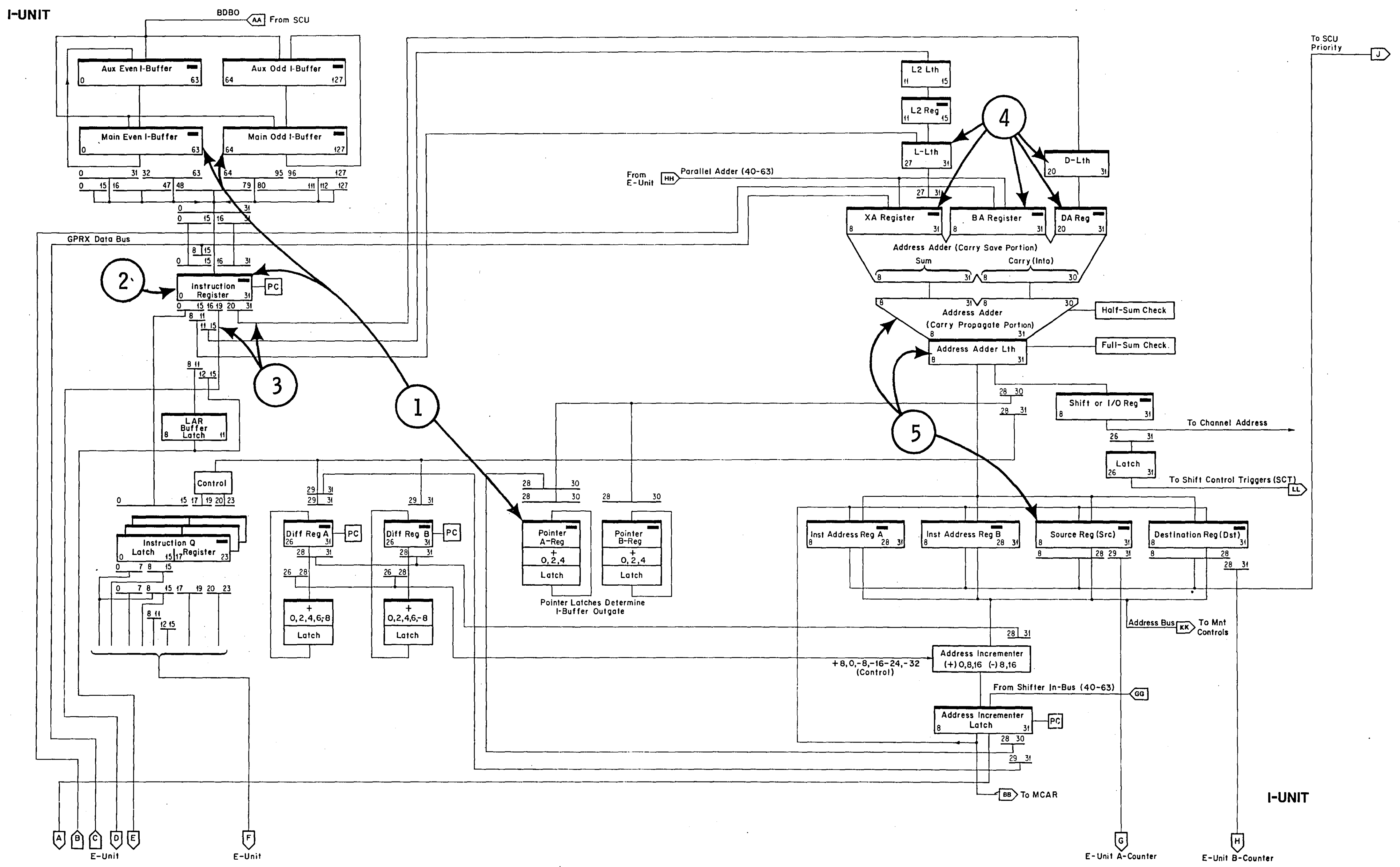


Figure 6-20. 2nd Cycle, SS Instruction

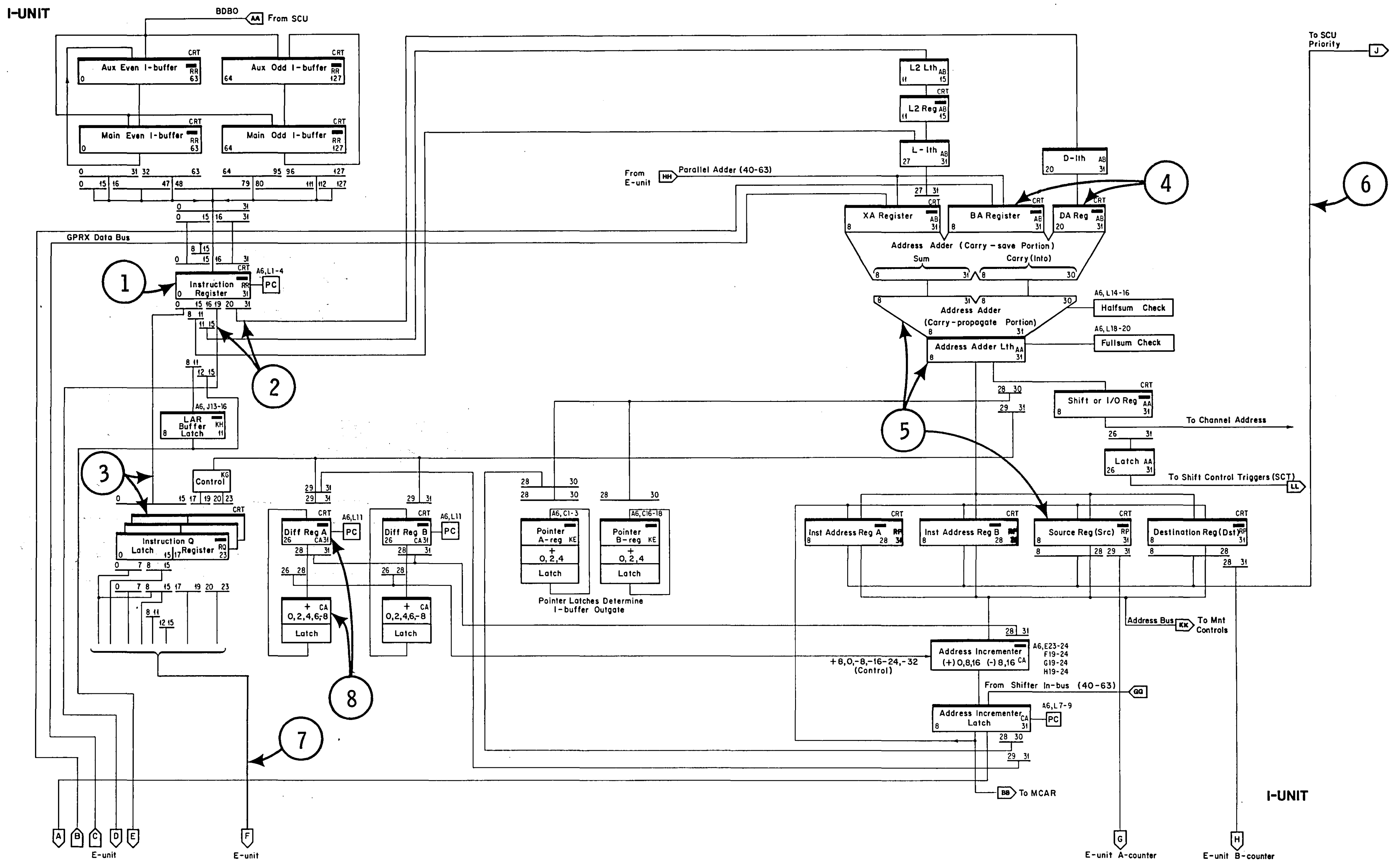


Figure 6-21. 3rd Cycle, SS Instruction

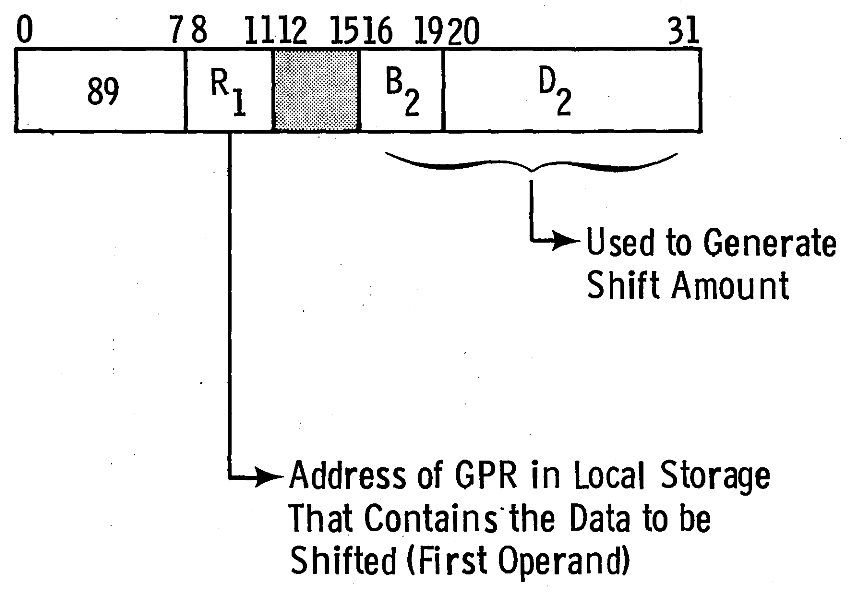

0
7 8
11 12
15 16
19 20
31
89
R_1
B_2
D_2
Used to Generate Shift Amount
Address of GPR in Local Storage That Contains the Data to be Shifted (First Operand)

Figure 6-22. RS Shift Format (SLL)

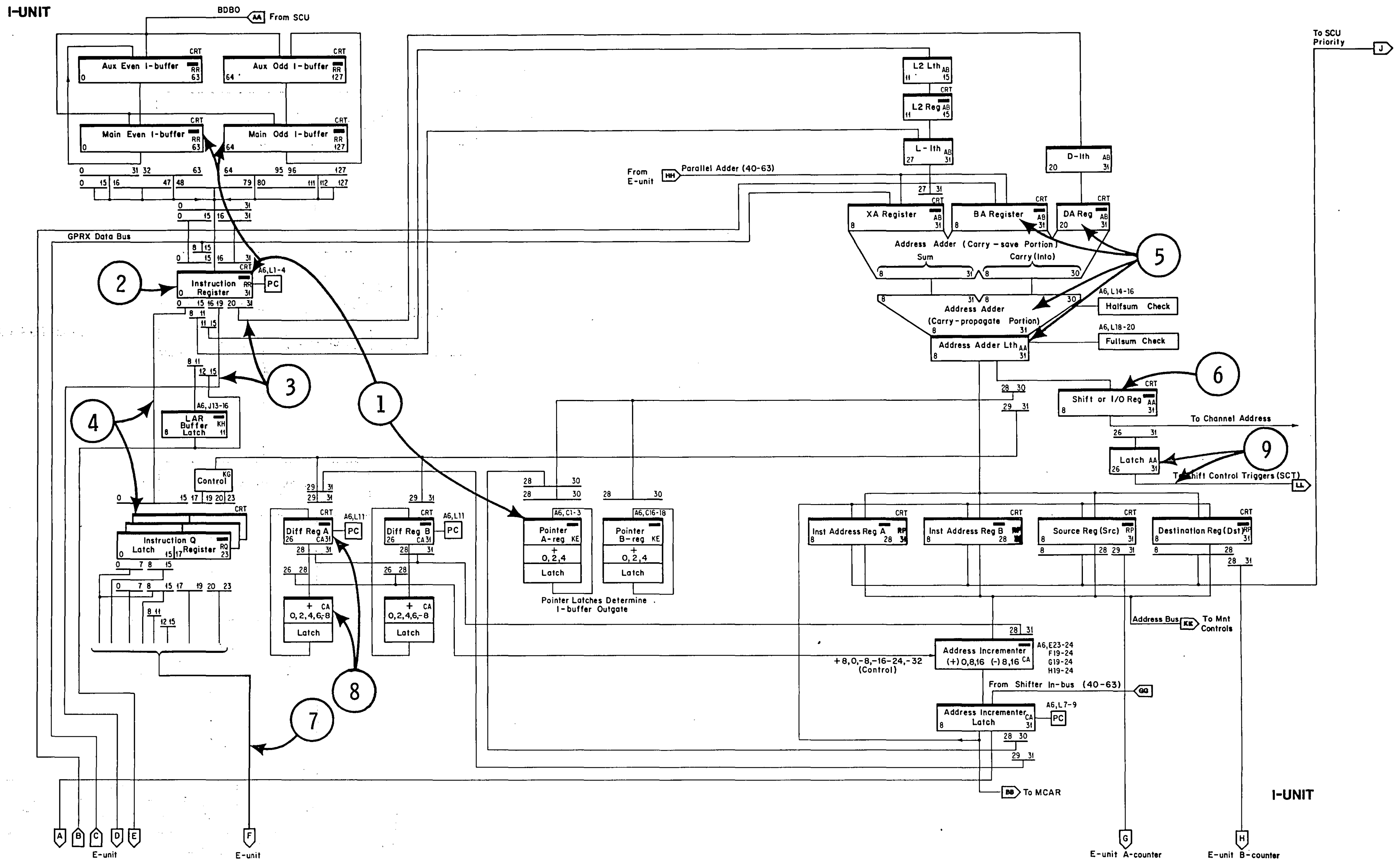


Figure 6-23. RS Shift

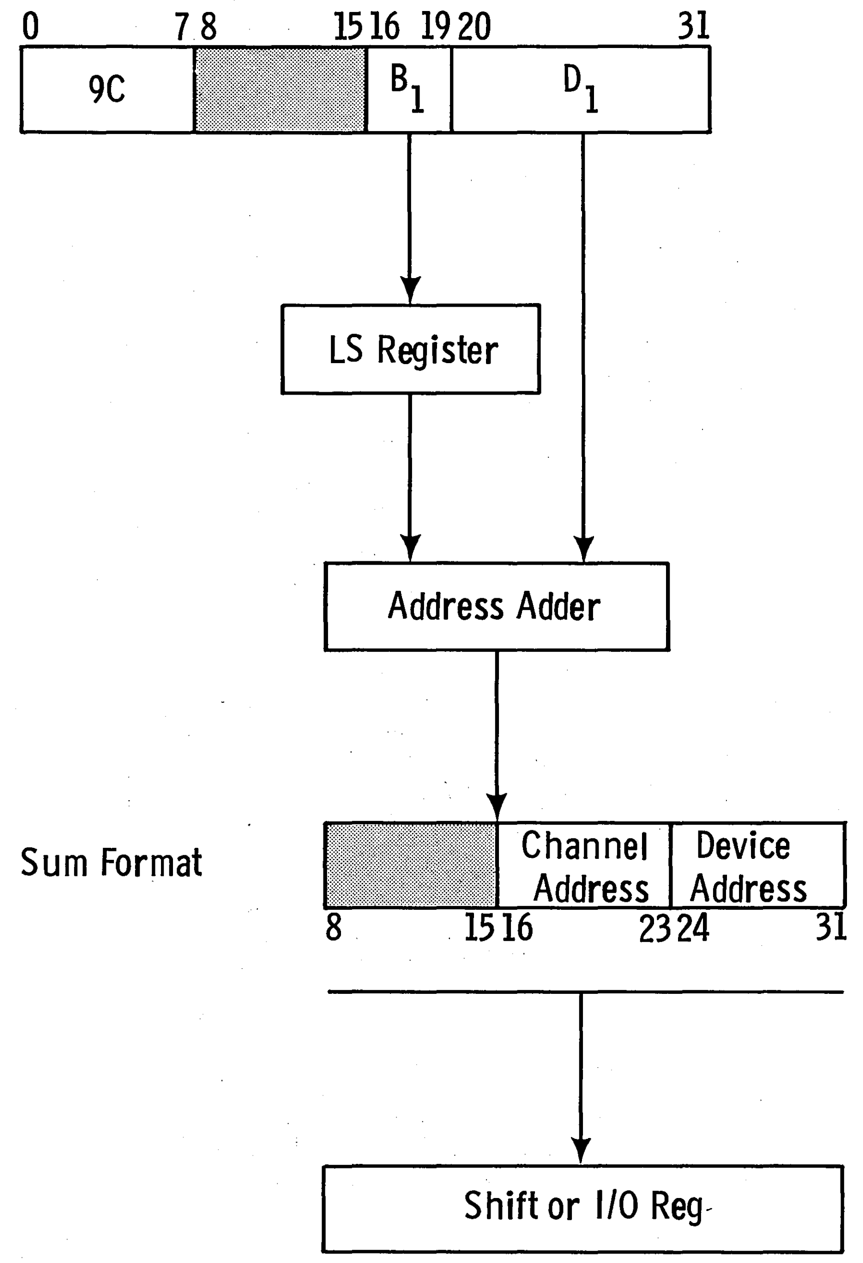


Figure 6-24. Start I/O

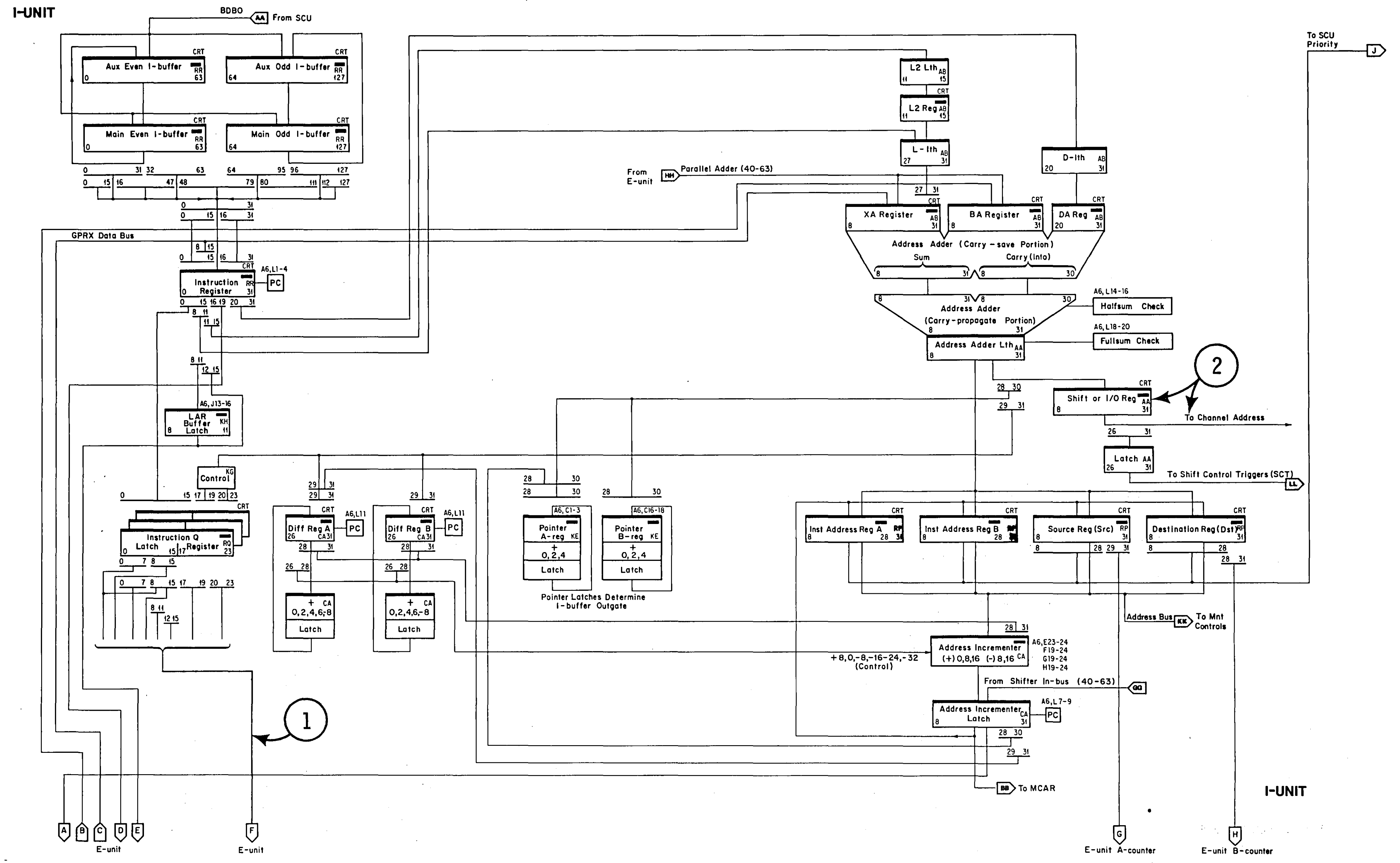


Figure 6-25. Start I/O

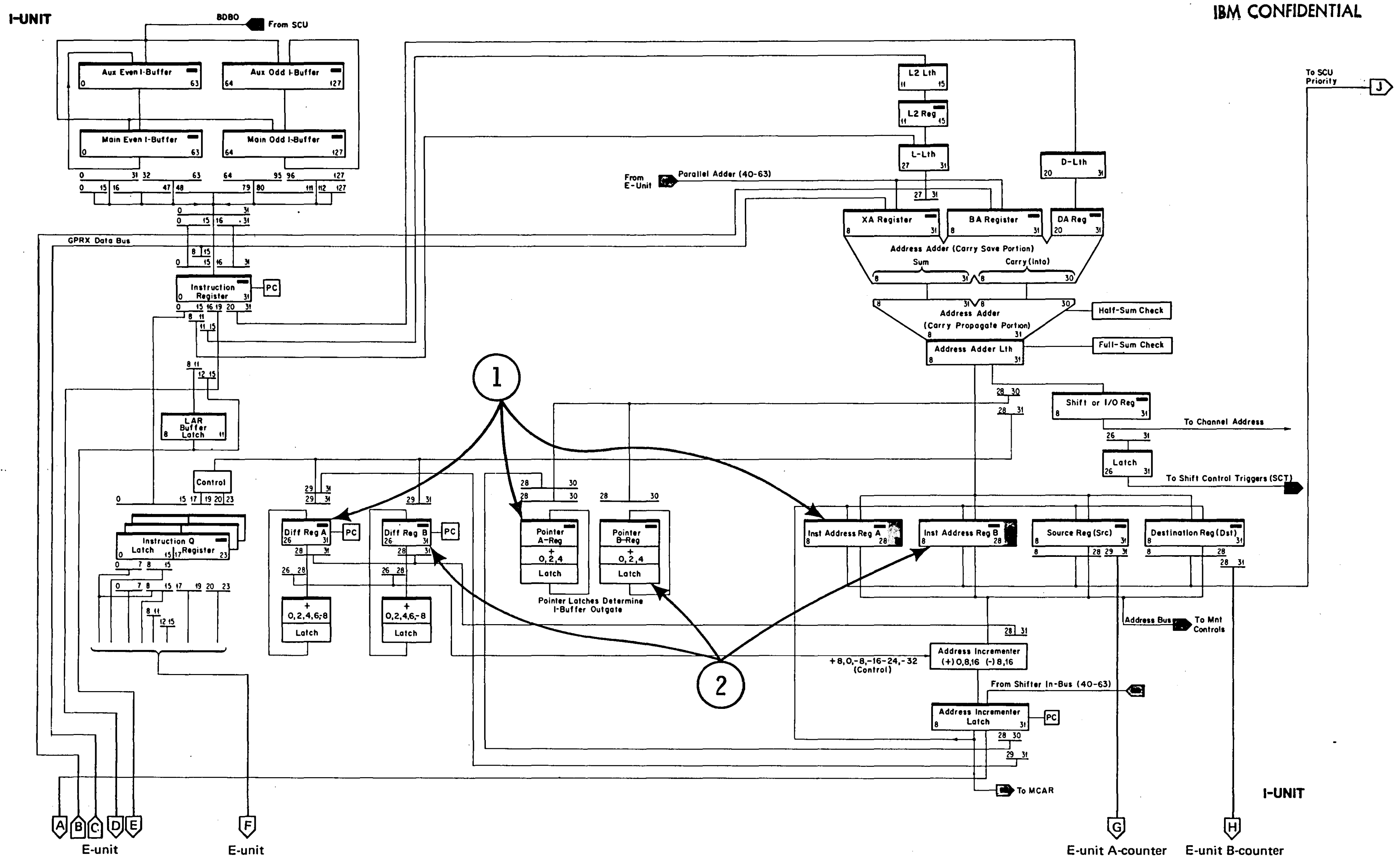


Figure 6-26. IARA And IARB

I-UNIT BRANCH INSTRUCTION PROCESSING

Special action is required when a branch instruction is decoded in the I-unit. When the address is calculated for a branch instruction, it is put into the instruction address register that is not currently active. For example, if IAR A is currently being used to fetch instructions for the main instruction buffer, the branch target address is put IAR B. An instruction fetch is then made per IAR B. The doubleword of instructions obtained by this fetch is part of the target instruction stream. Conversely the instructions obtained by a fetch per the instruction address register currently being used are part of the normal instruction stream. The I-unit makes an estimate as to the probable success of the branch by decoding the branch instruction op code, the branch mask, and the R2 field of the instruction, if applicable. (A successful branch is one in which the branch actually takes place during E-unit execution; conversely, an unsuccessful branch is one in which the branch does not occur when the branch is executed.)

Figure 6-27. I-Unit Branch Instruction Processing

BRANCH ESTIMATES

Unsuccessful branch estimates are made for:

Branch on condition (BC) when the mask bit is not equal to 0 or 15.
Branch on condition (BCR) when the mask bit is not equal to 0 or 15 and R2 is not equal to 0.

Successful branch estimates are made for:

Branch on condition (BC) when the mask bit equals 15.
Branch on condition (BCR) when the mask bit equals 15 and R2 is not equal to 0.
Branch and link (BAL).
Branch and link (BALR) when R2 is not equal to 0.
Branch on count (BCT)
Branch on count (BCTR) when R2 is not equal to 0.
Branch on index low or equal (BXLE).
Branch on index high (BXH).

The following instructions are not regarded as branch instructions by branch controls.

Branch on condition (BC) when the mask bit equals 0.
Branch on condition (BCR) when the mask bit or R2 equals 0.
Branch and link (BALR) when R2 equals 0.
Branch on count (BCTR) when R2 equals 0.

Figure 6-28. Branch Estimates

UNSUCCESSFUL BRANCH ESTIMATES

When a branch instruction is decoded that requires an unsuccessful branch estimate, the I-unit generates the target address and initiates a fetch request for the target stream of instructions. When the target fetch returns, the I-unit places the target instructions in the auxiliary buffer. Also, the I-unit continues processing the instructions following the branch instruction in the old instruction path without stopping.

Sometime later, the E-unit sends the result of the branch instruction back to the I-unit. If the decision is not to branch (I-unit estimate was correct), it is only necessary to rescind any outstanding requests for the target stream and to continue processing.

However, if the E-unit determines that a branch is necessary (I-unit estimate was incorrect), the I-unit must back up and process the target stream instead of the old instruction stream. The I-unit is reset to cancel any instructions that may have already been sent to a Q-register and the target instruction stream that was fetched and placed in the auxiliary buffer is moved to the main buffer. Instruction processing then resumes in the target stream.

Figure 6-29. Unsuccessful Estimate

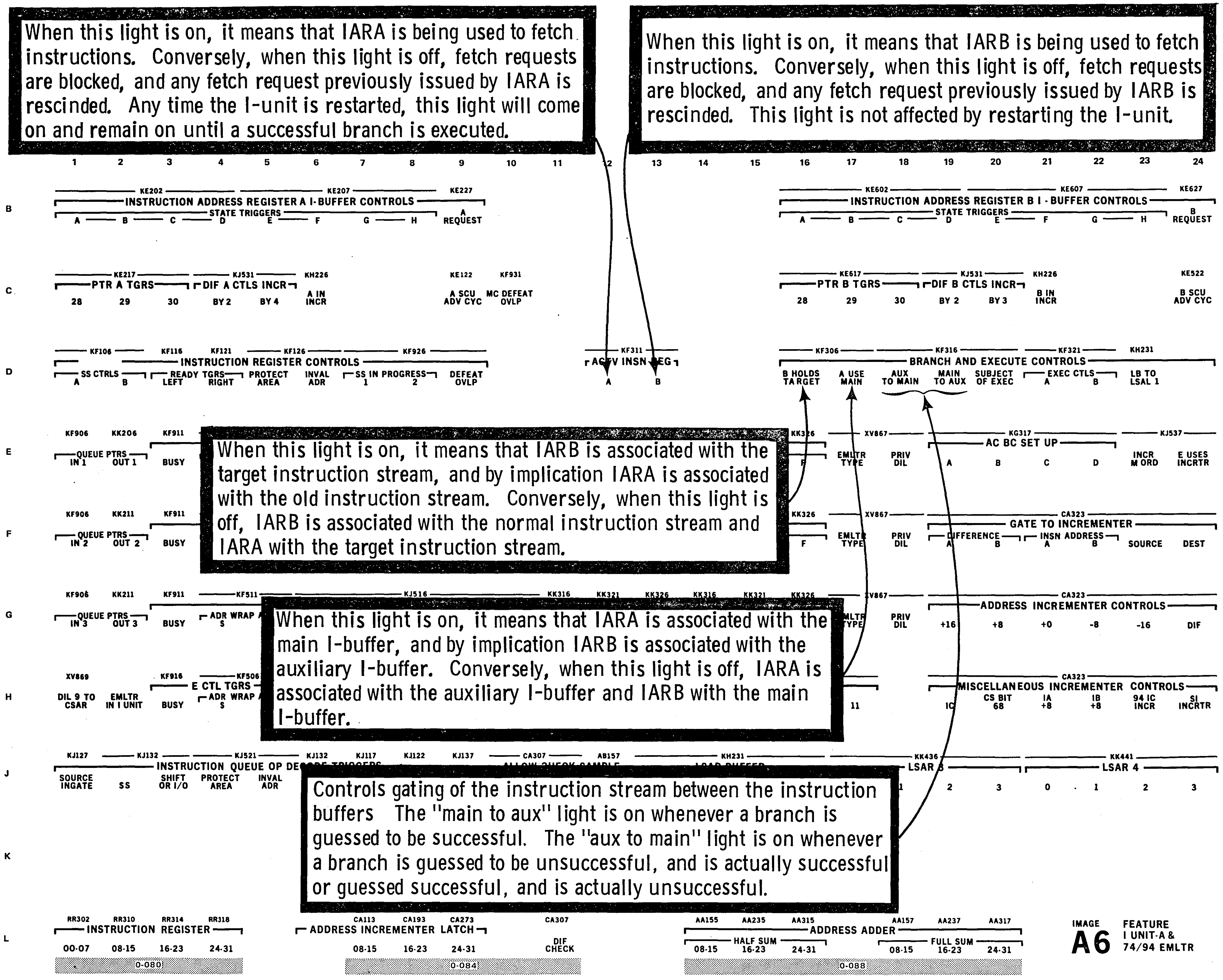
SUCCESSFUL BRANCH ESTIMATES

When a branch instruction is estimated to be successful, the target address is generated and a fetch request is initiated in the same manner as for unsuccessful estimates. I-unit instruction processing must wait until the target stream is actually available. The old instruction stream of instructions is moved to the auxiliary buffer. When the target stream instructions return, they are placed in the main buffer and instruction processing continues.

Sometime later, the E-unit advises the I-unit of the outcome of the branch instruction. If the estimate was correct, the I-unit has correctly started processing in the target stream and it is only necessary to rescind any outstanding request for the old instruction instruction stream.

However, if a successful estimate branch is actually unsuccessful, then the I-unit must be reset to cancel any instructions processed in the target stream. The old instruction stream must be returned to the main buffer from the auxiliary buffer and then processing is resumed in the old instruction stream.

Figure 6-30. Successful Estimates



When this light is on, it means that IARA is being used to fetch instructions. Conversely, when this light is off, fetch requests are blocked, and any fetch request previously issued by IARA is rescinded. Any time the I-unit is restarted, this light will come on and remain on until a successful branch is executed.

When this light is on, it means that IARB is being used to fetch instructions. Conversely, when this light is off, fetch requests are blocked, and any fetch request previously issued by IARB is rescinded. This light is not affected by restarting the I-unit.

When this light is on, it means that IARB is associated with the target instruction stream, and by implication IARA is associated with the old instruction stream. Conversely, when this light is off, IARB is associated with the normal instruction stream and IARA with the target instruction stream.

When this light is on, it means that IARA is associated with the main I-buffer, and by implication IARB is associated with the auxiliary I-buffer. Conversely, when this light is off, IARA is associated with the auxiliary I-buffer and IARB with the main I-buffer.

Controls gating of the instruction stream between the instruction buffers The "main to aux" light is on whenever a branch is guessed to be successful. The "aux to main" light is on whenever a branch is guessed to be unsuccessful, and is actually successful or guessed successful, and is actually unsuccessful.

Figure 6-31. MFI Branch Indicators

E-UNIT SETUP

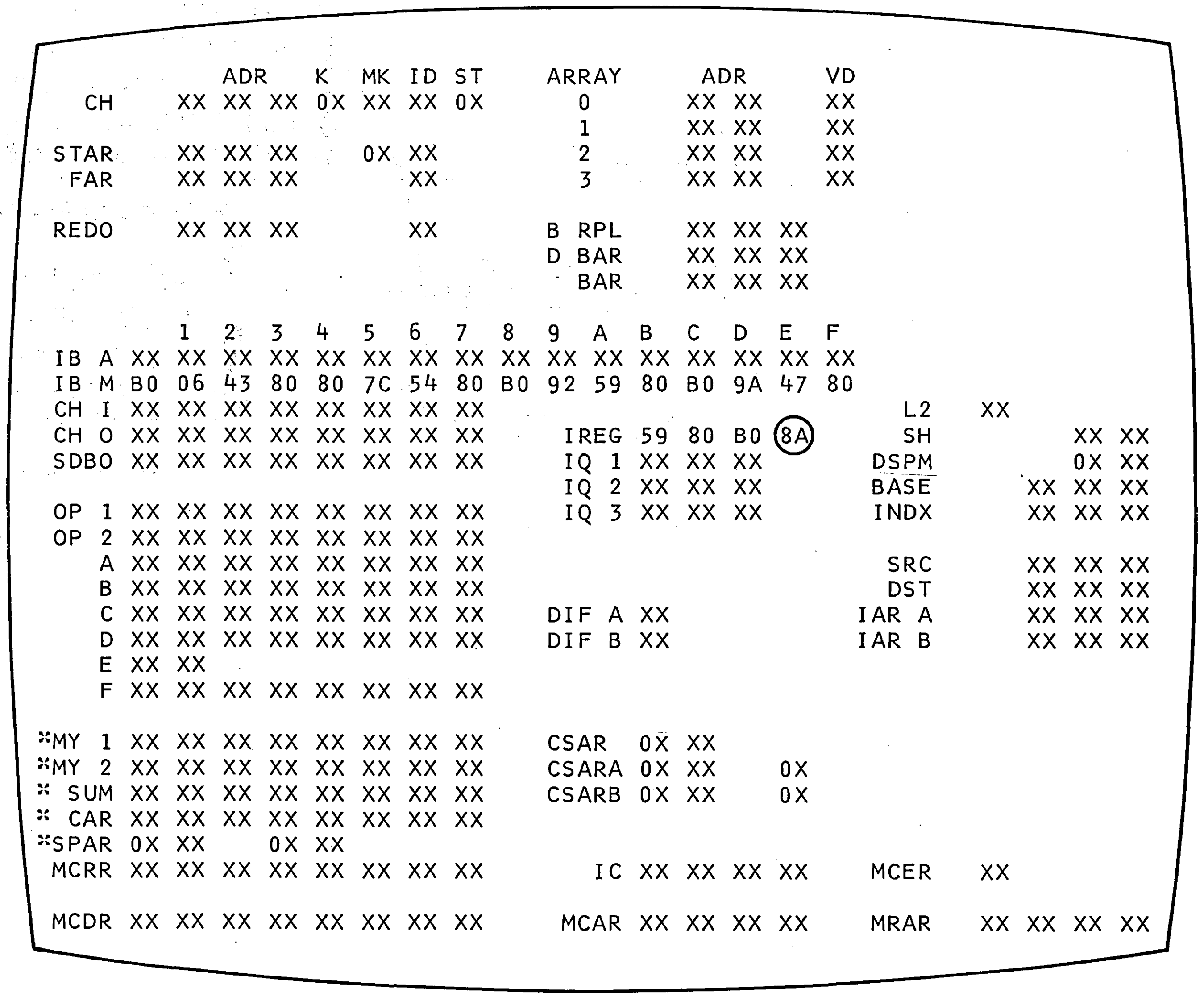
When an instruction is transferred from the I-unit to the E-unit, the E-unit registers must be set up with the values needed to begin the execution of that instruction. Setup includes placing the instruction in the E-register, fetching operands from local storage, ingating operands from the operand buffers (the operand buffers hold the operands fetched from storage by the I-unit), and sending starting values to ACAL and BCAL. Also, the op code must be put into the control storage address register (CSAR to cause a CS branch to the first word of the microprogram routine that will control the E-unit during the execution of this instruction. The instruction queue register in the I-unit must be released for use by another instruction.

The conditions for transferring a new instruction to the E-unit are:

1. The I-unit has an instruction ready.
2. The E-unit has completed execution of the last instruction.

When the I-unit has an instruction ready for the E-unit, the queue out-pointer gates the op code to the CSAR in-bus, and the 'instruction ready' line is turned on. When the end-op bit and "instruction ready" line are both on, the op code (Q-register bits 0-7) is ingated to CSAR, and the 'op branch taken' trigger is turned on. The "op branch taken" trigger causes most of the setup information to be transferred from the I-unit to the E-unit.

Figure 6-32. E-Unit Setup



```
            ADR   K  MK ID ST    ARRAY     ADR     VD
   CH    XX XX XX 0X XX XX 0X      0      XX XX    XX
                                   1      XX XX    XX
 STAR    XX XX XX    0X XX         2      XX XX    XX
  FAR    XX XX XX       XX         3      XX XX    XX

 REDO    XX XX XX       XX       B RPL    XX XX XX
                                 D BAR    XX XX XX
                                   BAR    XX XX XX

          1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
 IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
 IB M B0 06 43 80 80 7C 54 80 B0 92 59 80 B0 9A 47 80
 CH I XX XX XX XX XX XX XX XX                          L2    XX
 CH O XX XX XX XX XX XX XX XX     IREG 59 80 B0 (8A)    SH          XX XX
 SDBO XX XX XX XX XX XX XX XX     IQ 1 XX XX XX       DSPM          0X XX
                                  IQ 2 XX XX XX       BASE       XX XX XX
 OP 1 XX XX XX XX XX XX XX XX     IQ 3 XX XX XX       INDX       XX XX XX
 OP 2 XX XX XX XX XX XX XX XX
    A XX XX XX XX XX XX XX XX                          SRC       XX XX XX
    B XX XX XX XX XX XX XX XX                          DST       XX XX XX
    C XX XX XX XX XX XX XX XX    DIF A XX            IAR A       XX XX XX
    D XX XX XX XX XX XX XX XX    DIF B XX            IAR B       XX XX XX
    E XX XX
    F XX XX XX XX XX XX XX XX

∷MY 1 XX XX XX XX XX XX XX XX    CSAR  0X XX
∷MY 2 XX XX XX XX XX XX XX XX    CSARA 0X XX    0X
∷ SUM XX XX XX XX XX XX XX XX    CSARB 0X XX    0X
∷ CAR XX XX XX XX XX XX XX XX
∷SPAR 0X XX    0X XX
 MCRR XX XX XX XX XX XX XX XX       IC XX XX XX XX    MCER    XX

 MCDR XX XX XX XX XX XX XX XX     MCAR XX XX XX XX    MRAR    XX XX XX XX
```

◯ = FLASHING

Figure 7-1. CRT For Session 7, Question 1

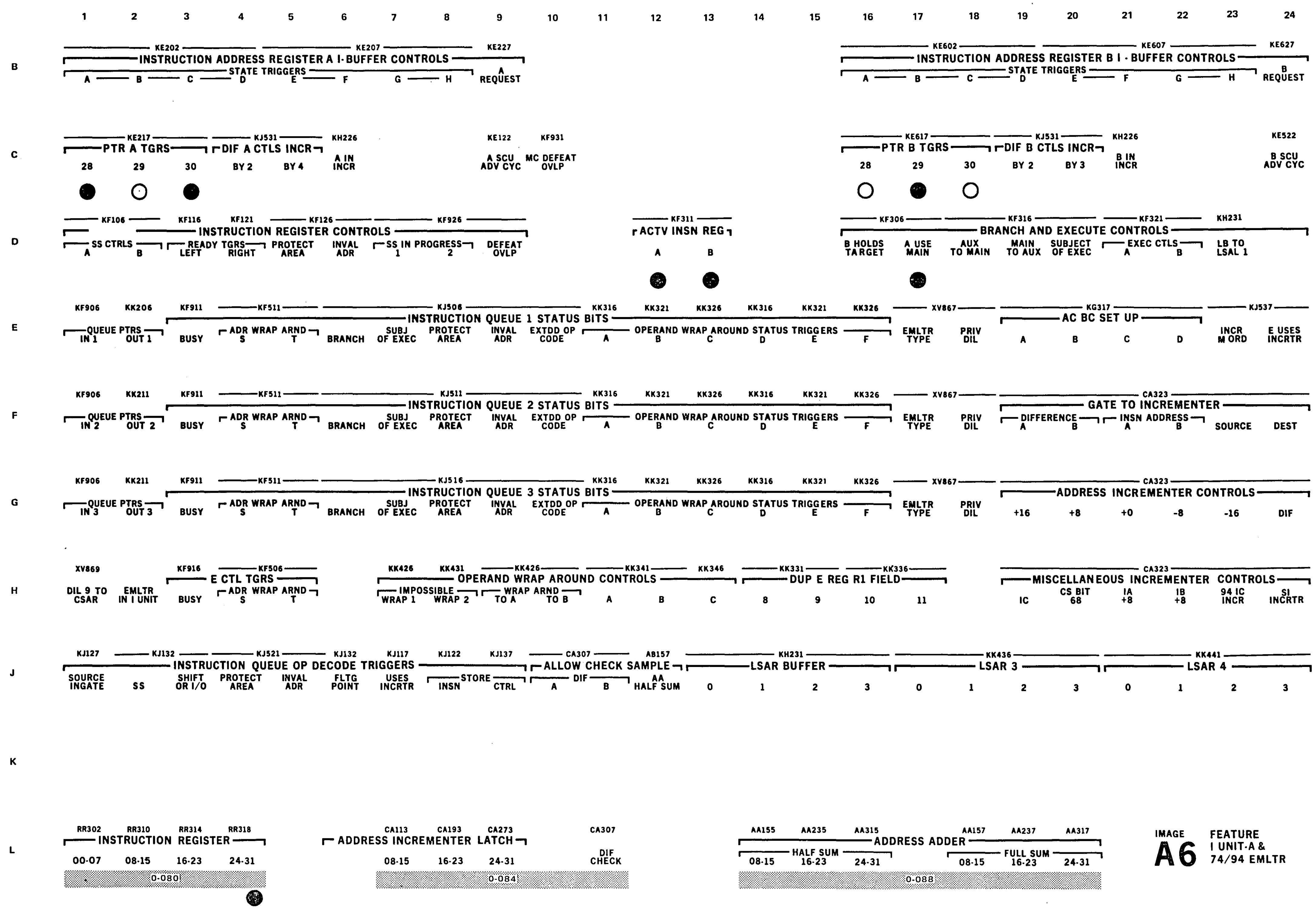


Figure 7-2. Image A6 For Session 7, Question 1

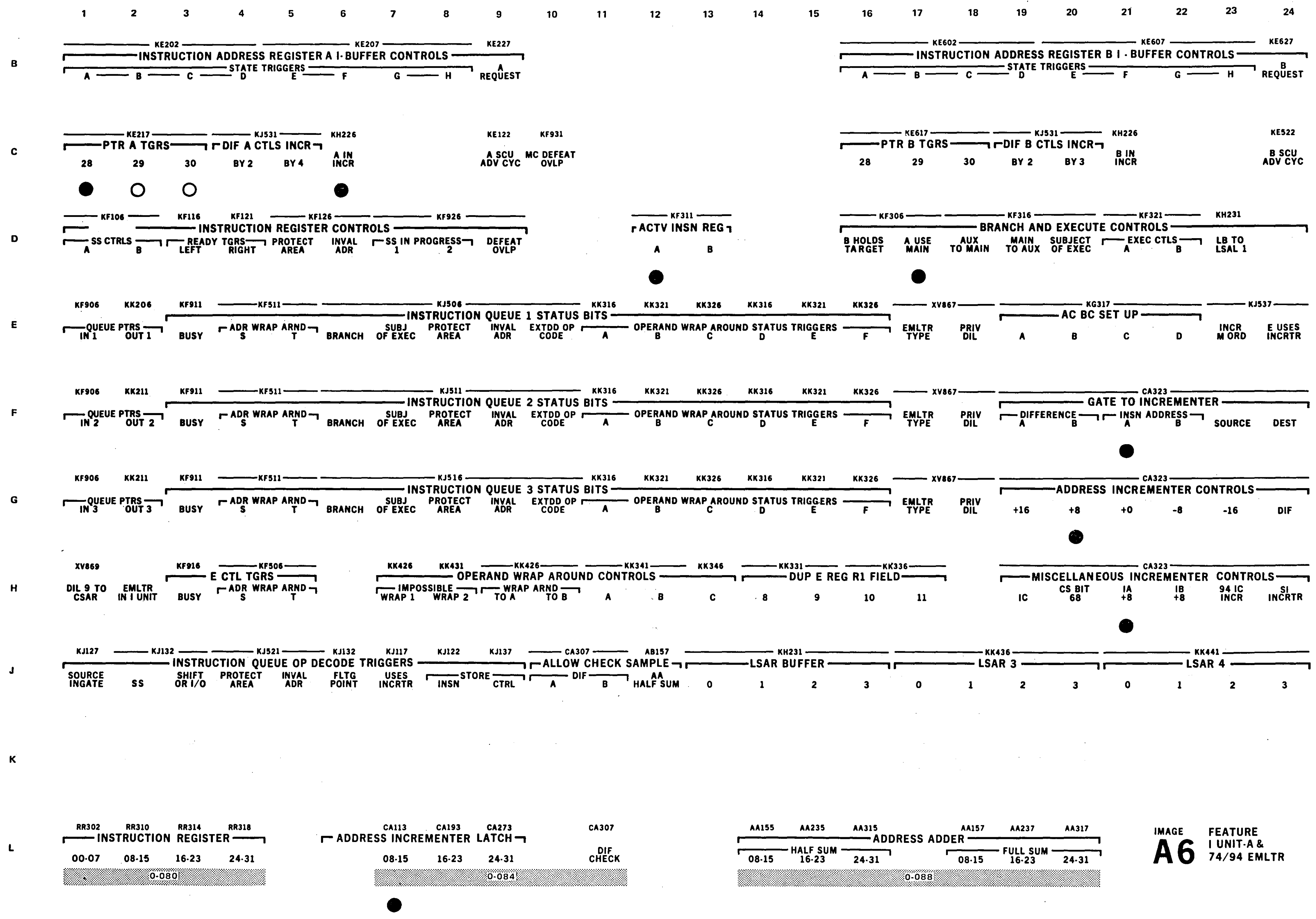
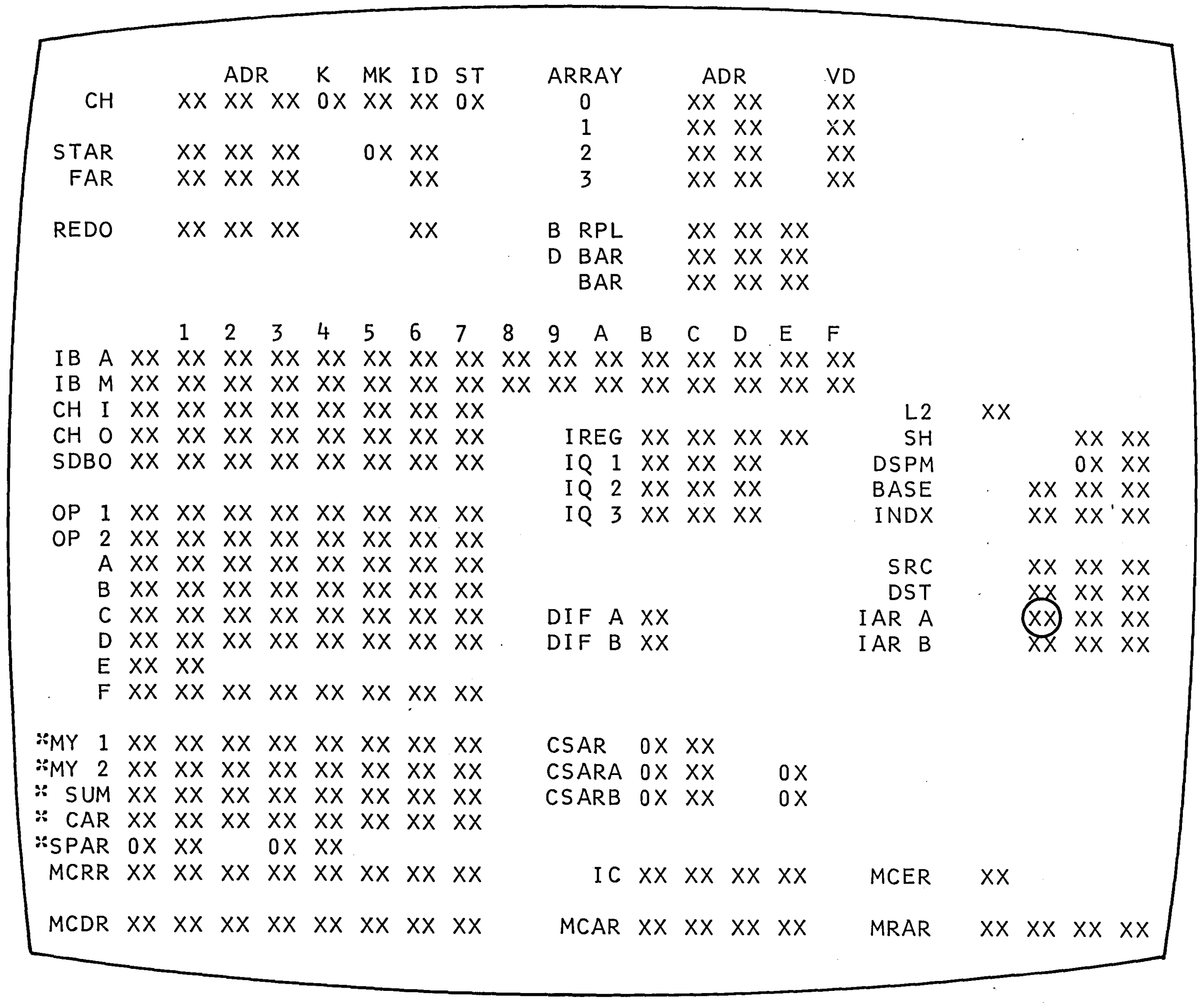


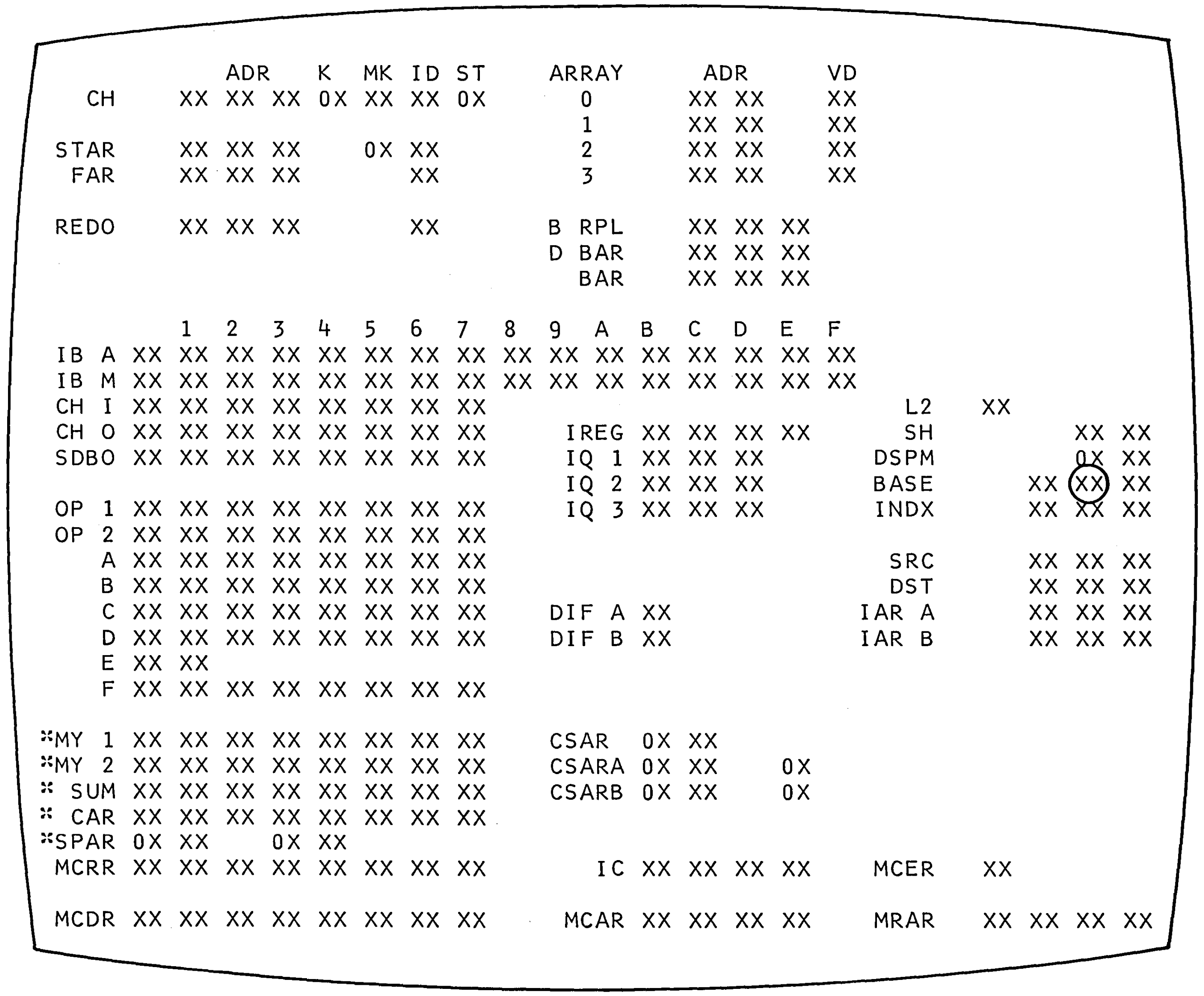
Figure 7-3. Image A6 For Session 7, Question 2



```
          ADR   K  MK ID ST      ARRAY      ADR      VD
  CH    XX XX XX 0X XX XX 0X       0        XX XX    XX
                                   1        XX XX    XX
 STAR   XX XX XX    0X XX          2        XX XX    XX
  FAR   XX XX XX       XX          3        XX XX    XX

 REDO   XX XX XX       XX        B RPL      XX XX XX
                                 D BAR      XX XX XX
                                   BAR      XX XX XX

        1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
 IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
 IB M XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
 CH I XX XX XX XX XX XX XX XX                              L2    XX
 CH O XX XX XX XX XX XX XX XX      IREG XX XX XX XX        SH           XX XX
 SDBO XX XX XX XX XX XX XX XX      IQ 1 XX XX XX         DSPM           0X XX
                                   IQ 2 XX XX XX         BASE        XX XX XX
 OP 1 XX XX XX XX XX XX XX XX      IQ 3 XX XX XX         INDX        XX XX XX
 OP 2 XX XX XX XX XX XX XX XX
    A XX XX XX XX XX XX XX XX                             SRC        XX XX XX
    B XX XX XX XX XX XX XX XX                             DST        XX XX XX
    C XX XX XX XX XX XX XX XX    DIF A XX                IAR A       XX XX XX
    D XX XX XX XX XX XX XX XX    DIF B XX                IAR B       XX XX XX
    E XX XX
    F XX XX XX XX XX XX XX XX

:MY 1 XX XX XX XX XX XX XX XX    CSAR  0X XX
:MY 2 XX XX XX XX XX XX XX XX    CSARA 0X XX     0X
: SUM XX XX XX XX XX XX XX XX    CSARB 0X XX     0X
: CAR XX XX XX XX XX XX XX XX
:SPAR 0X XX    0X XX
 MCRR XX XX XX XX XX XX XX XX       IC XX XX XX XX      MCER     XX

 MCDR XX XX XX XX XX XX XX XX      MCAR XX XX XX XX     MRAR     XX XX XX XX
```

○ = FLASHING

Figure 7-4. CRT For Session 7, Question 2



```
          ADR   K  MK ID ST      ARRAY      ADR      VD
  CH    XX XX XX 0X XX XX 0X       0        XX XX    XX
                                   1        XX XX    XX
 STAR   XX XX XX    0X XX          2        XX XX    XX
  FAR   XX XX XX       XX          3        XX XX    XX

 REDO   XX XX XX       XX        B RPL      XX XX XX
                                 D BAR      XX XX XX
                                   BAR      XX XX XX

        1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
 IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
 IB M XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
 CH I XX XX XX XX XX XX XX XX                              L2    XX
 CH O XX XX XX XX XX XX XX XX      IREG XX XX XX XX        SH           XX XX
 SDBO XX XX XX XX XX XX XX XX      IQ 1 XX XX XX         DSPM           0X XX
                                   IQ 2 XX XX XX         BASE        XX XX XX
 OP 1 XX XX XX XX XX XX XX XX      IQ 3 XX XX XX         INDX        XX XX XX
 OP 2 XX XX XX XX XX XX XX XX
    A XX XX XX XX XX XX XX XX                             SRC        XX XX XX
    B XX XX XX XX XX XX XX XX                             DST        XX XX XX
    C XX XX XX XX XX XX XX XX    DIF A XX                IAR A       XX XX XX
    D XX XX XX XX XX XX XX XX    DIF B XX                IAR B       XX XX XX
    E XX XX
    F XX XX XX XX XX XX XX XX

:MY 1 XX XX XX XX XX XX XX XX    CSAR  0X XX
:MY 2 XX XX XX XX XX XX XX XX    CSARA 0X XX     0X
: SUM XX XX XX XX XX XX XX XX    CSARB 0X XX     0X
: CAR XX XX XX XX XX XX XX XX
:SPAR 0X XX    0X XX
 MCRR XX XX XX XX XX XX XX XX       IC XX XX XX XX      MCER     XX

 MCDR XX XX XX XX XX XX XX XX      MCAR XX XX XX XX     MRAR     XX XX XX XX
```

○ = FLASHING

Figure 7-5. CRT For Session 7, Question 3

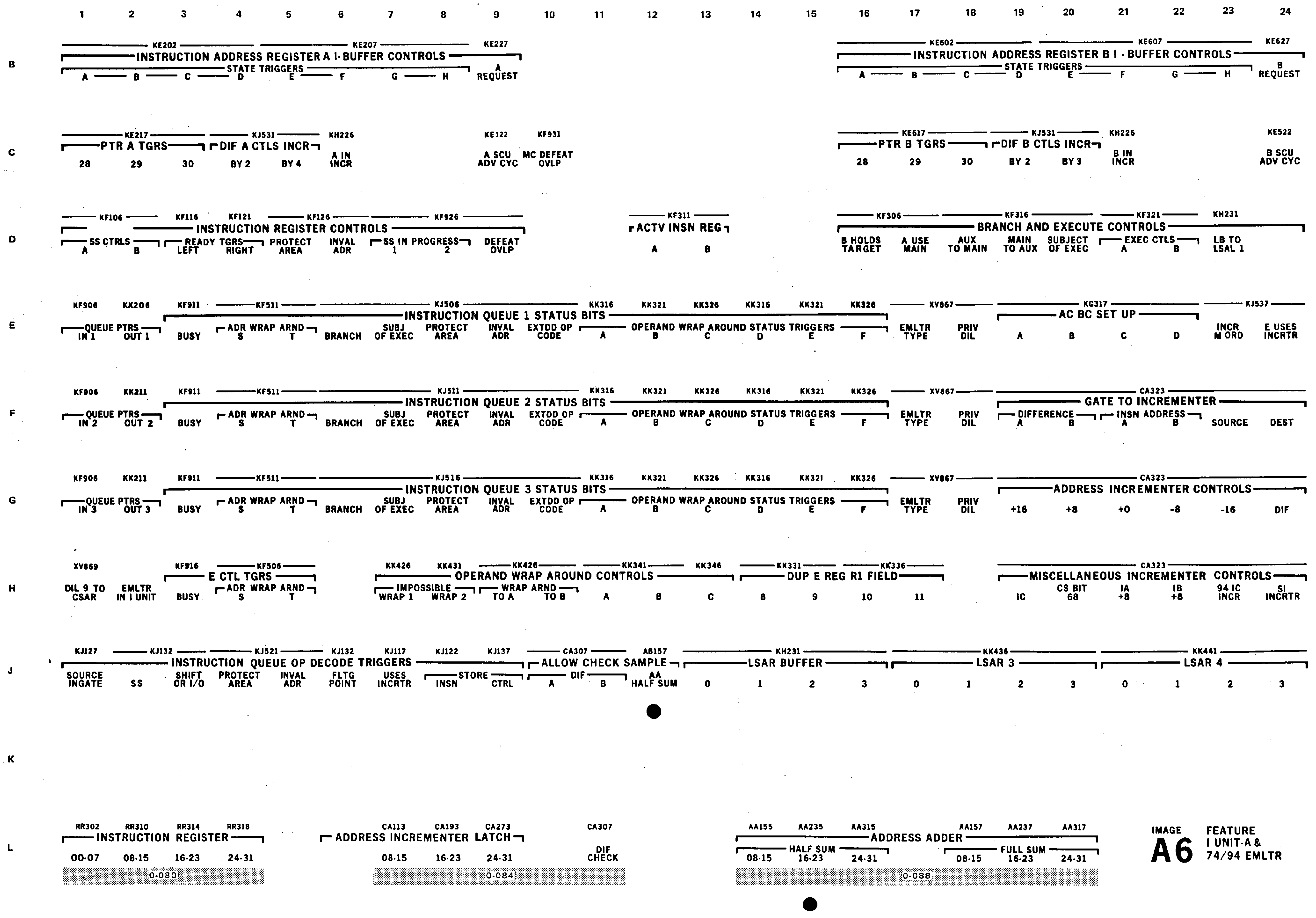


Figure 7-6. Image A6 For Session 7, Question 3

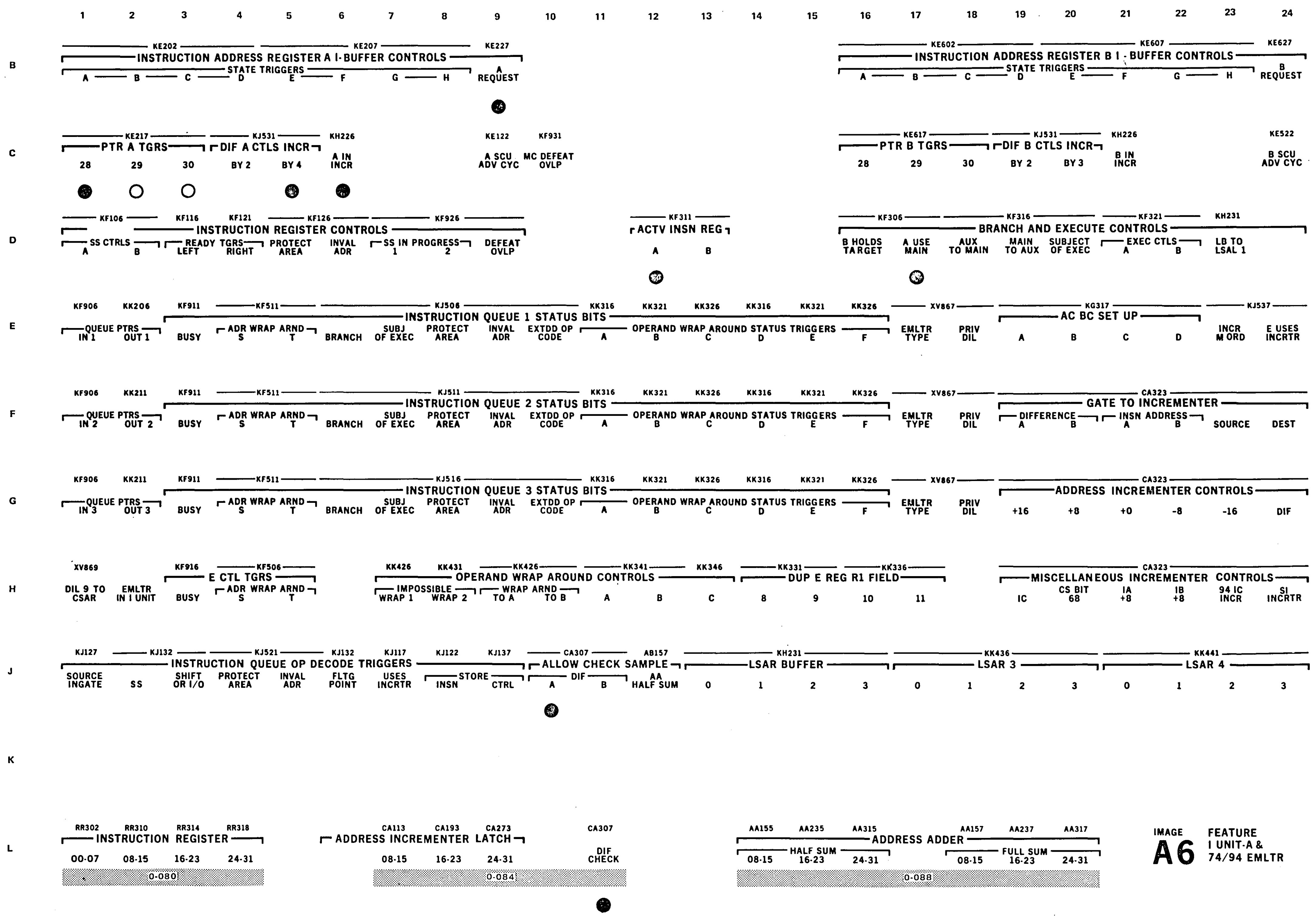
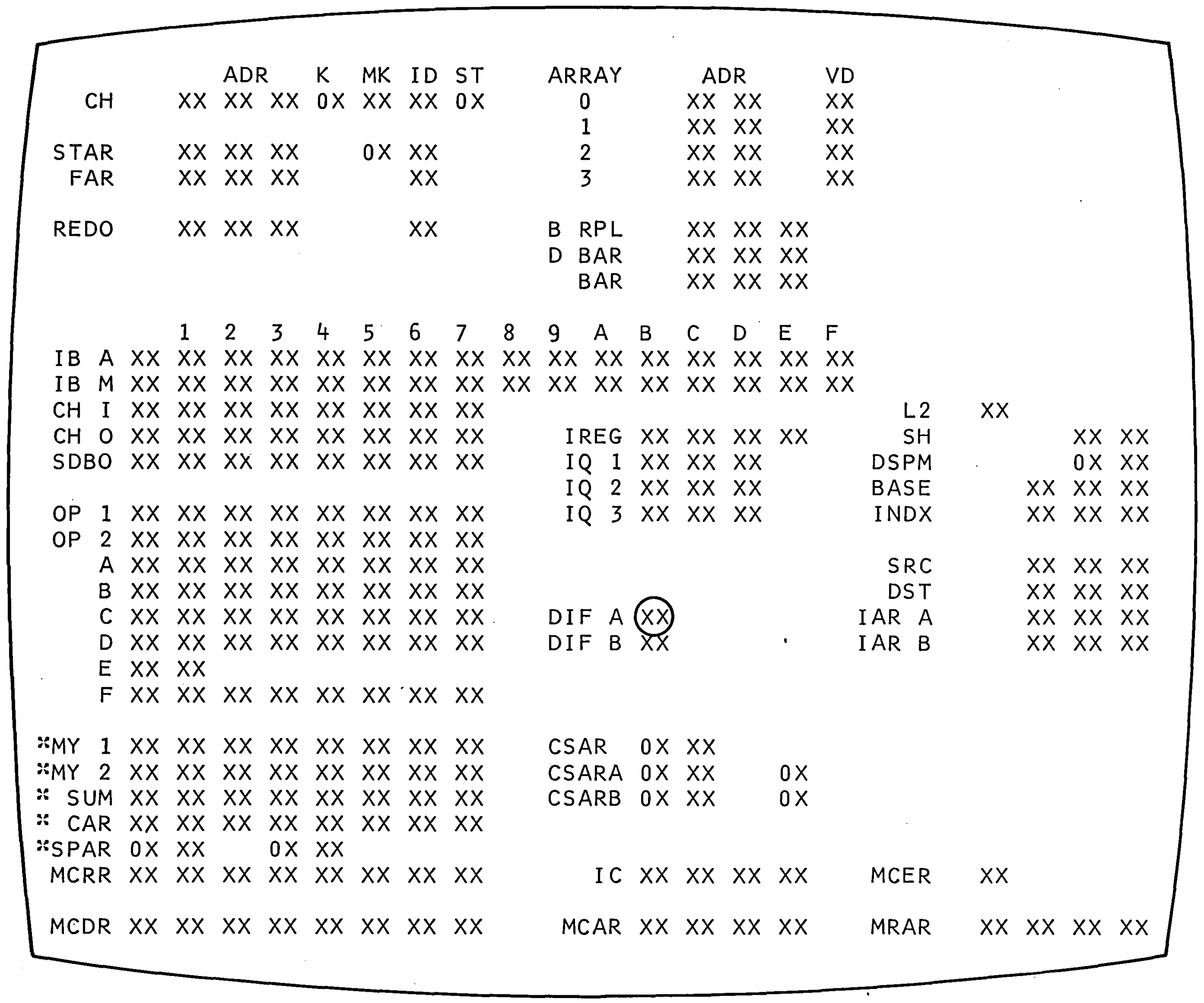


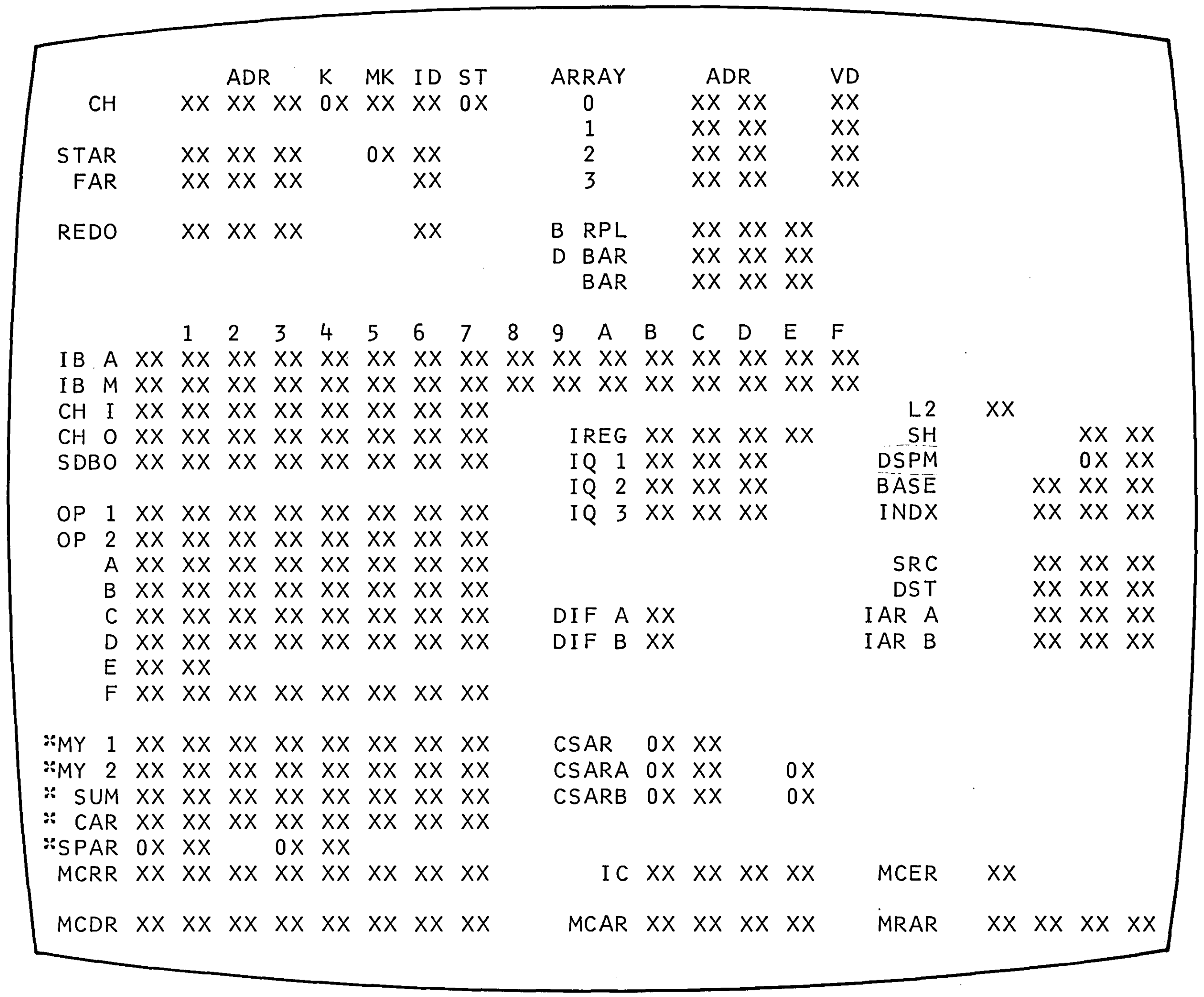
Figure 7-7. Image A6 For Session 7, Question 4



```
          ADR    K  MK ID ST     ARRAY      ADR      VD
 CH     XX XX XX 0X XX XX 0X       0       XX XX     XX
                                   1       XX XX     XX
STAR    XX XX XX    0X XX          2       XX XX     XX
 FAR    XX XX XX       XX          3       XX XX     XX

REDO    XX XX XX       XX        B RPL     XX XX XX
                                 D BAR     XX XX XX
                                   BAR     XX XX XX

        1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
IB M XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
CH I XX XX XX XX XX XX XX XX                       L2   XX
CH O XX XX XX XX XX XX XX XX   IREG XX XX XX XX    SH            XX XX
SDBO XX XX XX XX XX XX XX XX   IQ 1 XX XX XX      DSPM           0X XX
                               IQ 2 XX XX XX      BASE        XX XX XX
OP 1 XX XX XX XX XX XX XX XX   IQ 3 XX XX XX      INDX        XX XX XX
OP 2 XX XX XX XX XX XX XX XX
   A XX XX XX XX XX XX XX XX                       SRC        XX XX XX
   B XX XX XX XX XX XX XX XX                       DST        XX XX XX
   C XX XX XX XX XX XX XX XX   DIF A XX           IAR A       XX XX XX
   D XX XX XX XX XX XX XX XX   DIF B XX           IAR B       XX XX XX
   E XX XX
   F XX XX XX XX XX XX XX XX

:MY 1 XX XX XX XX XX XX XX XX  CSAR  0X XX
:MY 2 XX XX XX XX XX XX XX XX  CSARA 0X XX     0X
: SUM XX XX XX XX XX XX XX XX  CSARB 0X XX     0X
: CAR XX XX XX XX XX XX XX XX
:SPAR 0X XX    0X XX
MCRR XX XX XX XX XX XX XX XX     IC XX XX XX XX    MCER    XX

MCDR XX XX XX XX XX XX XX XX   MCAR XX XX XX XX    MRAR    XX XX XX XX
```

◯ = FLASHING

Figure 7-8. CRT For Session 7, Question 4



```
          ADR    K  MK ID ST     ARRAY      ADR      VD
 CH     XX XX XX 0X XX XX 0X       0       XX XX     XX
                                   1       XX XX     XX
STAR    XX XX XX    0X XX          2       XX XX     XX
 FAR    XX XX XX       XX          3       XX XX     XX

REDO    XX XX XX       XX        B RPL     XX XX XX
                                 D BAR     XX XX XX
                                   BAR     XX XX XX

        1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
IB M XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
CH I XX XX XX XX XX XX XX XX                       L2   XX
CH O XX XX XX XX XX XX XX XX   IREG XX XX XX XX    SH            XX XX
SDBO XX XX XX XX XX XX XX XX   IQ 1 XX XX XX      DSPM           0X XX
                               IQ 2 XX XX XX      BASE        XX XX XX
OP 1 XX XX XX XX XX XX XX XX   IQ 3 XX XX XX      INDX        XX XX XX
OP 2 XX XX XX XX XX XX XX XX
   A XX XX XX XX XX XX XX XX                       SRC        XX XX XX
   B XX XX XX XX XX XX XX XX                       DST        XX XX XX
   C XX XX XX XX XX XX XX XX   DIF A XX           IAR A       XX XX XX
   D XX XX XX XX XX XX XX XX   DIF B XX           IAR B       XX XX XX
   E XX XX
   F XX XX XX XX XX XX XX XX

:MY 1 XX XX XX XX XX XX XX XX  CSAR  0X XX
:MY 2 XX XX XX XX XX XX XX XX  CSARA 0X XX     0X
: SUM XX XX XX XX XX XX XX XX  CSARB 0X XX     0X
: CAR XX XX XX XX XX XX XX XX
:SPAR 0X XX    0X XX
MCRR XX XX XX XX XX XX XX XX     IC XX XX XX XX    MCER    XX

MCDR XX XX XX XX XX XX XX XX   MCAR XX XX XX XX    MRAR    XX XX XX XX
```

◯ = FLASHING

Figure 7-9. CRT For Session 7, Question 5

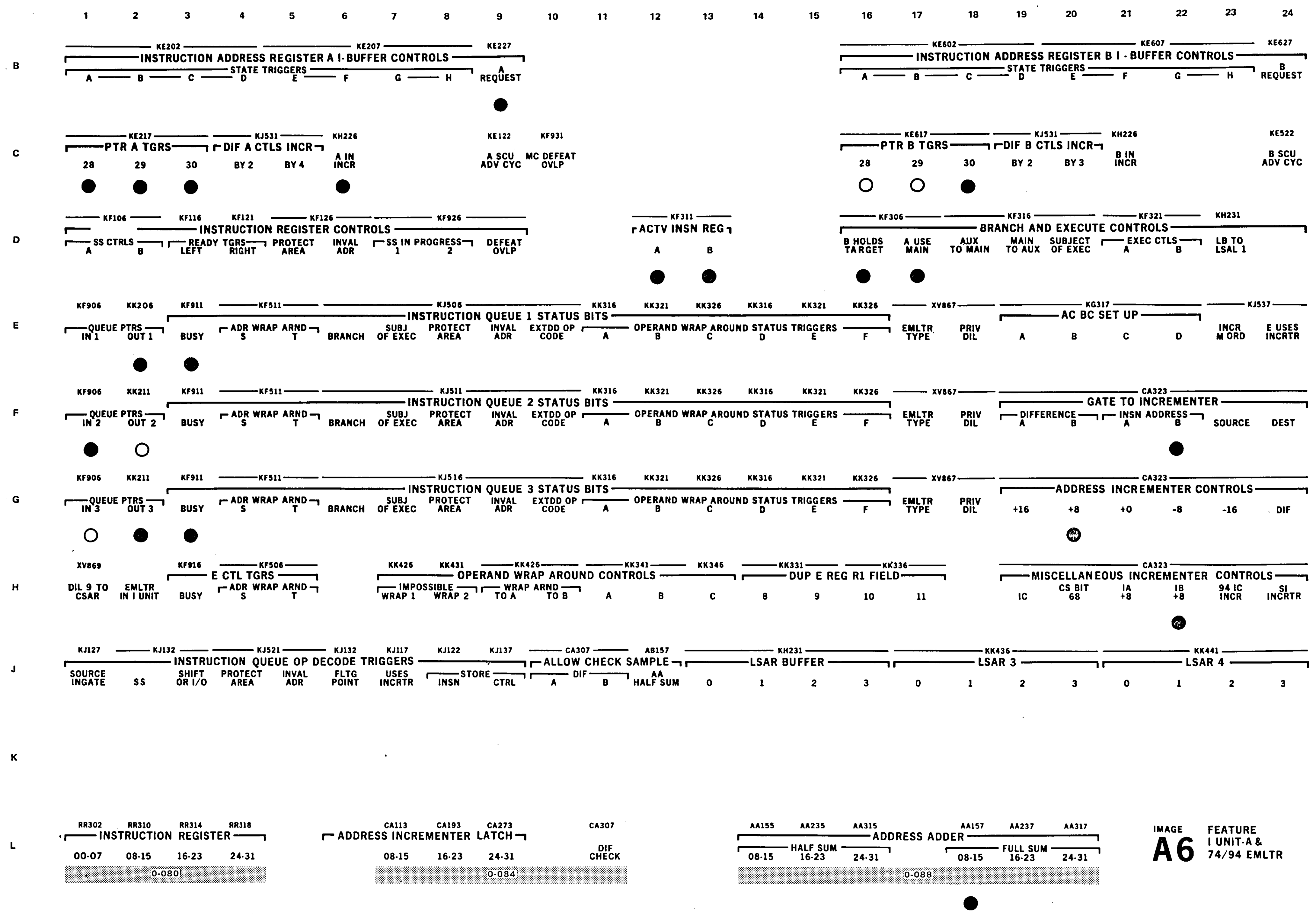
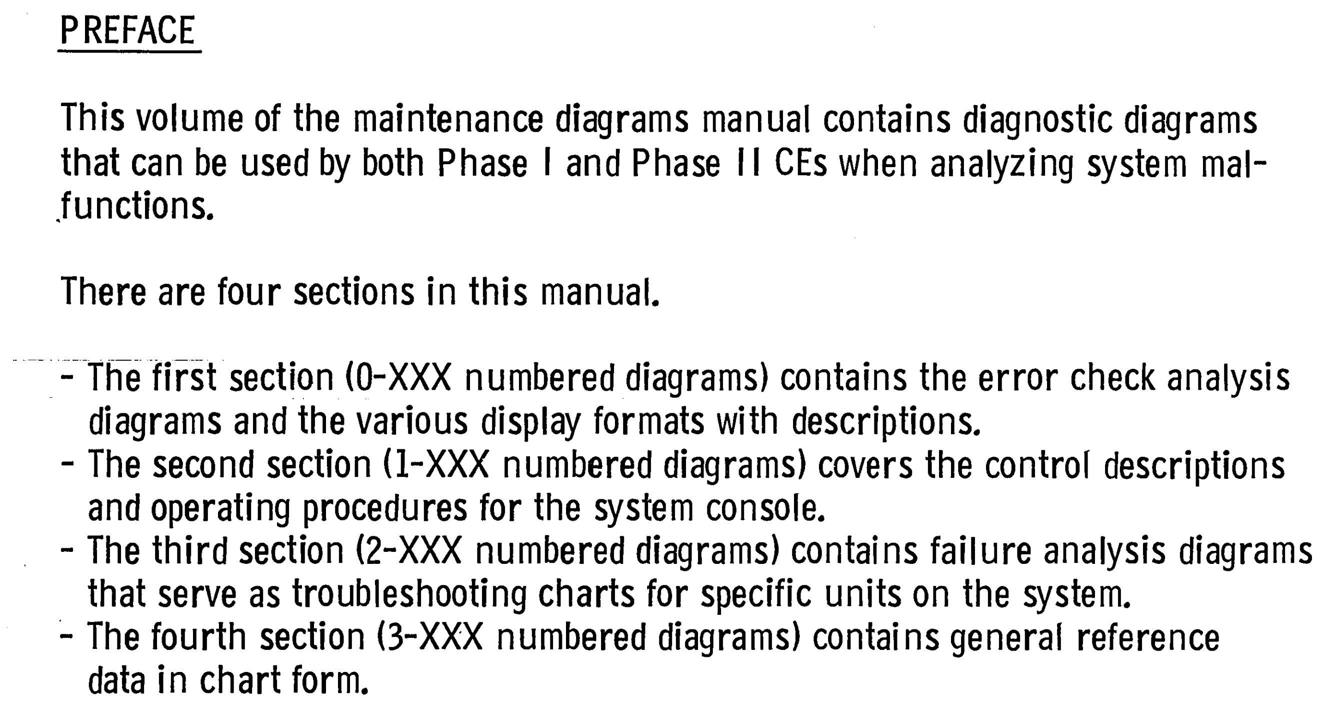


Figure 7-10. Image A6 For Session 7, Question 5



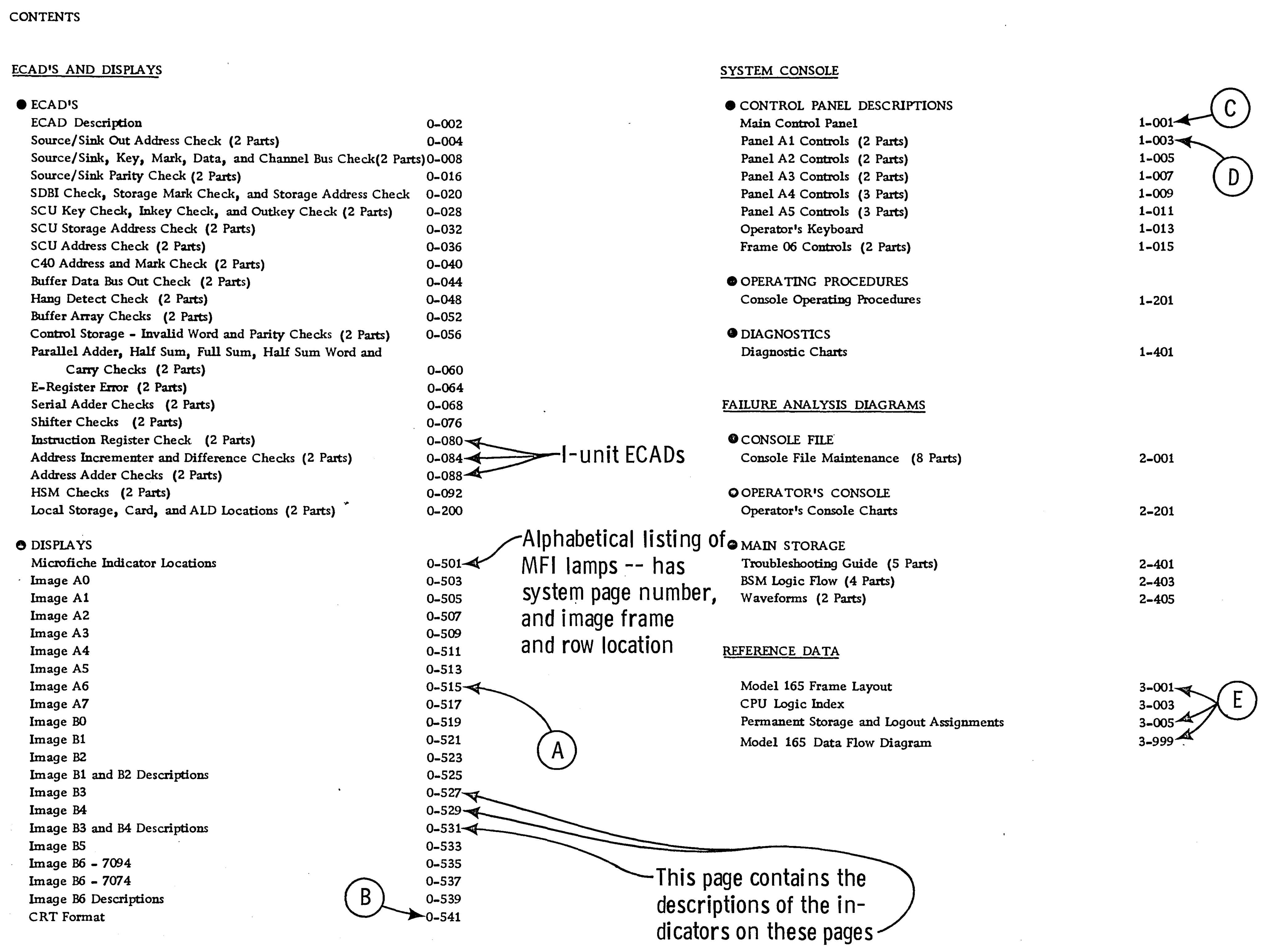
PREFACE

This volume of the maintenance diagrams manual contains diagnostic diagrams that can be used by both Phase I and Phase II CEs when analyzing system malfunctions.

There are four sections in this manual.

- The first section (0-XXX numbered diagrams) contains the error check analysis diagrams and the various display formats with descriptions.
- The second section (1-XXX numbered diagrams) covers the control descriptions and operating procedures for the system console.
- The third section (2-XXX numbered diagrams) contains failure analysis diagrams that serve as troubleshooting charts for specific units on the system.
- The fourth section (3-XXX numbered diagrams) contains general reference data in chart form.

Figure 7-11. Volume 1 Sections



CONTENTS

I-unit ECADs
Alphabetical listing of MFI lamps -- has system page number, and image frame and row location
A
B
C
D
E
This page contains the descriptions of the indicators on these pages

Figure 7-12. Volume 1 Contents

ERROR CHECK ANALYSIS DIAGRAMS (ECADS)

Error check analysis diagrams (ECADs) show line names, card locations, and logic pages for the circuits that activate the microfiche error lights. The ECADs do not necessarily show every logic line in the data flow path related to a given error light. They do, however, show all the cards in the data path. In addition, some control logic is shown. Note that ECADs are applicable only to machine-check type problems.

ECAD DESCRIPTION

Each ECAD is structured so that the user can go back from the microfiche error indicator through the display logic to the error detection point that controls the given indicator. The ECAD then covers the functional logic back to the previous error detection point in the data flow. Included on the ECAD (usually a multi-sheet diagram) is a data flow drawing of the logic area in question. The data flow shows the error checking points.

Additional information found on ECADs:

- Heavy broken lines show error check flow from the error detection logic to the indicator. Although not shown on most ECADs, the machine error check lines also go to the logic that stops the clocks and freezes the maintenance control registers that are buffering retry information.
- Heavy solid lines show the functional flow of data, addresses, marks, keys, etc.
- Normal-weigth solid lines show control flow.
- Some ECADs contain a list of suggested microdiagnostic and/or diagnostic programs.
- Where applicable, each ECAD lists related diagrams that are located in volumes 2 through 6 of the FEMDM.

HOW TO LOCATE AN ECAD

To locate an ECAD for a given error light refer to the ECAD Contents list. This list is in order by microfiche frame, row, and column. ECAD numbers are also shown on the microfiche images in diagrams 1-001 through 1-031 in this manual. The ECAD numbers are in the shaded areas directly under the error-light nomenclature.

HOW TO USE AN ECAD

Use an ECAD in the following manner to locate suspect cards.

1. Check the data flow connected by the heavy solid lines. Locate the block closest to the error light. Start at the first card location above the WDxxx logic pages and work back changing one card at a time.
2. Check the control flow (normal-weight solid lines) in the same manner as described above.
3. Check the error detection logic (heavy broken lines).

Figure 7-13. ECAD Information

● Machine Check Switch

The machine check switch is a three position (STOP ON CHK, PROC, DSBL) lever switch that is used to modify the handling of a machine check within the system.

1. Stop On Check: With the switch in this position, all machine checks cause a hard stop. There is no logout and no interrupt is taken. If the switch is moved to the process position and the start switch is pressed, a retry is attempted if the machine is in retry mode. If the machine is not in retry mode, results are unpredictable.

2. Process: With the switch in the process position, all machine checks cause a hard stop, a logout, and (if the retry switch is set to NORM) an instruction retry. If retry is disabled, a logout and machine check interrupt occurs. If retry is in the count mode, a logout and machine check interrupt occurs only on count overflow.

3. Disable: In the disable position, this switch causes all machine checks to turn on the associated check triggers. No other action is taken.

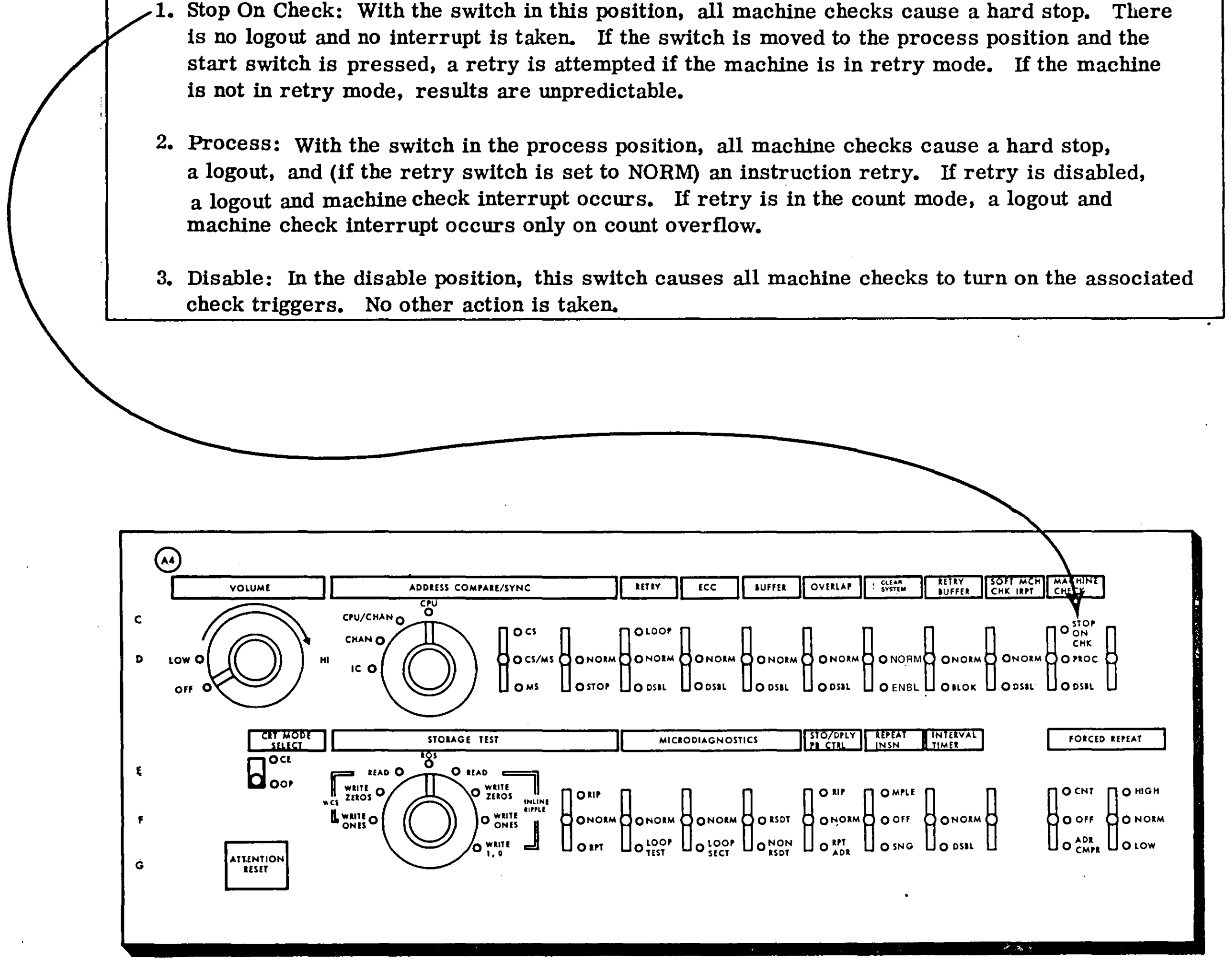
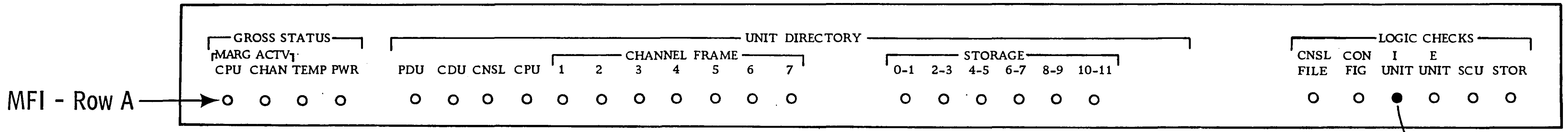


Figure 7-14. Stop On Check



FIXED INDICATORS (MFI DISPLAY)

GROSS STATUS INDICATORS

- Margin Active, CPU -- One of the four margin control switches is not set at the nominal value (1.25V dc).
- Margin Active, Channel -- A margin control on one of the channel frames is not set at the nominal value. (Does not apply to the 2880)
- Temperature -- One of the following coolant conditions exists in the CDU: high temperature, low temperature, low reservoir, or low flow.
- Power -- Power is not up on a unit that has its local/remote switch set to remote. This indicator is also on if the MG detects an over-voltage condition or if the MG is set to the local (voltage sense) mode.

UNIT DIRECTORY INDICATORS

- PDU -- One of the following PDU conditions exists: MG check, PDU thermal check, ac or dc circuit breaker tripped.
- CDU -- One of the following CDU conditions exists: empty reservoir, no 208V ac to the pump, or any of the conditions that turn on the gross status temperature indicator.
- Console -- There is a tripped CB in frame 06 or a thermal indication in frame 05 or 06.
- CPU -- There is a thermal indication in the CPU or PDU or a tripped CB in the PDU.
- Channel Frames 1 through 7 -- Power is not up on the indicated channel frame (local/remote switch set to remote).
- Storage 0,1 through 10,11 -- Power is not up on the indicated logical storage unit (local/remote switch set to remote).

LOGIC CHECK INDICATORS

The five logic check indicators are used to indicate a machine check condition. Additional information for a logic check condition is obtained by displaying the appropriate image on the MFI display.

●Configuration check	Image A1
●I-Unit	Image A6
●E-Unit	Image A5
●Storage Control Unit	Image A2
●Storage	Image A1

Figure 7-15. MFI Row A

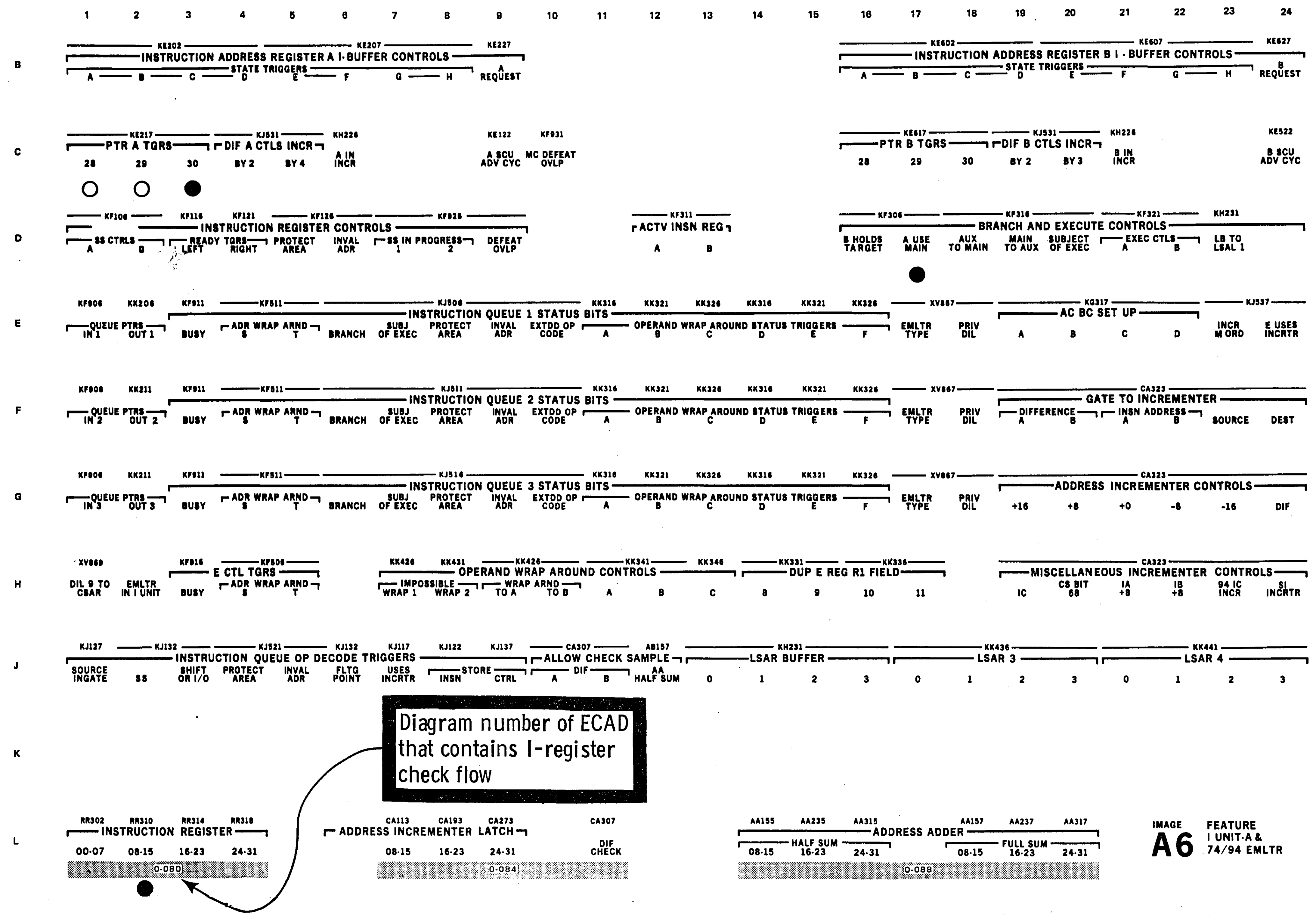
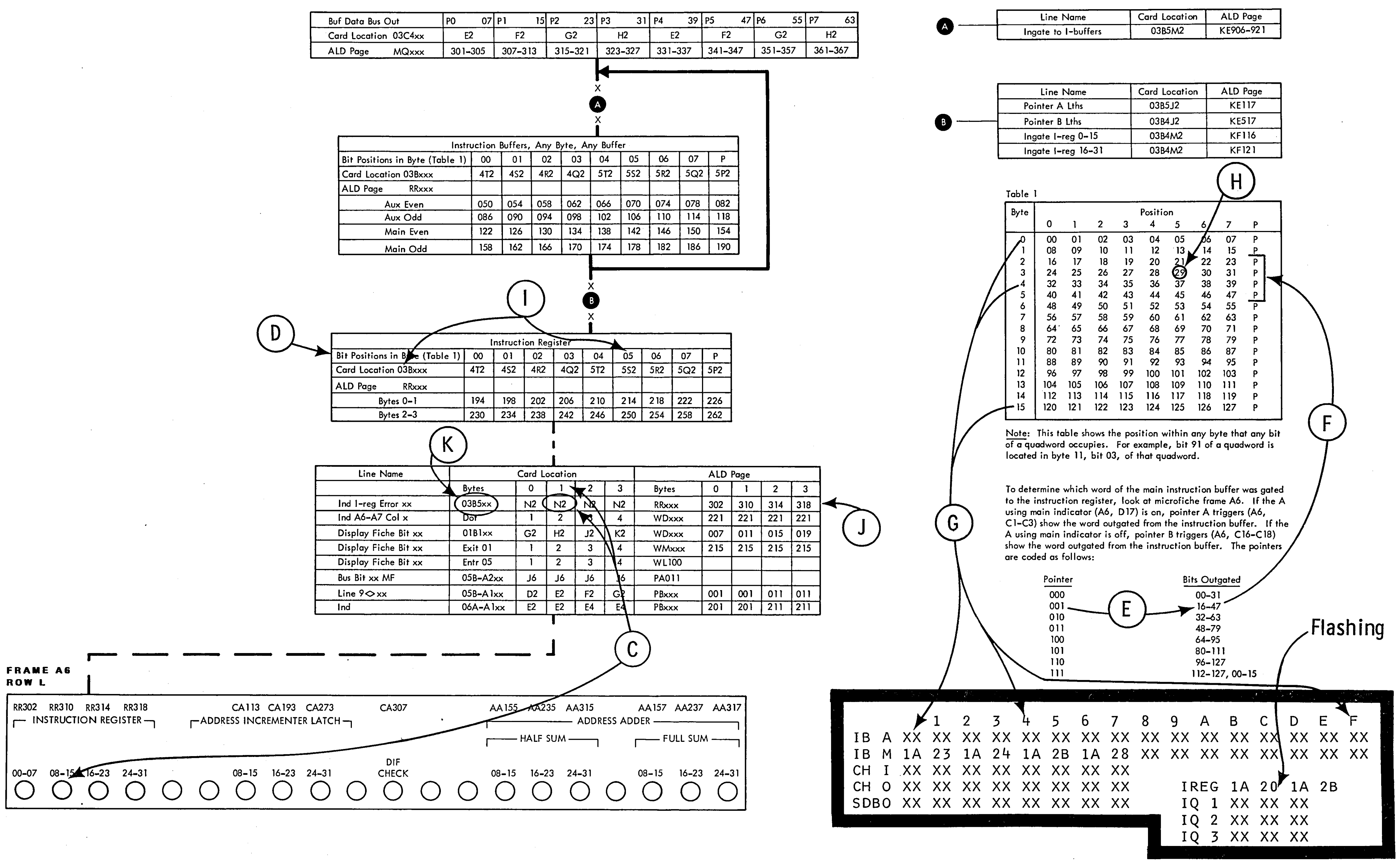


Figure 7-16. Image A6.



Buf Data Bus Out	P0 07	P1 15	P2 23	P3 31	P4 39	P5 47	P6 55	P7 63
Card Location 03C4xx	E2	F2	G2	H2	E2	F2	G2	H2
ALD Page MQxxx	301-305	307-313	315-321	323-327	331-337	341-347	351-357	361-367

Instruction Buffers, Any Byte, Any Buffer									
Bit Positions in Byte (Table 1)	00	01	02	03	04	05	06	07	P
Card Location 03Bxxx	4T2	4S2	4R2	4Q2	5T2	5S2	5R2	5Q2	5P2
ALD Page RRxxx									
Aux Even	050	054	058	062	066	070	074	078	082
Aux Odd	086	090	094	098	102	106	110	114	118
Main Even	122	126	130	134	138	142	146	150	154
Main Odd	158	162	166	170	174	178	182	186	190

Instruction Register									
Bit Positions in Byte (Table 1)	00	01	02	03	04	05	06	07	P
Card Location 03Bxxx	4T2	4S2	4R2	4Q2	5T2	5S2	5R2	5Q2	5P2
ALD Page RRxxx									
Bytes 0-1	194	198	202	206	210	214	218	222	226
Bytes 2-3	230	234	238	242	246	250	254	258	262

Line Name	Card Location					ALD Page				
	Bytes	0	1	2	3	Bytes	0	1	2	3
Ind I-reg Error xx	03B5xx	N2	N2	N2	N2	RRxxx	302	310	314	318
Ind A6-A7 Col x	Dot	1	2	3	4	WDxxx	221	221	221	221
Display Fiche Bit xx	01B1xx	G2	H2	J2	K2	WDxxx	007	011	015	019
Display Fiche Bit xx	Exit 01	1	2	3	4	WMxxx	215	215	215	215
Display Fiche Bit xx	Entr 05	1	2	3	4	WL100				
Bus Bit xx MF	05B-A2xx	J6	J6	J6	J6	PA011				
Line 9<>xx	05B-A1xx	D2	E2	F2	G2	PBxxx	001	001	011	011
Ind	06A-A1xx	E2	E2	E4	E4	PBxxx	201	201	211	211

Line Name	Card Location	ALD Page
Ingate to I-buffers	03B5M2	KE906-921

Line Name	Card Location	ALD Page
Pointer A Lths	03B5J2	KE117
Pointer B Lths	03B4J2	KE517
Ingate I-reg 0-15	03B4M2	KF116
Ingate I-reg 16-31	03B4M2	KF121

Table 1

Byte	Position								
	0	1	2	3	4	5	6	7	P
0	00	01	02	03	04	05	06	07	P
1	08	09	10	11	12	13	14	15	P
2	16	17	18	19	20	21	22	23	P
3	24	25	26	27	28	29	30	31	P
4	32	33	34	35	36	37	38	39	P
5	40	41	42	43	44	45	46	47	P
6	48	49	50	51	52	53	54	55	P
7	56	57	58	59	60	61	62	63	P
8	64	65	66	67	68	69	70	71	P
9	72	73	74	75	76	77	78	79	P
10	80	81	82	83	84	85	86	87	P
11	88	89	90	91	92	93	94	95	P
12	96	97	98	99	100	101	102	103	P
13	104	105	106	107	108	109	110	111	P
14	112	113	114	115	116	117	118	119	P
15	120	121	122	123	124	125	126	127	P

Note: This table shows the position within any byte that any bit of a quadword occupies. For example, bit 91 of a quadword is located in byte 11, bit 03, of that quadword.

To determine which word of the main instruction buffer was gated to the instruction register, look at microfiche frame A6. If the A using main indicator (A6, D17) is on, pointer A triggers (A6, C1-C3) show the word outgated from the instruction buffer. If the A using main indicator is off, pointer B triggers (A6, C16-C18) show the word outgated from the instruction buffer. The pointers are coded as follows:

Pointer	Bits Outgated
000	00-31
001	16-47
010	32-63
011	48-79
100	64-95
101	80-111
110	96-127
111	112-127, 00-15

```
          1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
IB  A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
IB  M 1A 23 1A 24 1A 2B 1A 28 XX XX XX XX XX XX XX XX
CH  I XX XX XX XX XX XX XX XX
CH  O XX XX XX XX XX XX XX XX      IREG 1A 20 1A 2B
SDBO  XX XX XX XX XX XX XX XX      IQ 1 XX XX XX
                                   IQ 2 XX XX XX
                                   IQ 3 XX XX XX
```

Figure 7-17. I-Register ECAD

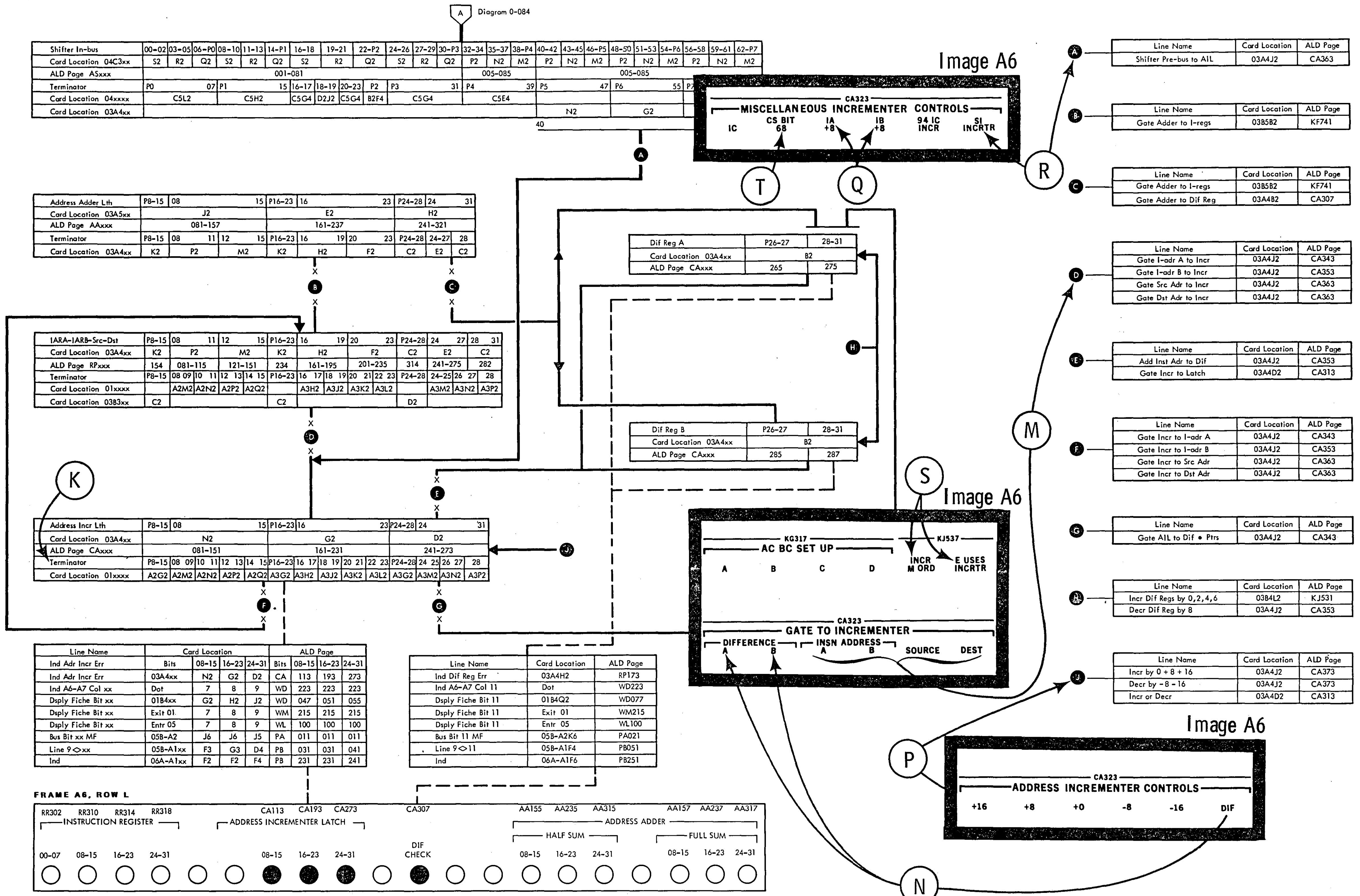


Diagram 0-084
Shifter In-bus | 00-02 | 03-05 | 06-P0 | 08-10 | 11-13 | 14-P1 | 16-18 | 19-21 | 22-P2 | 24-26 | 27-29 | 30-P3 | 32-34 | 35-37 | 38-P4 | 40-42 | 43-45 | 46-P5 | 48-50 | 51-53 | 54-P6 | 56-58 | 59-61 | 62-P7
Card Location 04C3xx | S2 | R2 | Q2 | S2 | R2 | Q2 | S2 | R2 | Q2 | S2 | R2 | Q2 | P2 | N2 | M2 | P2 | N2 | M2 | P2 | N2 | M2 | P2 | N2 | M2
ALD Page ASxxx | 001-081 | 005-085 | 005-085
Terminator | P0 07 | P1 15 | 16-17 | 18-19 | 20-23 | P2 | P3 31 | P4 39 | P5 47 | P6 55 | P7
Card Location 04xxxx | C5L2 | C5H2 | C5G4 | D2J2 | C5G4 | B2F4 | C5G4 | C5E4
Card Location 03A4xx | N2 | G2
40
Image A6
CA323
MISCELLANEOUS INCREMENTER CONTROLS
IC
CS BIT 68
IA +8
IB +8
94 IC INCR
SI INCRTR
T
Q
R
Address Adder Lth | P8-15 | 08 15 | P16-23 | 16 23 | P24-28 | 24 31
Card Location 03A5xx | J2 | E2 | H2
ALD Page AAxxx | 081-157 | 161-237 | 241-321
Terminator | P8-15 | 08 11 | 12 15 | P16-23 | 16 19 | 20 23 | P24-28 | 24-27 | 28
Card Location 03A4xx | K2 | P2 | M2 | K2 | H2 | F2 | C2 | E2 | C2
Dif Reg A | P26-27 | 28-31
Card Location 03A4xx | B2
ALD Page CAxxx | 265 | 275
H
IARA-IARB-Src-Dst | P8-15 | 08 11 | 12 15 | P16-23 | 16 19 | 20 23 | P24-28 | 24 27 | 28 31
Card Location 03A4xx | K2 | P2 | M2 | K2 | H2 | F2 | C2 | E2 | C2
ALD Page RPxxx | 154 | 081-115 | 121-151 | 234 | 161-195 | 201-235 | 314 | 241-275 | 282
Terminator | P8-15 | 08 09 | 10 11 | 12 13 | 14 15 | P16-23 | 16 17 | 18 19 | 20 21 | 22 23 | P24-28 | 24-25 | 26 27 | 28
Card Location 01xxxx | | A2M2 | A2N2 | A2P2 | A2Q2 | | A3H2 | A3J2 | A3K2 | A3L2 | | A3M2 | A3N2 | A3P2
Card Location 03B3xx | C2 | | C2 | | D2 |
Dif Reg B | P26-27 | 28-31
Card Location 03A4xx | B2
ALD Page CAxxx | 285 | 287
M
S
K
Address Incr Lth | P8-15 | 08 15 | P16-23 | 16 23 | P24-28 | 24 31
Card Location 03A4xx | N2 | G2 | D2
ALD Page CAxxx | 081-151 | 161-231 | 241-273
Terminator | P8-15 | 08 09 | 10 11 | 12 13 | 14 15 | P16-23 | 16 17 | 18 19 | 20 21 | 22 23 | P24-28 | 24 25 | 26 27 | 28
Card Location 01xxxx | A2G2 | A2M2 | A2N2 | A2P2 | A2Q2 | A3G2 | A3H2 | A3J2 | A3K2 | A3L2 | A3G2 | A3M2 | A3N2 | A3P2
Image A6
KG317
AC BC SET UP
A
B
C
D
KJ537
INCR M ORD
E USES INCRTR
CA323
GATE TO INCREMENTER
DIFFERENCE A B
INSN ADDRESS A B
SOURCE
DEST
Image A6
P
CA323
ADDRESS INCREMENTER CONTROLS
+16
+8
+0
-8
-16
DIF
N
FRAME A6, ROW L
RR302
RR310
RR314
RR318
INSTRUCTION REGISTER
00-07
08-15
16-23
24-31
CA113
CA193
CA273
ADDRESS INCREMENTER LATCH
08-15
16-23
24-31
CA307
DIF CHECK
AA155
AA235
AA315
AA157
AA237
AA317
ADDRESS ADDER
HALF SUM
08-15
16-23
24-31
FULL SUM
08-15
16-23
24-31

A

Line Name	Card Location	ALD Page
Shifter Pre-bus to AIL	03A4J2	CA363

B

Line Name	Card Location	ALD Page
Gate Adder to I-regs	03B5B2	KF741

C

Line Name	Card Location	ALD Page
Gate Adder to I-regs	03B5B2	KF741
Gate Adder to Dif Reg	03A4B2	CA307

D

Line Name	Card Location	ALD Page
Gate I-adr A to Incr	03A4J2	CA343
Gate I-adr B to Incr	03A4J2	CA353
Gate Src Adr to Incr	03A4J2	CA363
Gate Dst Adr to Incr	03A4J2	CA363

E

Line Name	Card Location	ALD Page
Add Inst Adr to Dif	03A4J2	CA353
Gate Incr to Latch	03A4D2	CA313

F

Line Name	Card Location	ALD Page
Gate Incr to I-adr A	03A4J2	CA343
Gate Incr to I-adr B	03A4J2	CA353
Gate Incr to Src Adr	03A4J2	CA363
Gate Incr to Dst Adr	03A4J2	CA363

G

Line Name	Card Location	ALD Page
Gate AIL to Dif • Ptrs	03A4J2	CA343

H

Line Name	Card Location	ALD Page
Incr Dif Regs by 0,2,4,6	03B4L2	KJ531
Decr Dif Reg by 8	03A4J2	CA353

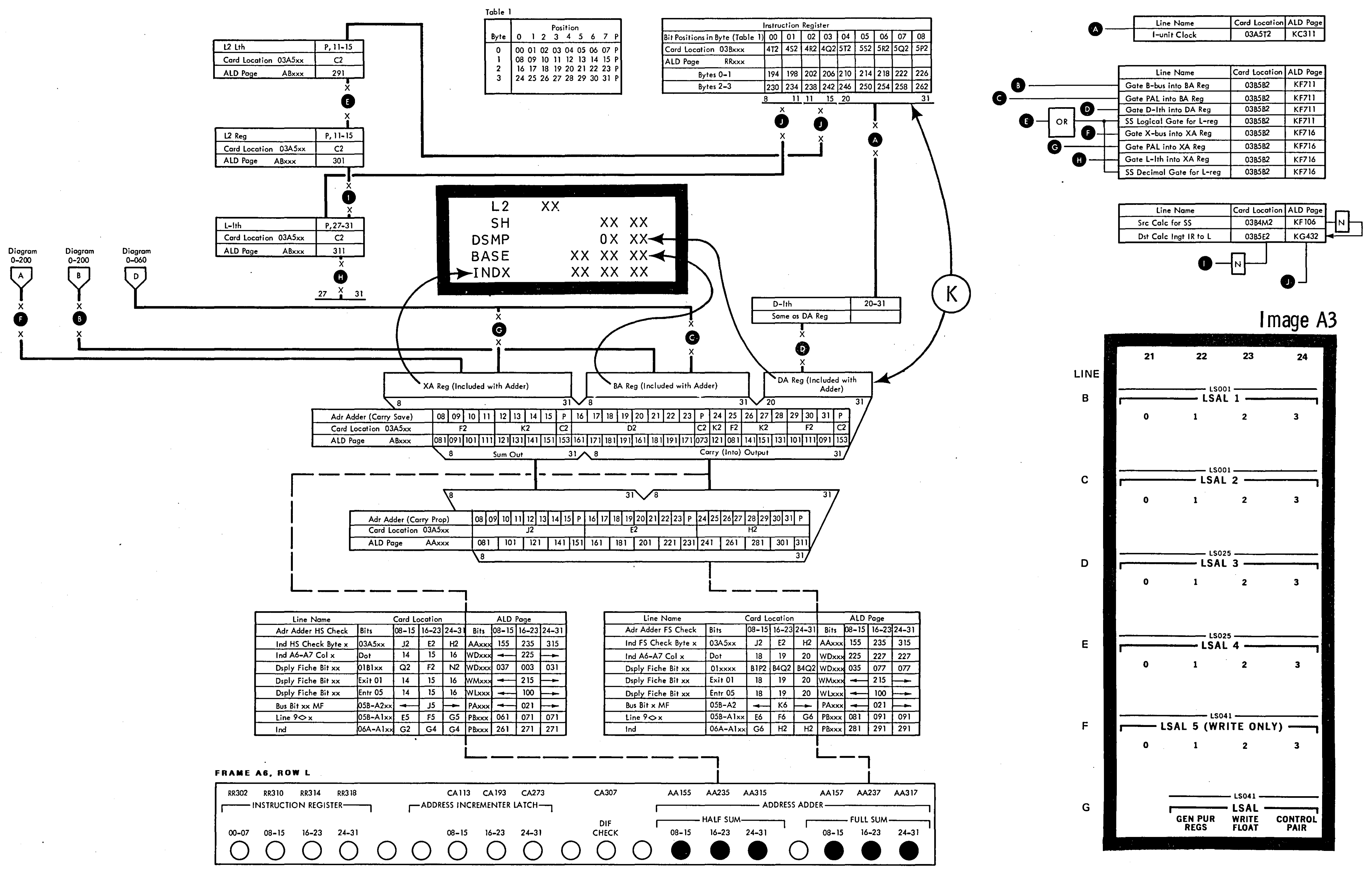
J

Line Name	Card Location	ALD Page
Incr by 0 + 8 + 16	03A4J2	CA373
Decr by − 8 − 16	03A4J2	CA373
Incr or Decr	03A4D2	CA313

Line Name	Card Location				ALD Page			
Ind Adr Incr Err	Bits	08-15	16-23	24-31	Bits	08-15	16-23	24-31
Ind Adr Incr Err	03A4xx	N2	G2	D2	CA	113	193	273
Ind A6-A7 Col xx	Dot	7	8	9	WD	223	223	223
Dsply Fiche Bit xx	01B4xx	G2	H2	J2	WD	047	051	055
Dsply Fiche Bit xx	Exit 01	7	8	9	WM	215	215	215
Dsply Fiche Bit xx	Entr 05	7	8	9	WL	100	100	100
Bus Bit xx MF	05B-A2	J6	J6	J5	PA	011	011	011
Line 9<>xx	05B-A1xx	F3	G3	D4	PB	031	031	041
Ind	06A-A1xx	F2	F2	F4	PB	231	231	241

Line Name	Card Location	ALD Page
Ind Dif Reg Err	03A4H2	RP173
Ind A6-A7 Col 11	Dot	WD223
Dsply Fiche Bit 11	01B4Q2	WD077
Dsply Fiche Bit 11	Exit 01	WM215
Dsply Fiche Bit 11	Entr 05	WL100
Bus Bit 11 MF	05B-A2K6	PA021
Line 9<>11	05B-A1F4	PB051
Ind	06A-A1F6	PB251

Figure 7-18. Address Incrementer ECAD



L2 Lth	P, 11-15
Card Location 03A5xx	C2
ALD Page ABxxx	291

L2 Reg	P, 11-15
Card Location 03A5xx	C2
ALD Page ABxxx	301

L-lth	P,27-31
Card Location 03A5xx	C2
ALD Page ABxxx	311

Table 1

Byte	Position 0	1	2	3	4	5	6	7	P
0	00	01	02	03	04	05	06	07	P
1	08	09	10	11	12	13	14	15	P
2	16	17	18	19	20	21	22	23	P
3	24	25	26	27	28	29	30	31	P

Instruction Register									
Bit Positions in Byte (Table 1)	00	01	02	03	04	05	06	07	08
Card Location 03Bxxx	4T2	4S2	4R2	4Q2	5T2	5S2	5R2	5Q2	5P2
ALD Page RRxxx									
Bytes 0-1	194	198	202	206	210	214	218	222	226
Bytes 2-3	230	234	238	242	246	250	254	258	262

Line Name	Card Location	ALD Page
I-unit Clock	03A5T2	KC311

Line Name	Card Location	ALD Page
Gate B-bus into BA Reg	03B5B2	KF711
Gate PAL into BA Reg	03B5B2	KF711
Gate D-lth into DA Reg	03B5B2	KF711
SS Logical Gate for L-reg	03B5B2	KF711
Gate X-bus into XA Reg	03B5B2	KF716
Gate PAL into XA Reg	03B5B2	KF716
Gate L-lth into XA Reg	03B5B2	KF716
SS Decimal Gate for L-reg	03B5B2	KF716

Line Name	Card Location	ALD Page
Src Calc for SS	03B4M2	KF106
Dst Calc Ingt IR to L	03B5E2	KG432

D-lth	20-31
Same as DA Reg	

Adr Adder (Carry Save)	08	09	10	11	12	13	14	15	P	16	17	18	19	20	21	22	23	P	24	25	26	27	28	29	30	31	P
Card Location 03A5xx	F2				K2				C2	D2								C2	K2	F2	K2			F2			C2
ALD Page ABxxx	081	091	101	111	121	131	141	151	153	161	171	181	191	161	181	191	171	073	121	081	141	151	131	101	111	091	153

Adr Adder (Carry Prop)	08	09	10	11	12	13	14	15	P	16	17	18	19	20	21	22	23	P	24	25	26	27	28	29	30	31	P
Card Location 03A5xx	J2									E2									H2								
ALD Page AAxxx	081		101		121		141		151	161		181		201		221		231	241		261		281		301		311

Line Name	Card Location				ALD Page			
Adr Adder HS Check	Bits	08-15	16-23	24-31	Bits	08-15	16-23	24-31
Ind HS Check Byte x	03A5xx	J2	E2	H2	AAxxx	155	235	315
Ind A6-A7 Col x	Dot	14	15	16	WDxxx	←	225	→
Dsply Fiche Bit xx	01B1xx	Q2	F2	N2	WDxxx	037	003	031
Dsply Fiche Bit xx	Exit 01	14	15	16	WMxxx	←	215	→
Dsply Fiche Bit xx	Entr 05	14	15	16	WLxxx	←	100	→
Bus Bit xx MF	05B-A2xx	←	J5	→	PAxxx	←	021	→
Line 9◇x	05B-A1xx	E5	F5	G5	PBxxx	061	071	071
Ind	06A-A1xx	G2	G4	G4	PBxxx	261	271	271

Line Name	Card Location				ALD Page			
Adr Adder FS Check	Bits	08-15	16-23	24-31	Bits	08-15	16-23	24-31
Ind FS Check Byte x	03A5xx	J2	E2	H2	AAxxx	155	235	315
Ind A6-A7 Col x	Dot	18	19	20	WDxxx	225	227	227
Dsply Fiche Bit xx	01xxxx	B1P2	B4Q2	B4Q2	WDxxx	035	077	077
Dsply Fiche Bit xx	Exit 01	18	19	20	WMxxx	←	215	→
Dsply Fiche Bit xx	Entr 05	18	19	20	WLxxx	←	100	→
Bus Bit x MF	05B-A2	←	K6	→	PAxxx	←	021	→
Line 9◇x	05B-A1xx	E6	F6	G6	PBxxx	081	091	091
Ind	06A-A1xx	G6	H2	H2	PBxxx	281	291	291

FRAME A6, ROW L

INSTRUCTION REGISTER						ADDRESS INCREMENTER LATCH				DIF CHECK		ADDRESS ADDER HALF SUM				ADDRESS ADDER FULL SUM		
RR302	RR310	RR314	RR318			CA113	CA193	CA273		CA307		AA155	AA235	AA315		AA157	AA237	AA317
00-07	08-15	16-23	24-31			08-15	16-23	24-31				08-15	16-23	24-31		08-15	16-23	24-31
○	○	○	○	○	○	○	○	○	○	○	○	●	●	●	○	●	●	●

Figure 7-19. Address Adder ECAD

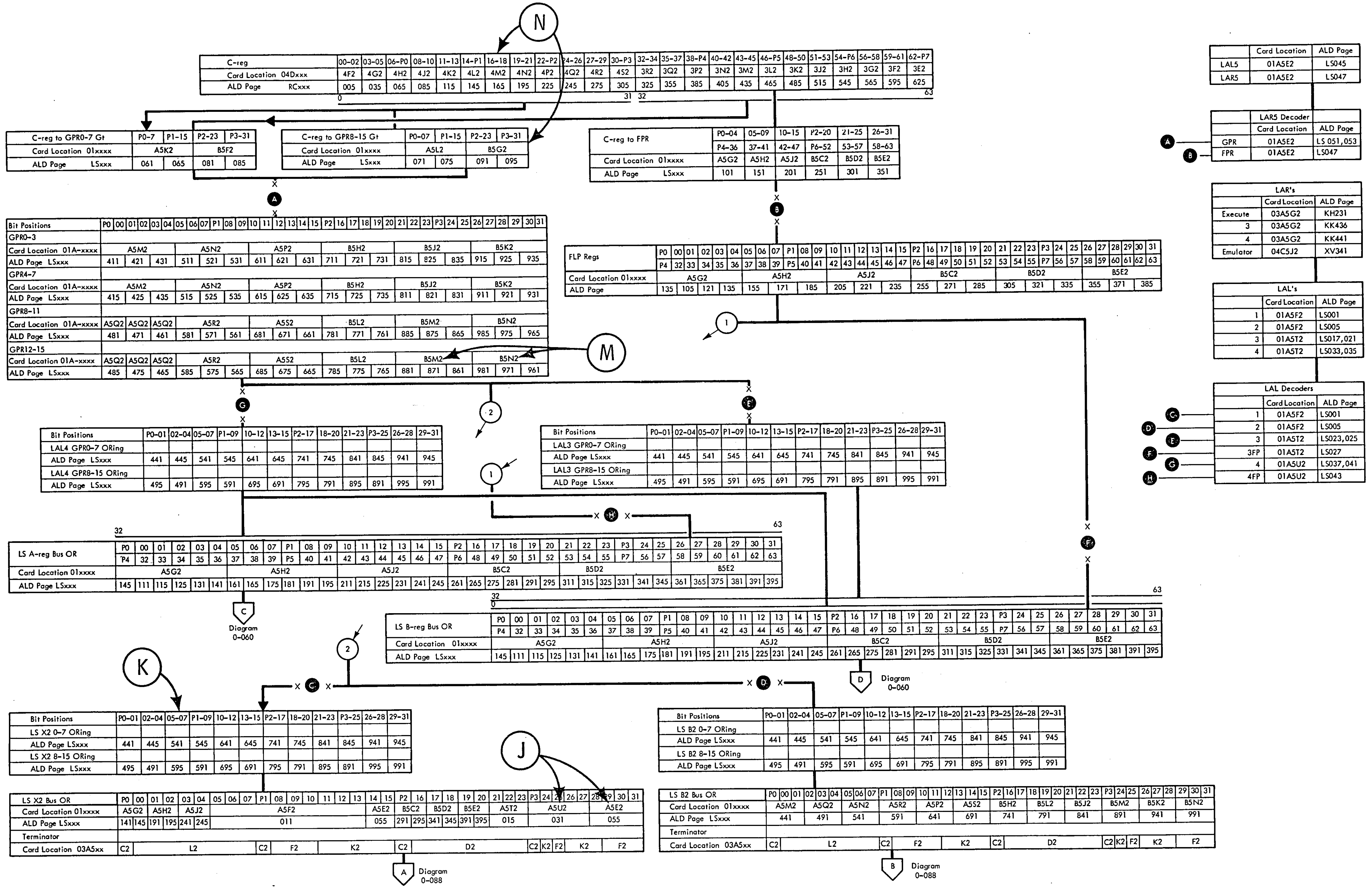


Figure 7-20. LS ECAD

Note 3:
Four card for complete BDBO.

Buf Data Bus Out	P0 07	P1 15	P2 23	P3 31	P4 39	P5 47	P6 55	P7 63
Card Location 03C4xx	E2	F2	G2	H2	E2	F2	G2	H2
ALD Page MQxxx	301-305	307-313	315-321	323-327	331-337	341-347	351-357	361-367

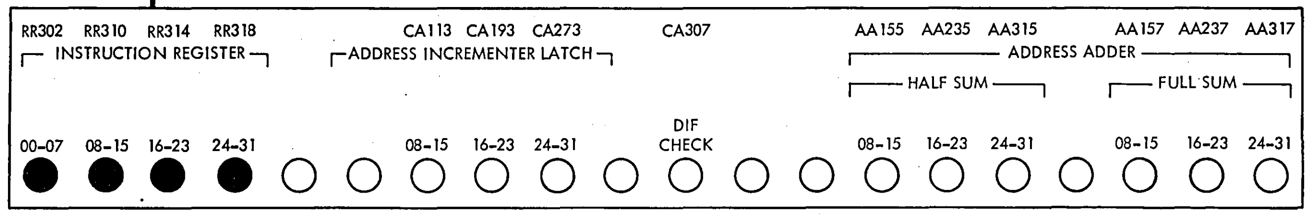
Instruction Buffers, Any Byte, Any Buffer									
Bit Positions in Byte (Table 1)	00	01	02	03	04	05	06	07	P
Card Location 03Bxxx	4T2	4S2	4R2	4Q2	5T2	5S2	5R2	5Q2	5P2
ALD Page RRxxx									
Aux Even	050	054	058	062	066	070	074	078	082
Aux Odd	086	090	094	098	102	106	110	114	118
Main Even	122	126	130	134	138	142	146	150	154
Main Odd	158	162	166	170	174	178	182	186	190

Instruction Register									
Bit Positions in Byte (Table 1)	00	01	02	03	04	05	06	07	P
Card Location 03Bxxx	4T2	4S2	4R2	4Q2	5T2	5S2	5R2	5Q2	5P2
ALD Page RRxxx									
Bytes 0-1	194	198	202	206	210	214	218	222	226
Bytes 2-3	230	234	238	242	246	250	254	258	262

Note 1:
One card contains all four positions.

Line Name	Card Location					ALD Page				
	Bytes	0	1	2	3	Bytes	0	1	2	3
Ind I-reg Error xx	03B5xx	N2	N2	N2	N2	RRxxx	302	310	314	318
Ind A6-A7 Col x	Dot	1	2	3	4	WDxxx	221	221	221	221
Display Fiche Bit xx	01B1xx	G2	H2	J2	K2	WDxxx	007	011	015	019
Display Fiche Bit xx	Exit 01	1	2	3	4	WMxxx	215	215	215	215
Display Fiche Bit xx	Entr 05	1	2	3	4	WL100				
Bus Bit xx MF	05B-A2xx	J6	J6	J6	J6	PA011				
Line 9<>xx	05B-A1xx	D2	E2	F2	G2	PBxxx	001	001	011	011
Ind	06A-A1xx	E2	E2	E4	E4	PBxxx	201	201	211	211

FRAME A6
ROW L



Ⓐ

Line Name	Card Location	ALD Page
Ingate to I-buffers	03B5M2	KE906-921

Ⓑ

Line Name	Card Location	ALD Page
Pointer A Lths	03B5J2	KE117
Pointer B Lths	03B4J2	KE517
Ingate I-reg 0-15	03B4M2	KF116
Ingate I-reg 16-31	03B4M2	KF121

Note 4:
All control is on only four cards.

Table 1

Byte	Position 0	1	2	3	4	5	6	7	P
0	00	01	02	03	04	05	06	07	P
1	08	09	10	11	12	13	14	15	P
2	16	17	18	19	20	21	22	23	P
3	24	25	26	27	28	29	30	31	P
4	32	33	34	35	36	37	38	39	P
5	40	41	42	43	44	45	46	47	P
6	48	49	50	51	52	53	54	55	P
7	56	57	58	59	60	61	62	63	P
8	64	65	66	67	68	69	70	71	P
9	72	73	74	75	76	77	78	79	P
10	80	81	82	83	84	85	86	87	P
11	88	89	90	91	92	93	94	95	P
12	96	97	98	99	100	101	102	103	P
13	104	105	106	107	108	109	110	111	P
14	112	113	114	115	116	117	118	119	P
15	120	121	122	123	124	125	126	127	P

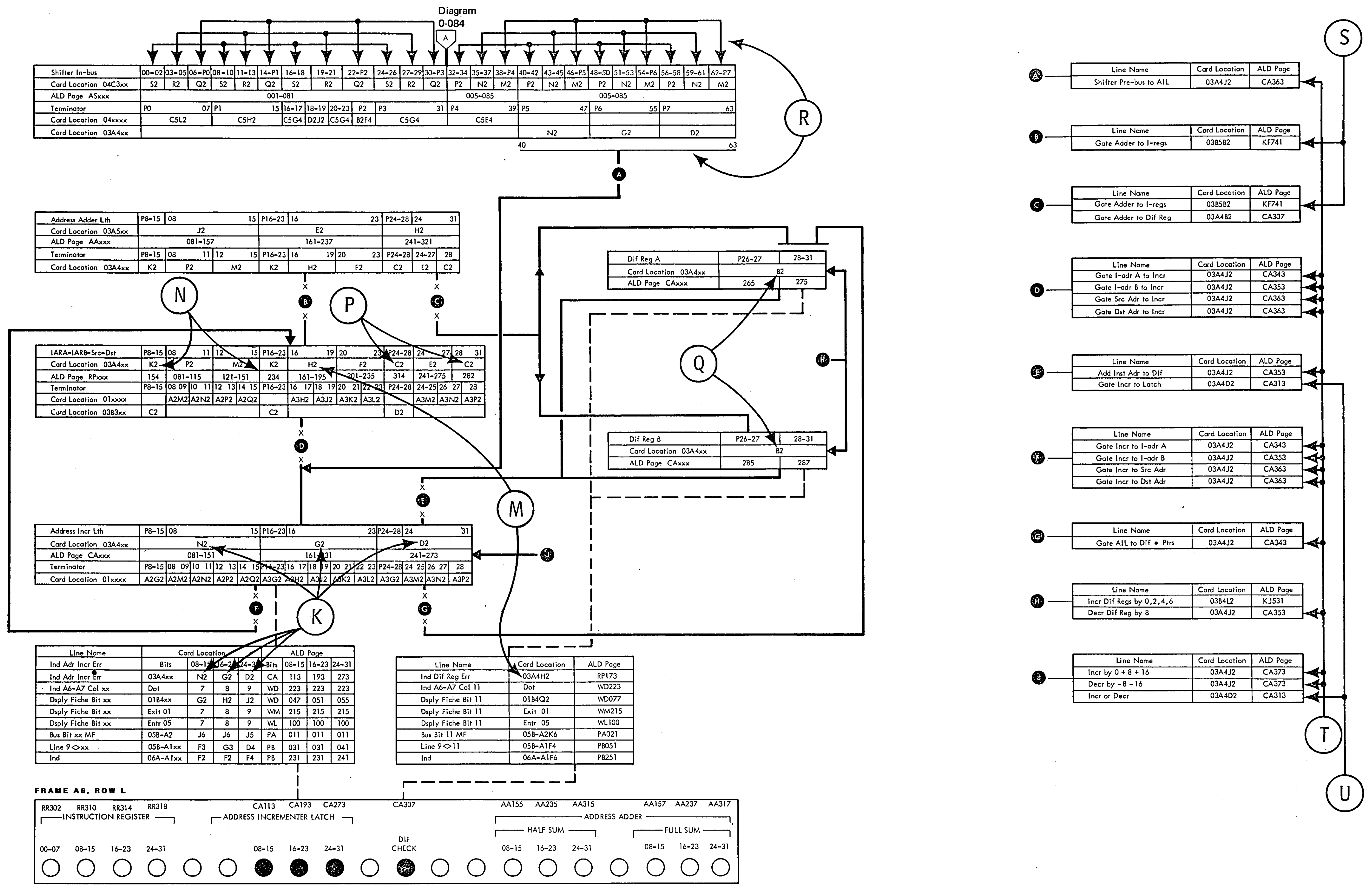
Note: This table shows the position within any byte that any bit of a quadword occupies. For example, bit 91 of a quadword is located in byte 11, bit 03, of that quadword.

To determine which word of the main instruction buffer was gated to the instruction register, look at microfiche frame A6. If the A using main indicator (A6, D17) is on, pointer A triggers (A6, C1-C3) show the word outgated from the instruction buffer. If the A using main indicator is off, pointer B triggers (A6, C16-C18) show the word outgated from the instruction buffer. The pointers are coded as follows:

Pointer	Bits Outgated
000	00-31
001	16-47
010	32-63
011	48-79
100	64-95
101	80-111
110	96-127
111	112-127, 00-15

Note 2:
Changing this card will change bit 7 of every byte in the main I-buffer, the aux I-buffer, and the I-register.

Figure 7-21. I-Buffer Packaging



Shifter In-bus	00-02	03-05	06-P0	08-10	11-13	14-P1	16-18	19-21	22-P2	24-26	27-29	30-P3	32-34	35-37	38-P4	40-42	43-45	46-P5	48-50	51-53	54-P6	56-58	59-61	62-P7
Card Location 04C3xx	S2	R2	Q2	S2	R2	Q2	S2	R2	Q2	S2	R2	Q2	P2	N2	M2	P2	N2	M2	P2	N2	M2	P2	N2	M2
ALD Page ASxxx	001-081												005-085			005-085								

Terminator	P0 07	P1 15	16-17	18-19	20-23	P2	P3 31	P4 39	P5 47	P6 55	P7 63
Card Location 04xxxx	C5L2	C5H2	C5G4	D2J2	C5G4	B2F4	C5G4	C5E4			
Card Location 03A4xx									N2	G2	D2

40 63

Address Adder Lth	P8-15	08		15	P16-23	16		23	P24-28	24	31
Card Location 03A5xx	J2				E2				H2		
ALD Page AAxxx	081-157				161-237				241-321		
Terminator	P8-15	08 11	12 15		P16-23	16 19	20 23		P24-28	24-27	28
Card Location 03A4xx	K2	P2	M2		K2	H2	F2		C2	E2	C2

IARA-IARB-Src-Dst	P8-15	08 11	12 15	P16-23	16 19	20 23	P24-28	24 27	28 31
Card Location 03A4xx	K2	P2	M2	K2	H2	F2	C2	E2	C2
ALD Page RPxxx	154	081-115	121-151	234	161-195	201-235	314	241-275	282

Terminator	P8-15	08 09	10 11	12 13	14 15	P16-23	16 17	18 19	20 21	22 23	P24-28	24-25	26 27	28
Card Location 01xxxx		A2M2	A2N2	A2P2	A2Q2		A3H2	A3J2	A3K2	A3L2		A3M2	A3N2	A3P2
Card Location 03B3xx	C2					C2					D2			

Dif Reg A	P26-27	28-31
Card Location 03A4xx	B2	
ALD Page CAxxx	265	275

Dif Reg B	P26-27	28-31
Card Location 03A4xx	B2	
ALD Page CAxxx	285	287

Address Incr Lth	P8-15	08				15	P16-23	16				23	P24-28	24		31
Card Location 03A4xx	N2						G2						D2			
ALD Page CAxxx	081-151						161-231						241-273			
Terminator	P8-15	08 09	10 11	12 13	14 15		P16-23	16 17	18 19	20 21	22 23		P24-28	24 25	26 27	28
Card Location 01xxxx	A2G2	A2M2	A2N2	A2P2	A2Q2		A3G2	A3H2	A3J2	A3K2	A3L2		A3G2	A3M2	A3N2	A3P2

Line Name	Card Location				ALD Page			
Ind Adr Incr Err	Bits	08-15	16-23	24-31	Bits	08-15	16-23	24-31
Ind Adr Incr Err	03A4xx	N2	G2	D2	CA	113	193	273
Ind A6-A7 Col xx	Dot	7	8	9	WD	223	223	223
Dsply Fiche Bit xx	01B4xx	G2	H2	J2	WD	047	051	055
Dsply Fiche Bit xx	Exit 01	7	8	9	WM	215	215	215
Dsply Fiche Bit xx	Entr 05	7	8	9	WL	100	100	100
Bus Bit xx MF	05B-A2	J6	J6	J5	PA	011	011	011
Line 9◇xx	05B-A1xx	F3	G3	D4	PB	031	031	041
Ind	06A-A1xx	F2	F2	F4	PB	231	231	241

Line Name	Card Location	ALD Page
Ind Dif Reg Err	03A4H2	RP173
Ind A6-A7 Col 11	Dot	WD223
Dsply Fiche Bit 11	01B4Q2	WD077
Dsply Fiche Bit 11	Exit 01	WM215
Dsply Fiche Bit 11	Entr 05	WL100
Bus Bit 11 MF	05B-A2K6	PA021
Line 9◇11	05B-A1F4	PB051
Ind	06A-A1F6	PB251

Line Name	Card Location	ALD Page
Shifter Pre-bus to AIL	03A4J2	CA363

Line Name	Card Location	ALD Page
Gate Adder to I-regs	03B5B2	KF741

Line Name	Card Location	ALD Page
Gate Adder to I-regs	03B5B2	KF741
Gate Adder to Dif Reg	03A4B2	CA307

Line Name	Card Location	ALD Page
Gate I-adr A to Incr	03A4J2	CA343
Gate I-adr B to Incr	03A4J2	CA353
Gate Src Adr to Incr	03A4J2	CA363
Gate Dst Adr to Incr	03A4J2	CA363

Line Name	Card Location	ALD Page
Add Inst Adr to Dif	03A4J2	CA353
Gate Incr to Latch	03A4D2	CA313

Line Name	Card Location	ALD Page
Gate Incr to I-adr A	03A4J2	CA343
Gate Incr to I-adr B	03A4J2	CA353
Gate Incr to Src Adr	03A4J2	CA363
Gate Incr to Dst Adr	03A4J2	CA363

Line Name	Card Location	ALD Page
Gate AIL to Dif • Ptrs	03A4J2	CA343

Line Name	Card Location	ALD Page
Incr Dif Regs by 0,2,4,6	03B4L2	KJ531
Decr Dif Reg by 8	03A4J2	CA353

Line Name	Card Location	ALD Page
Incr by 0 + 8 + 16	03A4J2	CA373
Decr by - 8 - 16	03A4J2	CA373
Incr or Decr	03A4D2	CA313

Figure 7-22. Address Incrementer Packaging

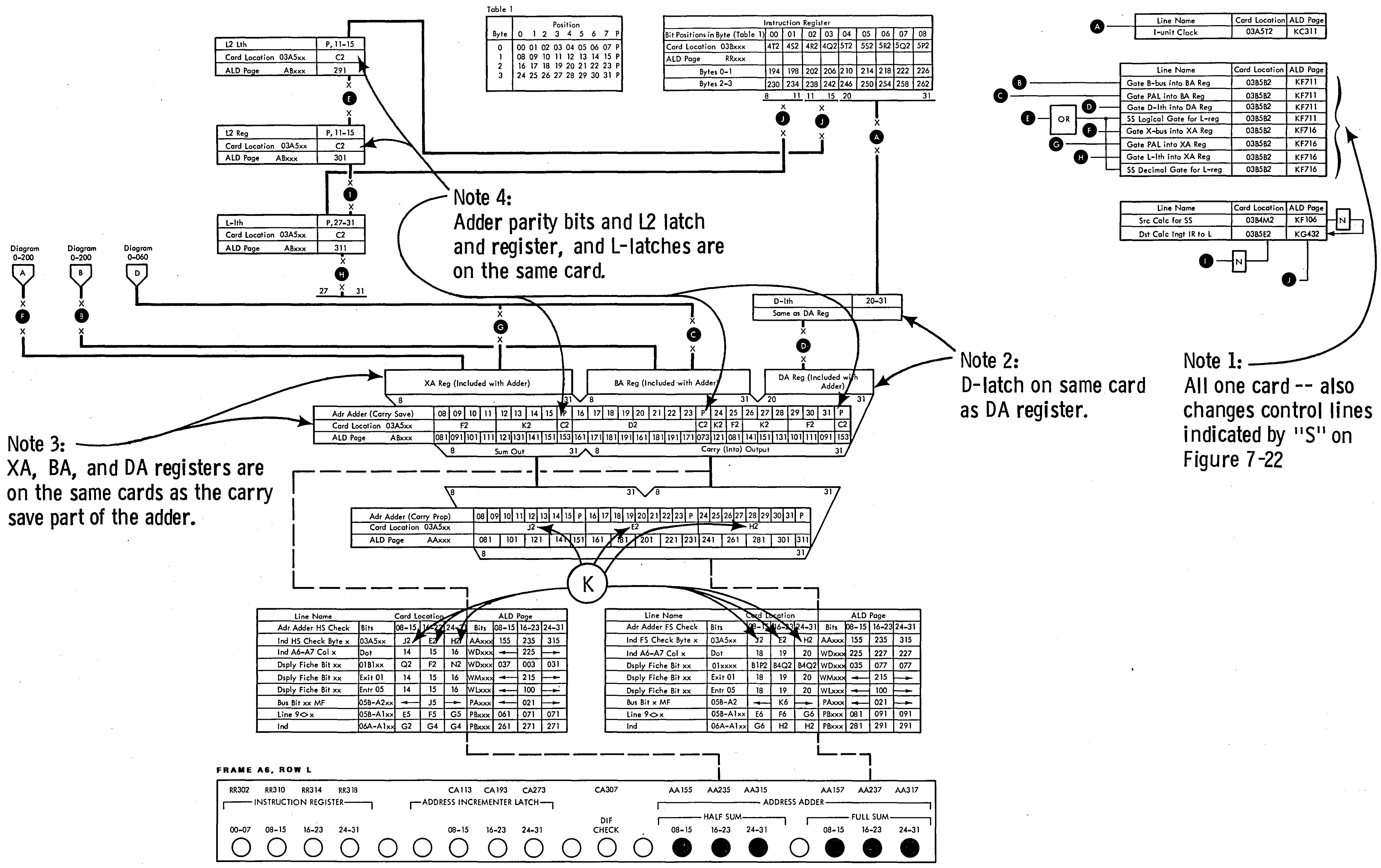


Table 1

Byte	Position 0	1	2	3	4	5	6	7	P
0	00	01	02	03	04	05	06	07	P
1	08	09	10	11	12	13	14	15	P
2	16	17	18	19	20	21	22	23	P
3	24	25	26	27	28	29	30	31	P

Instruction Register									
Bit Positions in Byte (Table 1)	00	01	02	03	04	05	06	07	08
Card Location 03Bxxx	4T2	4S2	4R2	4Q2	5T2	5S2	5R2	5Q2	5P2
ALD Page RRxxx									
Bytes 0-1	194	198	202	206	210	214	218	222	226
Bytes 2-3	230	234	238	242	246	250	254	258	262

L2 Lth	P, 11-15
Card Location 03A5xx	C2
ALD Page ABxxx	291

L2 Reg	P, 11-15
Card Location 03A5xx	C2
ALD Page ABxxx	301

L-lth	P,27-31
Card Location 03A5xx	C2
ALD Page ABxxx	311

Line Name	Card Location	ALD Page
I-unit Clock	03A5T2	KC311

Line Name	Card Location	ALD Page
Gate B-bus into BA Reg	03B5B2	KF711
Gate PAL into BA Reg	03B5B2	KF711
Gate D-lth into DA Reg	03B5B2	KF711
SS Logical Gate for L-reg	03B5B2	KF711
Gate X-bus into XA Reg	03B5B2	KF716
Gate PAL into XA Reg	03B5B2	KF716
Gate L-lth into XA Reg	03B5B2	KF716
SS Decimal Gate for L-reg	03B5B2	KF716

Line Name	Card Location	ALD Page
Src Calc for SS	03B4M2	KF106
Dst Calc Ingt IR to L	03B5E2	KG432

Note 4:
Adder parity bits and L2 latch and register, and L-latches are on the same card.

Note 2:
D-latch on same card as DA register.

Note 1:
All one card -- also changes control lines indicated by "S" on Figure 7-22

Note 3:
XA, BA, and DA registers are on the same cards as the carry save part of the adder.

Adr Adder (Carry Save)	08	09	10	11	12	13	14	15	P	16	17	18	19	20	21	22	23	P	24	25	26	27	28	29	30	31	P
Card Location 03A5xx	F2				K2				C2	D2								C2	K2	F2	K2			F2			C2
ALD Page ABxxx	081	091	101	111	121	131	141	151	153	161	171	181	191	161	181	191	171	073	121	081	141	151	131	101	111	091	153

Adr Adder (Carry Prop)	08	09	10	11	12	13	14	15	P	16	17	18	19	20	21	22	23	P	24	25	26	27	28	29	30	31	P
Card Location 03A5xx	J2									E2									H2								
ALD Page AAxxx	081		101		121		141	151		161		181		201		221	231		241		261		281		301	311	

Line Name	Card Location				ALD Page			
Adr Adder HS Check	Bits	08-15	16-23	24-31	Bits	08-15	16-23	24-31
Ind HS Check Byte x	03A5xx	J2	E2	H2	AAxxx	155	235	315
Ind A6-A7 Col x	Dot	14	15	16	WDxxx	←	225	→
Dsply Fiche Bit xx	01B1xx	Q2	F2	N2	WDxxx	037	003	031
Dsply Fiche Bit xx	Exit 01	14	15	16	WMxxx	←	215	→
Dsply Fiche Bit xx	Entr 05	14	15	16	WLxxx	←	100	→
Bus Bit xx MF	05B-A2xx	←	J5	→	PAxxx	←	021	→
Line 9◇x	05B-A1xx	E5	F5	G5	PBxxx	061	071	071
Ind	06A-A1xx	G2	G4	G4	PBxxx	261	271	271

Line Name	Card Location				ALD Page			
Adr Adder FS Check	Bits	08-15	16-23	24-31	Bits	08-15	16-23	24-31
Ind FS Check Byte x	03A5xx	J2	E2	H2	AAxxx	155	235	315
Ind A6-A7 Col x	Dot	18	19	20	WDxxx	225	227	227
Dsply Fiche Bit xx	01xxxx	B1P2	B4Q2	B4Q2	WDxxx	035	077	077
Dsply Fiche Bit xx	Exit 01	18	19	20	WMxxx	←	215	→
Dsply Fiche Bit xx	Entr 05	18	19	20	WLxxx	←	100	→
Bus Bit x MF	05B-A2	←	K6	→	PAxxx	←	021	→
Line 9◇x	05B-A1xx	E6	F6	G6	PBxxx	081	091	091
Ind	06A-A1xx	G6	H2	H2	PBxxx	281	291	291

FRAME A6, ROW L

RR302	RR310	RR314	RR318			CA113	CA193	CA273		CA307		AA155	AA235	AA315		AA157	AA237	AA317
INSTRUCTION REGISTER						ADDRESS INCREMENTER LATCH						ADDRESS ADDER						
												HALF SUM				FULL SUM		
00-07	08-15	16-23	24-31			08-15	16-23	24-31		DIF CHECK		08-15	16-23	24-31		08-15	16-23	24-31
○	○	○	○	○	○	○	○	○	○	○	○	●	●	●	○	●	●	●

Figure 7-23. Address Adder Packaging

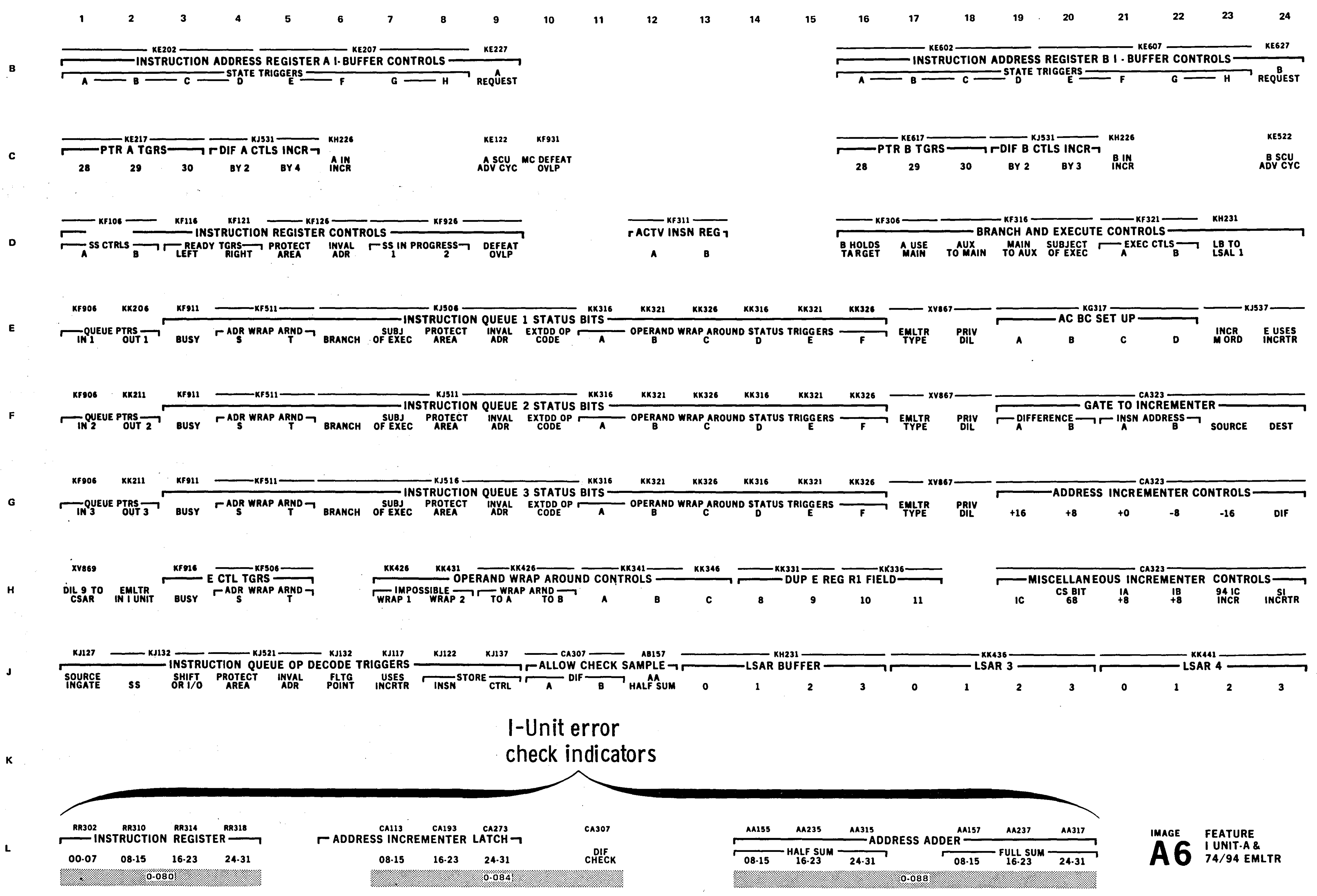

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
B
KE202 KE207 KE227
INSTRUCTION ADDRESS REGISTER A I-BUFFER CONTROLS
STATE TRIGGERS
A B C D E F G H A REQUEST
KE602 KE607 KE627
INSTRUCTION ADDRESS REGISTER B I-BUFFER CONTROLS
STATE TRIGGERS
A B C D E F G H B REQUEST
C
KE217 KJ531 KH226 KE122 KF931
PTR A TGRS DIF A CTLS INCR
28 29 30 BY 2 BY 4 A IN INCR A SCU ADV CYC MC DEFEAT OVLP
KE617 KJ531 KH226 KE522
PTR B TGRS DIF B CTLS INCR
28 29 30 BY 2 BY 3 B IN INCR B SCU ADV CYC
D
KF106 KF116 KF121 KF126 KF926
INSTRUCTION REGISTER CONTROLS
SS CTRLS A B READY TGRS LEFT RIGHT PROTECT AREA INVAL ADR SS IN PROGRESS 1 2 DEFEAT OVLP
KF311
ACTV INSN REG A B
KF306 KF316 KF321 KH231
BRANCH AND EXECUTE CONTROLS
B HOLDS TARGET A USE MAIN AUX TO MAIN MAIN TO AUX SUBJECT OF EXEC EXEC CTLS A B LB TO LSAL 1
E
KF906 KK206 KF911 KF511 KJ506 KK316 KK321 KK326 KK316 KK321 KK326 XV867 KG317 KJ537
INSTRUCTION QUEUE 1 STATUS BITS
QUEUE PTRS IN 1 OUT 1 BUSY ADR WRAP ARND S T BRANCH SUBJ OF EXEC PROTECT AREA INVAL ADR EXTDD OP CODE
OPERAND WRAP AROUND STATUS TRIGGERS A B C D E F
EMLTR TYPE PRIV DIL
AC BC SET UP A B C D
INCR M ORD E USES INCRTR
F
KF906 KK211 KF911 KF511 KJ511 KK316 KK321 KK326 KK316 KK321 KK326 XV867 CA323
INSTRUCTION QUEUE 2 STATUS BITS
QUEUE PTRS IN 2 OUT 2 BUSY ADR WRAP ARND S T BRANCH SUBJ OF EXEC PROTECT AREA INVAL ADR EXTDD OP CODE
OPERAND WRAP AROUND STATUS TRIGGERS A B C D E F
EMLTR TYPE PRIV DIL
GATE TO INCREMENTER
DIFFERENCE A B INSN ADDRESS A B SOURCE DEST
G
KF906 KK211 KF911 KF511 KJ516 KK316 KK321 KK326 KK316 KK321 KK326 XV867 CA323
INSTRUCTION QUEUE 3 STATUS BITS
QUEUE PTRS IN 3 OUT 3 BUSY ADR WRAP ARND S T BRANCH SUBJ OF EXEC PROTECT AREA INVAL ADR EXTDD OP CODE
OPERAND WRAP AROUND STATUS TRIGGERS A B C D E F
EMLTR TYPE PRIV DIL
ADDRESS INCREMENTER CONTROLS
+16 +8 +0 -8 -16 DIF
H
XV869 KF916 KF506 KK426 KK431 KK426 KK341 KK346 KK331 KK336
DIL 9 TO CSAR EMLTR IN I UNIT BUSY
E CTL TGRS ADR WRAP ARND S T
OPERAND WRAP AROUND CONTROLS
IMPOSSIBLE WRAP 1 WRAP 2 WRAP ARND TO A TO B A B C
DUP E REG R1 FIELD 8 9 10 11
CA323
MISCELLANEOUS INCREMENTER CONTROLS
IC CS BIT 68 IA +8 IB +8 94 IC INCR SI INCRTR
J
KJ127 KJ132 KJ521 KJ132 KJ117 KJ122 KJ137 CA307 AB157 KH231 KK436 KK441
INSTRUCTION QUEUE OP DECODE TRIGGERS
SOURCE INGATE SS SHIFT OR I/O PROTECT AREA INVAL ADR FLTG POINT USES INCRTR STORE INSN CTRL
ALLOW CHECK SAMPLE DIF A B AA HALF SUM
LSAR BUFFER 0 1 2 3
LSAR 3 0 1 2 3
LSAR 4 0 1 2 3
K
I-Unit error check indicators
L
RR302 RR310 RR314 RR318
INSTRUCTION REGISTER
00-07 08-15 16-23 24-31
0-080
CA113 CA193 CA273
ADDRESS INCREMENTER LATCH
08-15 16-23 24-31
0-084
CA307
DIF CHECK
AA155 AA235 AA315 AA157 AA237 AA317
ADDRESS ADDER
HALF SUM 08-15 16-23 24-31
FULL SUM 08-15 16-23 24-31
0-088
IMAGE A6
FEATURE I UNIT-A & 74/94 EMLTR

Figure 7-24. I-Unit Error Lights

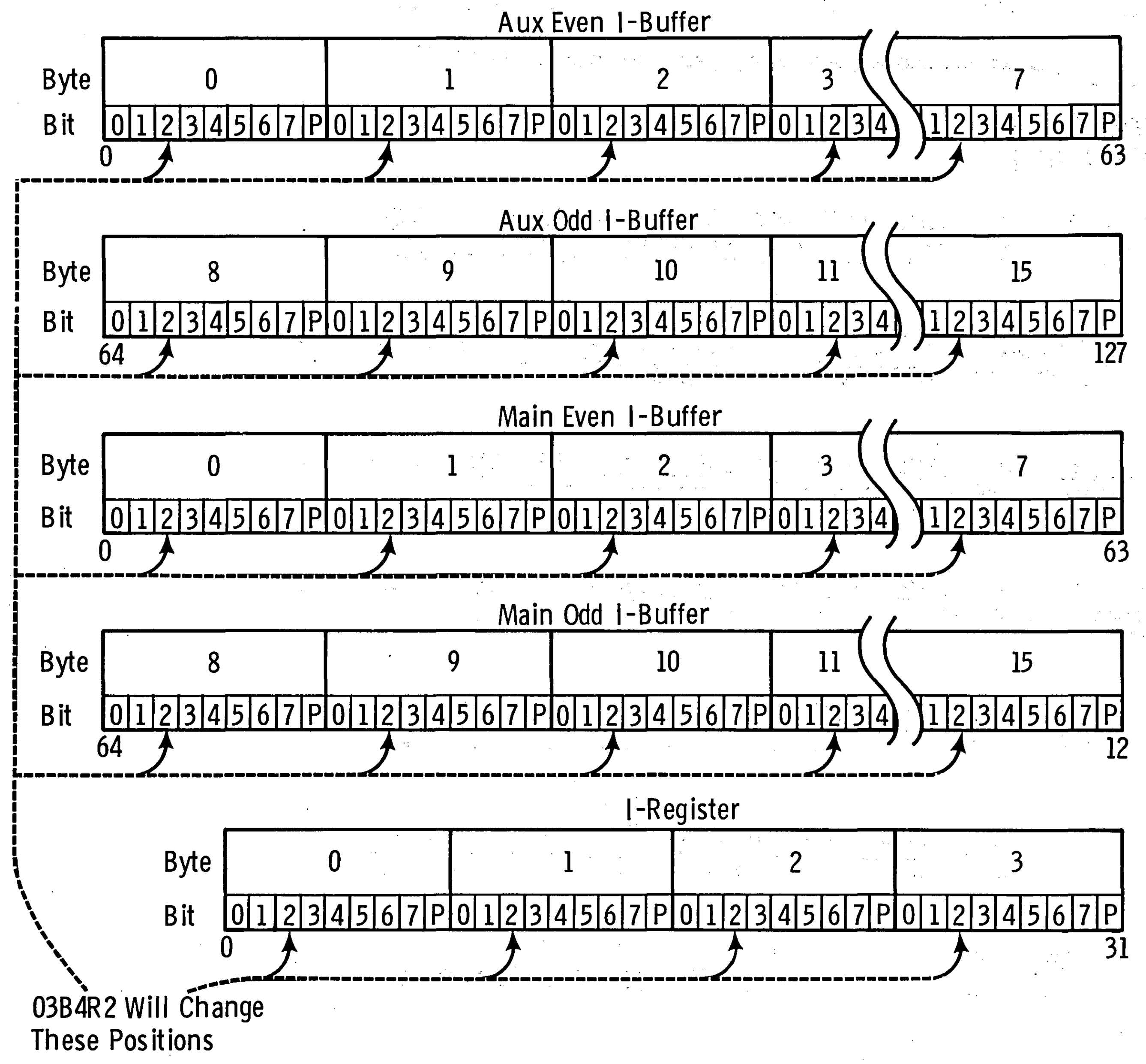


Figure 7-25 I-Buffer Functional Packaging

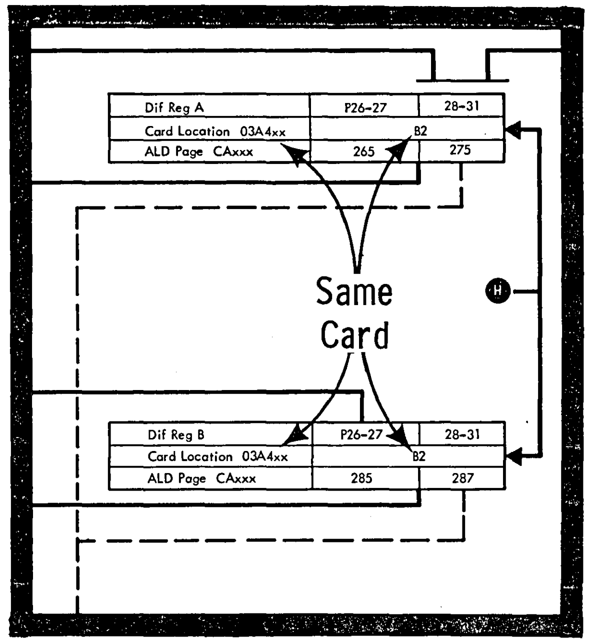
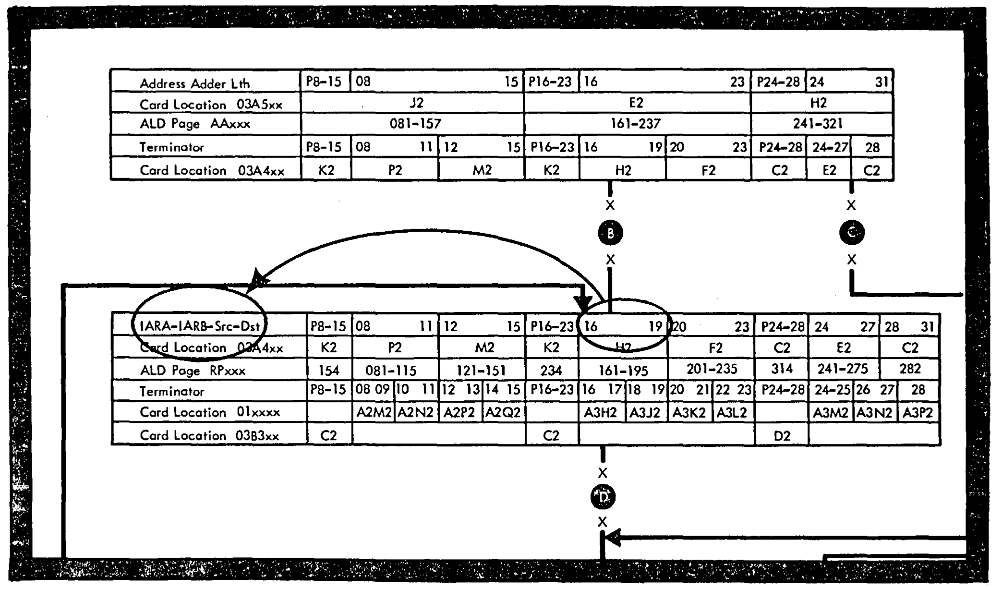


Figure 7-26. Session 7, Question 8 Remedial


Address Adder Lth | P8-15 | 08 15 | P16-23 | 16 23 | P24-28 | 24 31
Card Location 03A5xx | J2 | E2 | H2
ALD Page AAxxx | 081-157 | 161-237 | 241-321
Terminator | P8-15 | 08 11 | 12 15 | P16-23 | 16 19 | 20 23 | P24-28 | 24-27 | 28
Card Location 03A4xx | K2 | P2 | M2 | K2 | H2 | F2 | C2 | E2 | C2
B
C
IARA-IARB-Src-Dst | P8-15 | 08 11 | 12 15 | P16-23 | 16 19 | 20 23 | P24-28 | 24 27 | 28 31
Card Location 03A4xx | K2 | P2 | M2 | K2 | H2 | F2 | C2 | E2 | C2
ALD Page RPxxx | 154 | 081-115 | 121-151 | 234 | 161-195 | 201-235 | 314 | 241-275 | 282
Terminator | P8-15 | 08 09 | 10 11 | 12 13 | 14 15 | P16-23 | 16 17 | 18 19 | 20 21 | 22 23 | P24-28 | 24-25 | 26 27 | 28
Card Location 01xxxx | A2M2 | A2N2 | A2P2 | A2Q2 | A3H2 | A3J2 | A3K2 | A3L2 | A3M2 | A3N2 | A3P2
Card Location 03B3xx | C2 | C2 | D2
D

This tells you that bits 16 to 19 of the SRC and DST registers, and IARA and IARB, are all on the same card.

Figure 7-27. Session 7, Question 7 Remedial

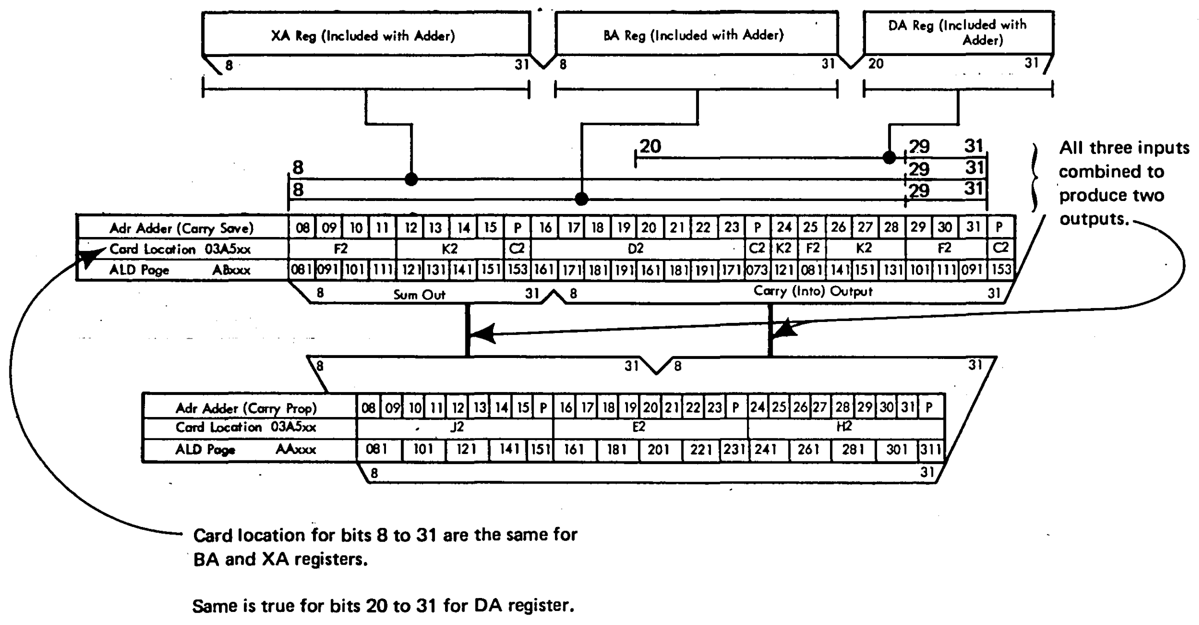

XA Reg (Included with Adder)
BA Reg (Included with Adder)
DA Reg (Included with Adder)
8 31 8 31 20 31
20 29 31
8 29 31
8 29 31
All three inputs combined to produce two outputs.
Adr Adder (Carry Save) 08 09 10 11 12 13 14 15 P 16 17 18 19 20 21 22 23 P 24 25 26 27 28 29 30 31 P
Card Location 03A5xx F2 K2 C2 D2 C2 K2 F2 K2 F2 C2
ALD Page ABxxx 081 091 101 111 121 131 141 151 153 161 171 181 191 161 181 191 171 073 121 081 141 151 131 101 111 091 153
8 Sum Out 31 8 Carry (Into) Output 31
8 31 8 31
Adr Adder (Carry Prop) 08 09 10 11 12 13 14 15 P 16 17 18 19 20 21 22 23 P 24 25 26 27 28 29 30 31 P
Card Location 03A5xx J2 E2 H2
ALD Page AAxxx 081 101 121 141 151 161 181 201 221 231 241 261 281 301 311
8 31
Card location for bits 8 to 31 are the same for BA and XA registers.
Same is true for bits 20 to 31 for DA register.

Figure 7-28. Session 7, Question 9 Remedial

PARTIAL LIST OF SYSTEM LOGIC UNITS

1. DBAR
2. Address Adder
3. LAR Buffer Latch
4. ACAL
5. MCDR
6. Buffer Bypass Latch
7. L2 Register
8. Parallel Adder
9. Channel Data-Out Buffer
10. A-Register
11. Shift Control Triggers
12. Destination Register
13. Main I-Buffer
14. A-Pointer
15. C-Register
16. STAR
17. FAR
18. Source Register
19. LAL5
20. F-Register
21. Instruction Queues
22. Serial Adder
23. MCWR
24. Difference Register B
25. CSAR
26. B-Register

Figure 8-1. Sesion 8, Question 1

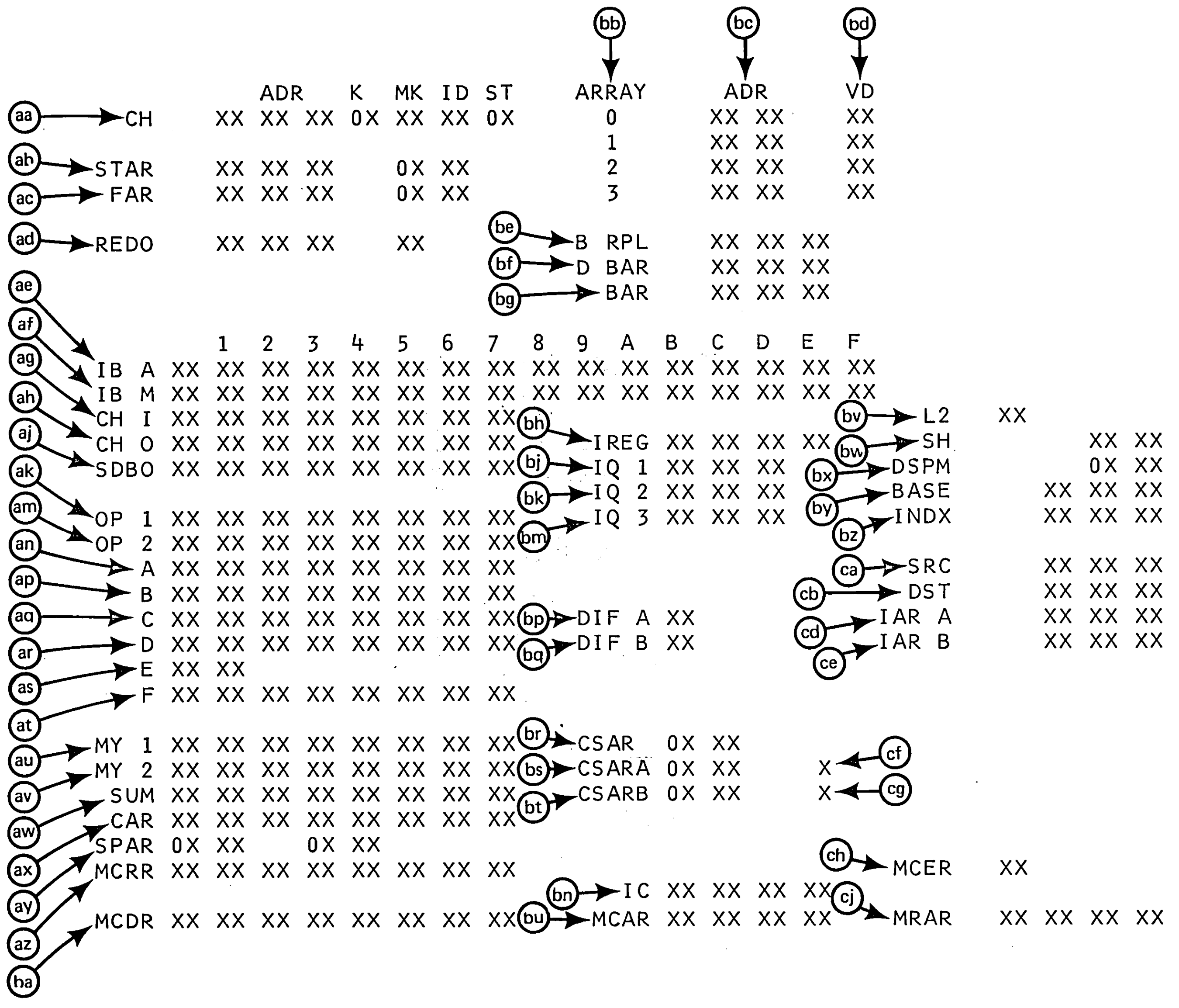


Figure 8-2. Session 8, Questions 10 & 11

RED-LIGHT ERRORS

1. Store Check
2. Hang Detect
3. Parallel Adder Halfsum Check
4. Parallel Adder Fullsum Check
5. Difference Check
6. I-Register Check
7. Address Adder Fullsum
8. Serial Adder Halfsum
9. E-Register Check
10. Address Adder Halfsum
11. Serial Adder Fullsum
12. Parallel Adder Carry Check
13. Parallel Adder Halfsum Word Check
14. Outkey Check
15. Buffer Array Check
16. Shifter Input Check
17. Address Incrementer Latch Check
18. Shifter Output Check
19. Non-Retry Storage Check
20. Shifter Control Check
21. Address Storage Check
22. Enter Error

Figure 8-3. Session 8, Questions 19 & 20

1. Op 1 Register
2. CSAR
3. Shifter-In Bus
4. D-Register
5. C40 ROS
6. C50 WCS
7. Output Latch
8. PAL
9. CSDR
10. C-Register
11. LAL3
12. E-Register
13. CSAR A
14. ACAL
15. Mark Buffer Latch

Figure 8-4. Session 8, Question 22

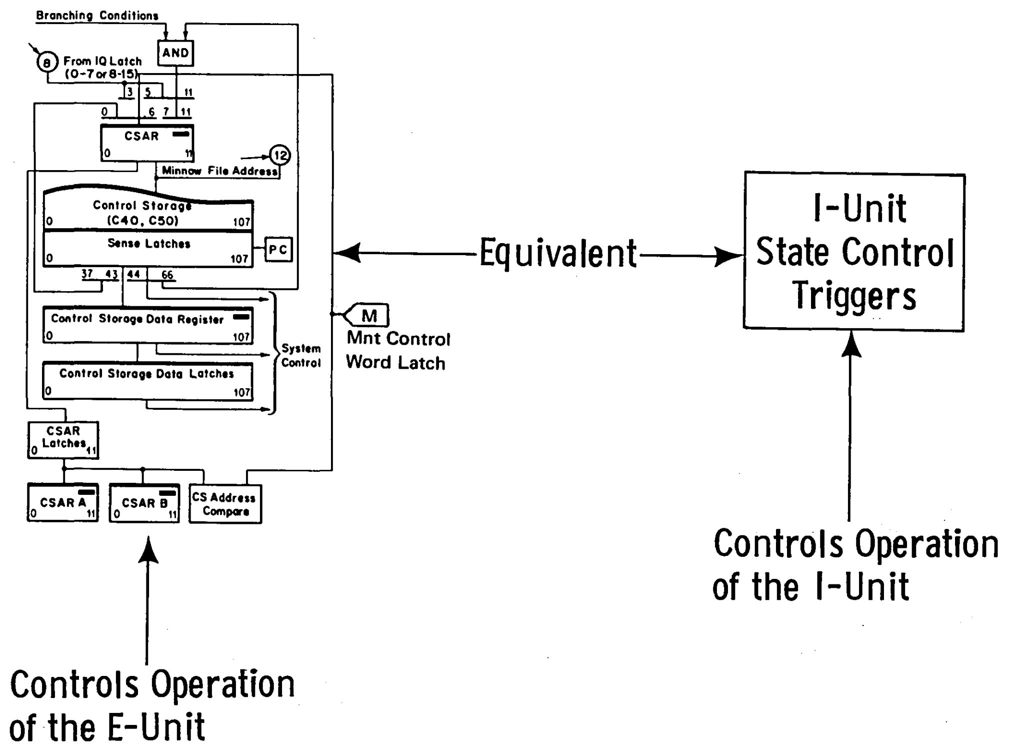

Branching Conditions
AND
8 From IQ Latch (0-7 or 8-15)
3
5
11
0
6
7
11
CSAR
0
11
12
Minnow File Address
Control Storage (C40, C50)
0
107
Sense Latches
0
107
PC
37
43
44
66
Control Storage Data Register
0
107
Control Storage Data Latches
0
107
System Control
M
Mnt Control Word Latch
CSAR Latches
0
11
CSAR A
0
11
CSAR B
0
11
CS Address Compare
Controls Operation of the E-Unit
Equivalent
I-Unit State Control Triggers
Controls Operation of the I-Unit

Figure 8-5. E And I-Unit Control Areas

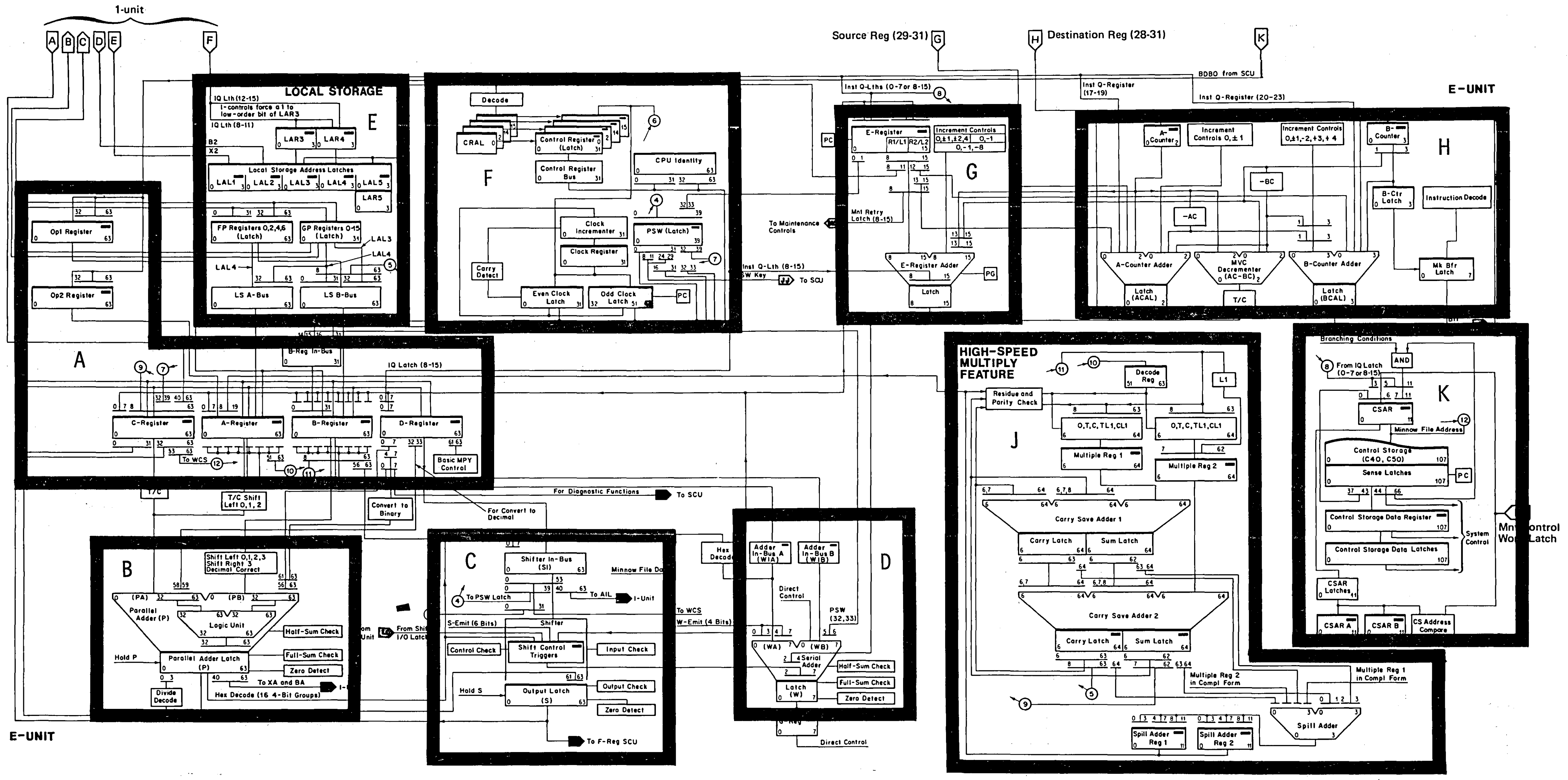


Figure 8-6. E-Unit Logic Units

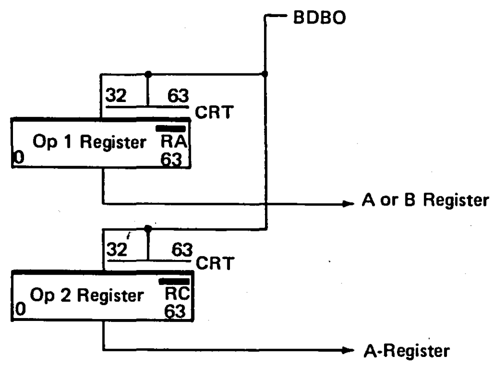


Figure 8-7. Operand Registers

OPERAND REGISTERS - OP 1, OP 2

The two eight-byte operand registers, referred to as operand register 1 and operand register 2, are used to buffer prefetched storage operands.

The only ingate to the buffers is the buffer data bus-out (BDBO), which may be gated directly or shifted right 32. The shift allows the I-unit to place all fixed-point and floating-point short operands in the right half of the buffers.

Ingating data to the registers is done by the I-unit.

Transfer of data from the operand registers to the working registers is dependent on the operation performed and upon which operand register contains the active data.

Operand buffer 1 can be gated to the A- or B-register; operand buffer 2 can be gated only to the A-register. This allows the I-unit to use either buffer for RX instruction setup. For SS instructions, operand buffer 1 is used solely for destination operands and operand buffer 2 is used only for source operands.

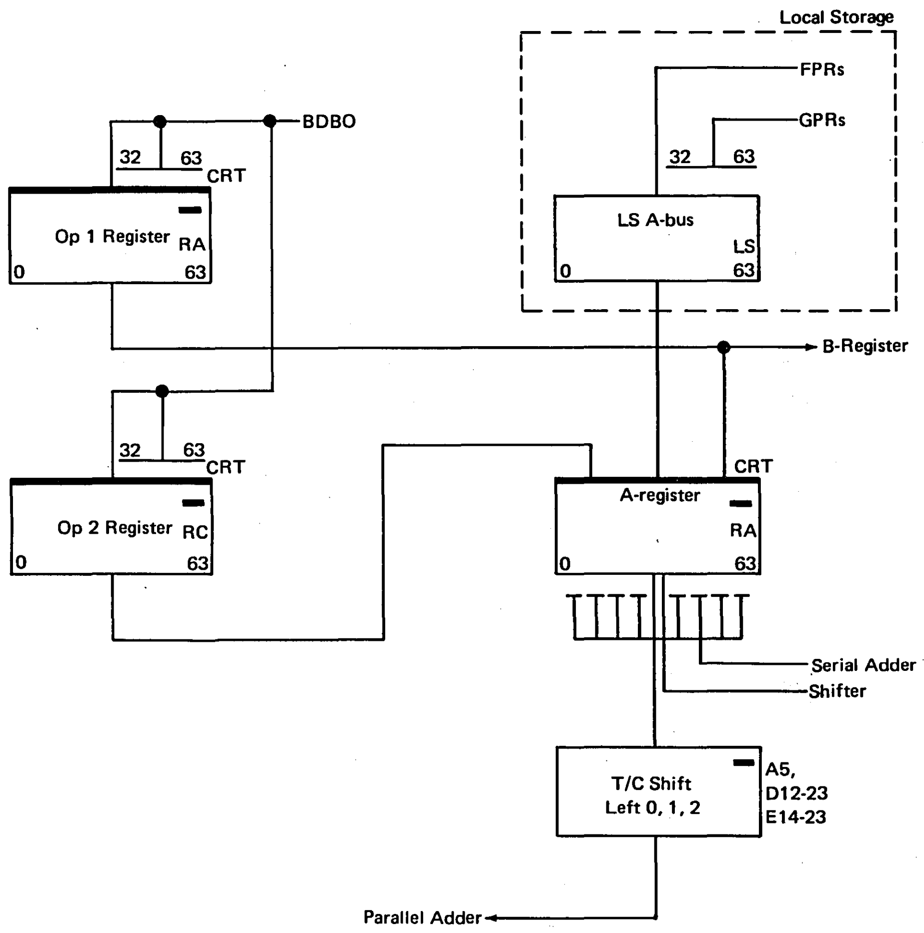


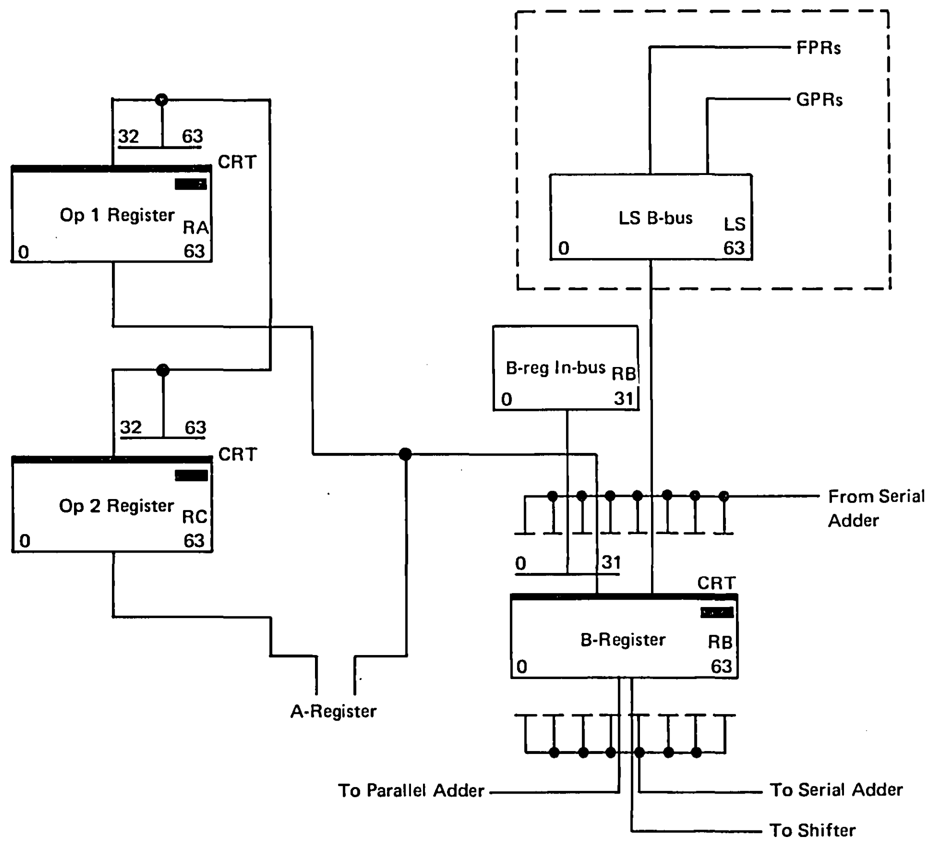
Figure 8-8. A-Register

A-REGISTER

The eight-byte A-register is used primarily as the second operand register. The second operand is the contents of a local storage register, or the source operand for RR, RX, and SS formats.

Ingates to the A-register include the parallel adder, shifter, operand register 1, operand register 2, and local storage. The serial adder latch may be gated into the high-order byte of the A-register to handle floating-point characteristics.

The A-register can be outgated to the parallel adder (true or complement form) with shifts of 0, 1, or 2. The A-register also feeds the shifter and contains byte gates to the serial adder.



B REGISTER

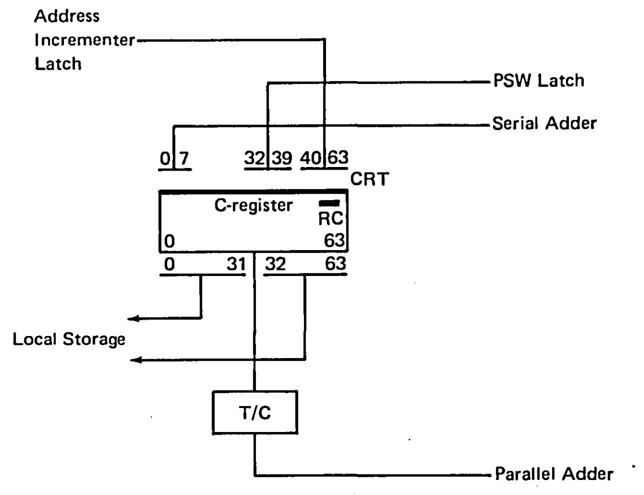
The eight-byte B-register is used primarily as the first operand register. The first operand is the content of a local storage register for RR and RX formats and the destination field for SS format.

The ingates to the B-register include the parallel adder, the shifter, operand register 1, local storage, the B-reg prebus and the byte ingates from the serial adder.

The B-reg prebus is a 32-bit bus to the high-order 32-bits of the B-register. This bus accommodates inputs from the PSW, the time of day clock, the control registers, the CPU identification and the machine check interruption code.

The B-register can be outgated to the parallel adder with a "right three" shift or a left shift of 0, 1, 2, and 3. The B-register also feeds the shifter and the serial adder by byte.

Figure 8-9. B-Register



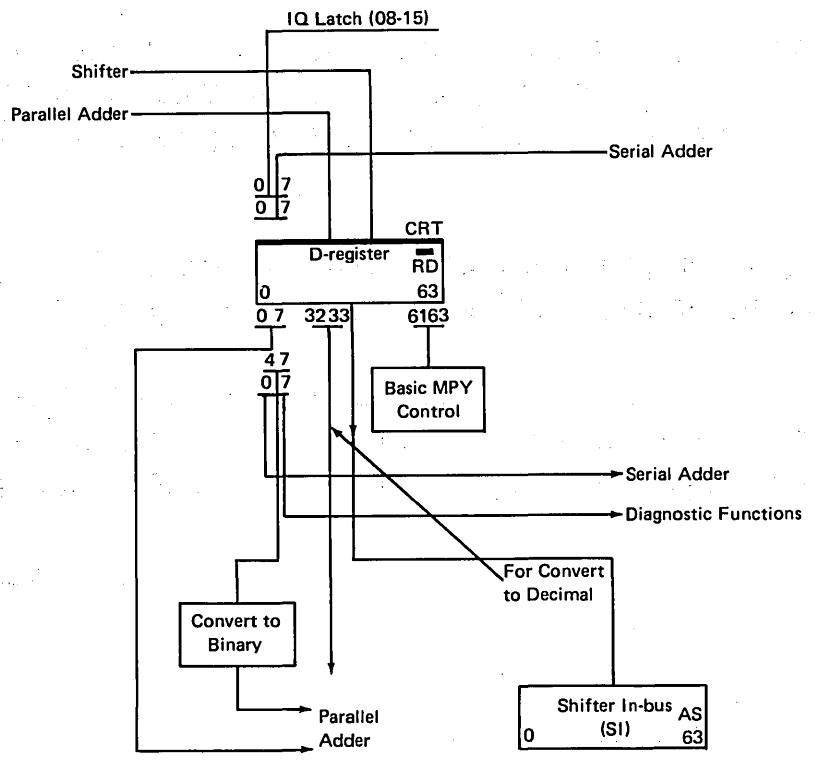
C-REGISTER

The eight-byte C-register is used primarily as a buffer facility to input data to the local storage. For those RR and RX format instructions that require a result to be placed in local storage, the result is held in the C-register and written in local storage during the first cycle of the next instruction. The C-register is used in fixed- and floating-point multiply to hold a multiple of the multiplicand.

The ingates to the C-register include the parallel adder, shifter, the I-unit address incrementer latch (used for Load Address, PSW handling, VFL instructions, and such, and the serial adder to the high-order byte (used in floating-point characteristic handling).

The C-register can be outgated to local storage and (in true or complement form) to the parallel adder.

Figure 8-10. C-Register

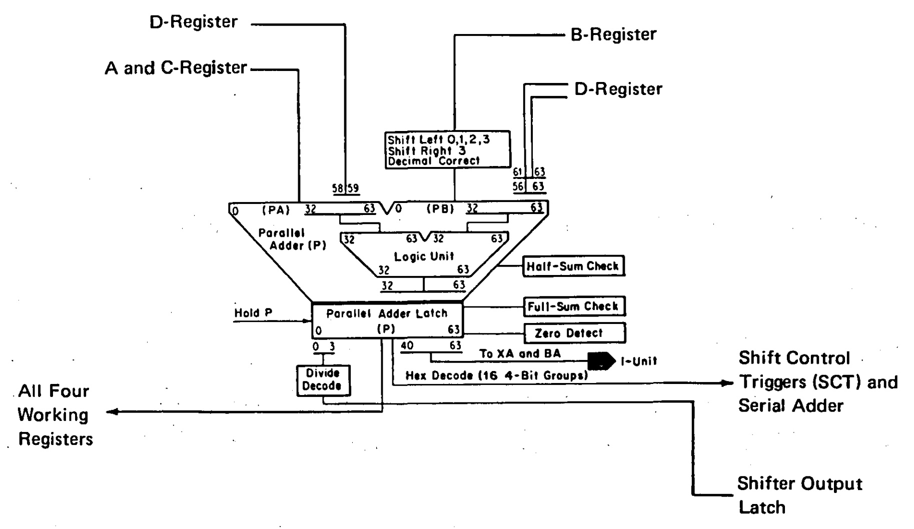


D-REGISTER

The eight-byte D-register is used for multiplier decoding in fixed- and floating-point multiply and as the quotient accumulator in divide instructions. The high-order byte may be ingated from and outgated to the serial adder and serves as a byte buffer for serial adder operations.

The ingates to the D-register include the parallel adder, shifter, and the serial adder (to the high-order byte). The high-order byte may be gated to the serial adder as previously mentioned.

Figure 8-11. D-Register



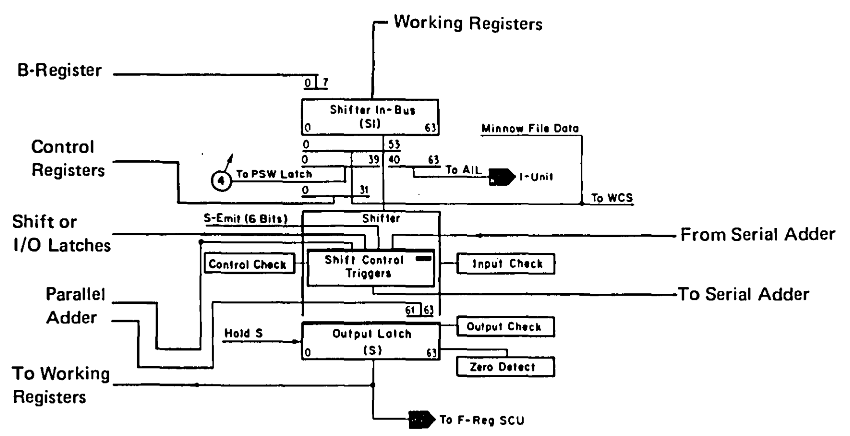
PARALLEL ADDER

The parallel adder is a 64-bit (plus parity) full-binary arithmetic unit. In addition to arithmetic operations, the parallel adder performs the following operations: four-byte-wide logical (AND, OR, exclusive OR), algebraic compare, convert-to-decimal (CVD), convert-to-binary (CVB), and simple data transfer. Error-checking facilities within the adder consist of: (1) a halfsum check that verifies the validity of the incoming data (operands) and (2) a fullsum check that verifies the validity of the adder result. As part of the fullsum check, parity is predicted on a byte basis. These parity bits accompany the adder result to its destination.

Important points concerning the adder are:

- Is 64-bit (plus parity) full-binary adder.
- Includes halfsum and fullsum error checking.
- Inputs are from A-, B-, C-, and D-registers.
- Outputs go to shifter, serial adder, address adder, and A-, B-, C-, and D-registers.
- Operation rate is one operation per machine cycle.
- Adder operates every machine cycle. If no register data is gated into the adder, zero bytes with good parity are forced through the adder.

Figure 8-12. Parallel Adder



SHIFTER

An eight-byte shifter in the E-unit displaces the bits of an operand from 0-63 positions to the right or left for fast data alignment, multiplication, and division. The shifter is also used as a "feed-thru" device to transfer data from one arithmetic register to another. It performs left or right, arithmetic, or logical shifts. If a left shift is given, bits are moved from low-order to high-order positions, and vice versa for a right shift. On arithmetic shifts, the sign bit is propagated as the operand is shifted; during logical shifts, the sign is shifted in the same manner as other bits (zeros fill in vacated positions).

The shifter requires only one machine cycle to perform a shift operation, regardless of the number of positions shifted. Thus, any Shift instruction can be executed within one CS cycle.

The shifter is logically divided into six stages: shifter in-bus, stage A, stage B, stage C, stage D, and the shifter output latches. The shifter in-bus (SI) provides the ingating from the E-unit arithmetic registers. Shifter-stages A-D perform the actual shifting operations. Data is gated to SI, filters through the shifter stages, and is then set into the output latches. Each stage shifts the data a different amount. The contents of the shifter output latches are gated to the A-, B-, C-, D-, or F-register.

Figure 8-13. Shifter

SHIFT CONTROL TRIGGERS (SCTs)

The direction (left or right) of shifts, the number of positions shifted, and sign propagation are controlled by shift control triggers (SCTs). The SCTs are a "left shift" trigger which is on for left shifts and off for right shifts, a "shift single arithmetic" trigger which is on for a word-length operand during an arithmetic shift operation, a "shift double arithmetic" trigger which is on for a doubleword length operand during arithmetic shifts, and three groups of four SCTs which control the number of positions shifted. Each group of the last-mentioned SCTs is associated with a certain stage of the shifter. The SCTs associated with stage B cause the operand to be shifted 0, 1, 2, or 3 positions in stage B; the SCTs associated with stage C cause the operand to be shifted 0, 4, 8, or 12 positions in stage C; and the SCTs associated with stage D cause the operand to be shifted 0, 16, 32, or 48 positions in stage D.

The SCTs are set up by micro-orders that specify an operation to be performed by the shifter. For most shifts, the shift amount is ingated from a specific register, is decoded, and is sent to stage B, C, and D SCTs. One SCT is set in each of these three groups. The arithmetic and left shift triggers are set according to the operation.

Figure 8-14. SCTs

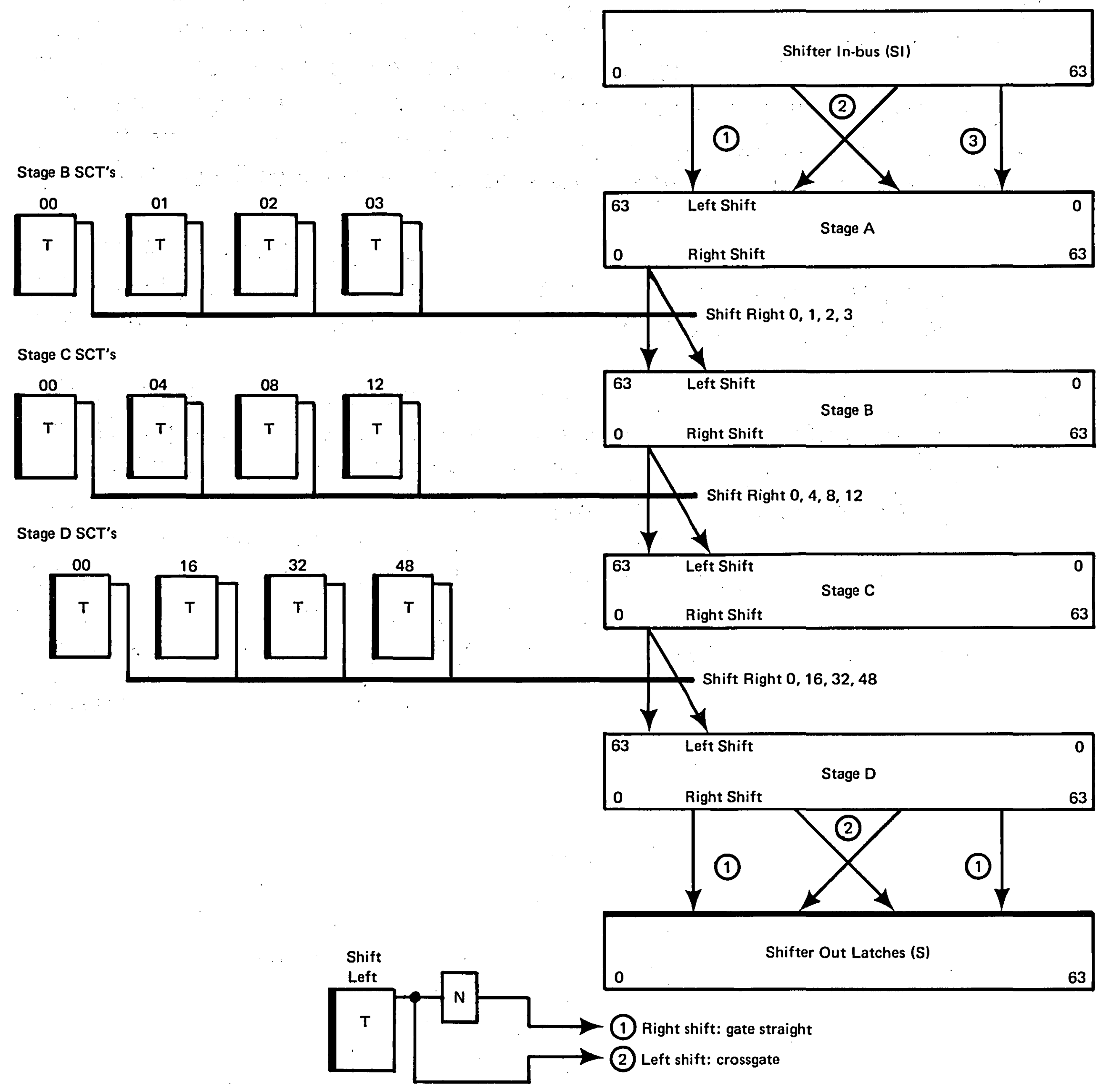
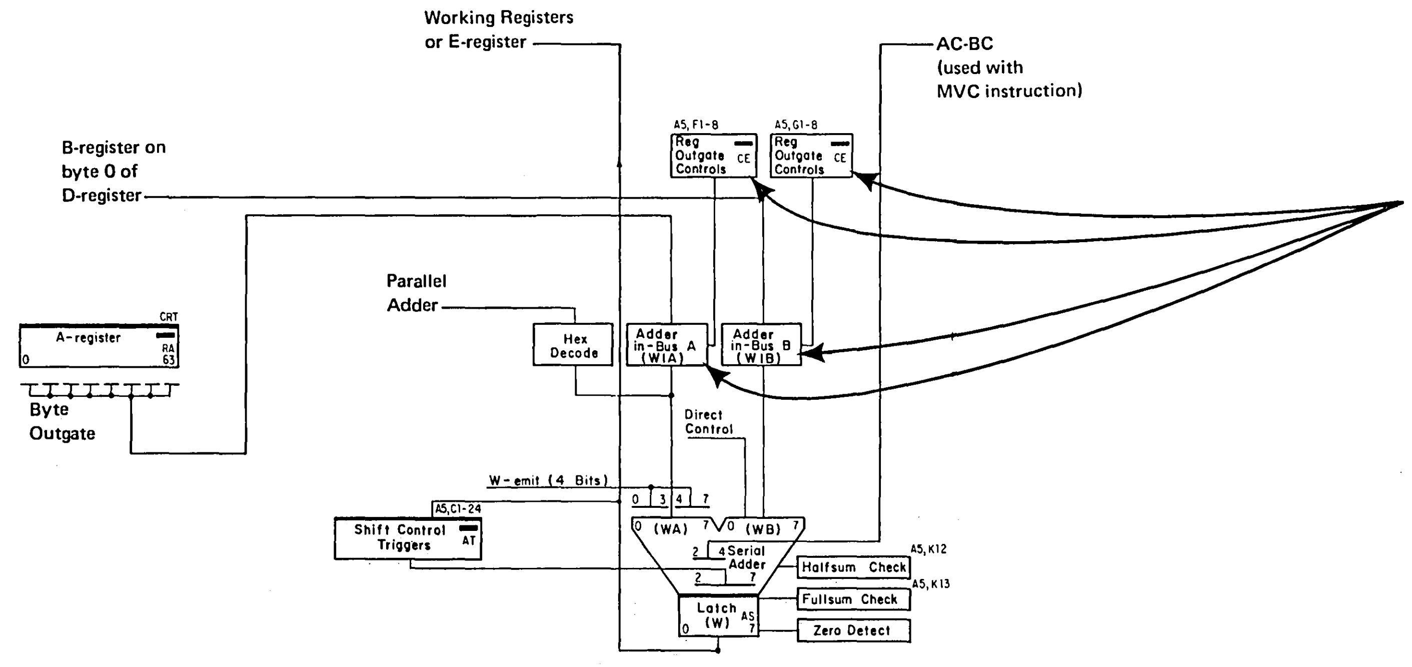


Figure 8-15. Shifter Stages



SERIAL ADDER

- Handles information on a byte basis.
- Processes information from the A-register, the B-register, and byte D0.
- Operation rate is one operation per machine cycle.
- Adder proper must add in every cycle. In serial-adder no-op cycles, zero is added to zero and good parity is forced into the "W parity" latch.

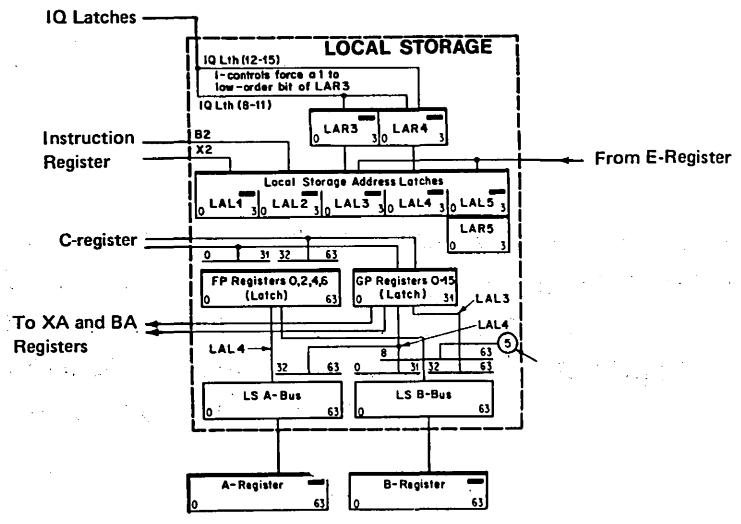
Figure 8-16. Serial Adder

Preliminary Input Buses, WIA and WIB

A particular A-register byte is selected for outgating to WIA by turning on one of eight gate control triggers ('A GCT 0' through 'A GCT 7'). Decoding of ACAL determines which of these GCTs turns on. A particular B-register byte or byte D0 is selected for outgating to WIB by turning on one of nine GCTs ('B GCT 0' through 'B GCT 7' and the GCT for D0). The control program determines whether a B-register byte or byte D0 is selected, and decoding of BCAL determines which of the B-register GCTs turns on.

The primary function of the serial adder is to perform arithmetic operations (binary or decimal) and logical operations (AND, OR, or Exclusive OR) on two one-byte-wide operands. The miscellaneous functions of the serial adder include the following:

1. Check for valid decimal data and sign.
2. Generate preferred decimal signs and zones.
3. Produce constants or increments by use of the four-bit emit field in control storage.
4. Post-normalize the floating-point characteristic. To enable this, the parallel-adder-hex-decode constant is placed on WA for addition to the characteristic that is entering the serial-adder B-side from D0.
5. Act as a straightforward data path. This is done by adding zero to data that is being routed through the serial adder. Further, the data that is being thus routed may be manipulated by cross-gating or by gating only one of the two incoming digits.
6. Decode the pattern characters for the Edit instruction.



LOCAL STORAGE

A high speed local storage (LS) reduces the number of main storage references that are required by the CPU during each operation. Local storage consists of 20 latch registers: 16 general-purpose registers (GPRs) for fixed-point operands and addressing components and four floating-point registers (FPRs) for floating-point operands. The GPRs are fullword registers (32 bits); the FPRs, double-word registers (64 bits). Information can be stored in or fetched from an LS register during one machine cycle (80 nsec), and up to four LS fetch (plus one LS store) operations can occur simultaneously. In other words, LS can be accessed from four sources and written into from one source during a single machine cycle.

The 16 GPRs can be used as base and index registers in address calculations and indexing and as accumulators in fixed-point arithmetic and logical operations. The four FPRs are used for floating-point operands and are used only during execution of floating-point instructions.

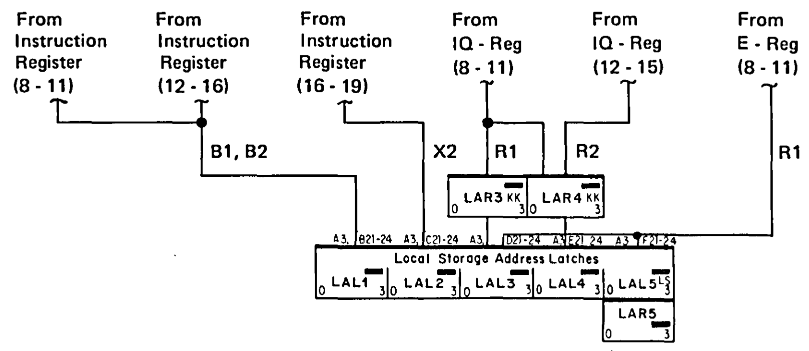
Figure 8-17. Local Storage

Local storage is shared by the I-unit and the E-unit in their efforts to set up and execute instructions, respectively. Therefore, to provide complete I-unit/E-unit overlap and the ability to process an instruction in one cycle, four GPRs can be read and a fifth can be written into the same cycle. This multiple-accessing ability of LS allows the operand put-away initiated by one instruction to be overlapped with the execution of the next instruction. In addition, execution of an instruction may require that the contents of one or two LS registers be fetched at the same time that the I-unit is calculating an effective address, which may also require the use of one or two registers of LS.

The I-unit accesses LS for two operations: E-unit setup and address calculations. When an instruction is passed from the I-unit to the E-unit, the E-unit registers must be setup with the values needed to begin instruction execution. Setup includes fetching operands from local storage and ingating them into the A- or B-register as required. To perform address calculations, the I-unit contains a three-input address adder. Two of the three principal inputs to address-adder input registers are two local storage buses, LS-X-data bus and LS-B-data bus. During the instruction decode cycle, the contents of the required GPRs are fetched from LS, and the other addressing components to be added are set up. The addressing components are ingated to the adder input registers (XA, BA, and DA), and the add occurs, producing a main storage address which is sent to one of the I-unit address registers.

LS fetches by the E-unit are always for operands. As directed by the control program in the E-unit, the contents of the specified GRP or FPR are gated to the A- and/or B-register, as required. Local storage stores are always performed by the E-unit. The control program gates the contents of the C-register to the specified GPR or FPR.

Figure 8-18. Local Storage



LOCAL STORAGE ADDRESS LATCHES

Read addressing is performed by four sets of local storage address latches: LAL1, LAL2, LAL3, and LAL4; write addressing is performed by local storage address register 5 (LAR5).

LAL1 addresses LS to fetch the index component of an effective address. The index component is a 24-bit number contained in a GPR that is specified by the X-field of an RX format instruction. When LAL1 is addressed from the I-unit instruction register, data from the specified GPR is gated to the I-unit XA register.

LAL2 addresses LS to fetch the base-address component of an effective address. The base address is a 24-bit number contained in a GPR that is specified by the B-field of an instruction that references main storage. When LAL2 is addressed by the I-unit instruction register, data from the specified GPR is gated to the I-unit BA register.

LAL3 addresses LS for GPR and FPR fetches to the B-register. If the I-unit initiates the operation, the address is received from the IQ register via LAR3. If the E-unit initiates the operation, the address is received from the E-register.

LAL4 addresses LS for GPR and FPR fetches to the A-register and to bytes 0-3 of the B-register. If the I-unit initiates the operation, the address is received from the IQ register via LAR4. If the E-unit initiates the operation, the address is also received from the IQ register via LAR4. (LAR4 can be addressed from the E-unit only on the first cycle of an instruction execution.)

LAR5 addresses LS for GPR and FPR stores. The contents of E (8-11) are always used to address LS on a write operation via LAL5. LS stores are controlled by the E-unit.

Figure 8-19. LS Address Latches

CALENDAR CLOCK

- Programmer can read precision calendar time.
- 52 latches provide 135-year cycle.
- One-microsecond increments are independent of other system activities.
- Parity check assures clock validity.
- Security switch reduces chance of erroneous update.

The calendar clock provides a high-precision calendar time to the programmer. The clock is set by the system programmer and a switch on the console helps assure its security. A programmer can read (store) the clock at any time.

The calendar clock is a 52-bit counter increased by one every microsecond. Each clock advancement is independent of other CPU and channel activities. The clock content ranges from one microsecond to 135 years. Clock overflow is recorded but no interrupt condition is provided.

Bits 0 through 51 of a doubleword latch arrangement provide the binary storage needed to count to 135 years by microseconds. The doubleword latch is advanced each microsecond by adding 1 to bit 51. Bits 52 through 63 retain the status originally set, while bits 0 through 51 hold the binary representation of the current time of day.

The programmer reads the clock by means of the store clock instruction. Store clock reads the current calendar time and puts it in main storage. Reading and storing the clock does not affect the content of the latches or the incremental activity.

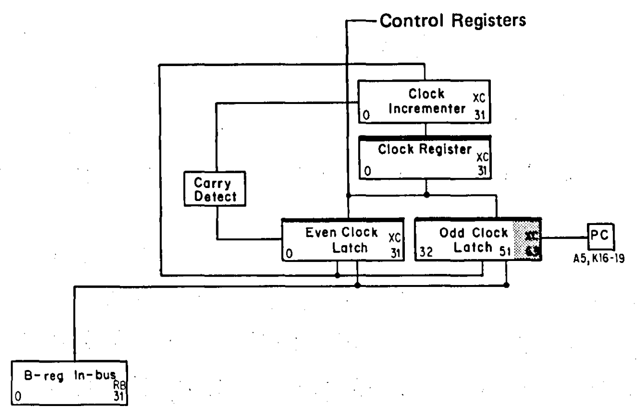


Figure 8-20. Calendar Clock

CPU IDENTITY REGISTER

- Register is composed of jumper wires.
- Programmer requests readout.

The CPU identity register provides processor identification to the system programmer. The programmer requests CPU identification and the control storage microprogram routines transfer the data to main storage. The CPU identity register is composed of jumper wires. The bits, when tied up, represent the desired data. Any arrangement may be set into the register for CPU identification; however, the data set into the register can not be readily changed.

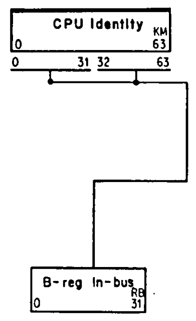


Figure 8-21. CPU Identity Register

CONTROL REGISTERS

Extended system control requires four of sixteen control registers. The control registers, accessable to the system programmer, are addressed 0, 2, 14, and 15. The four control registers provide extended control of external masks, channel masks, error recovery and logout.

Control registers are not addressable as part of main storage. The programmer must store the register data in main storage when required; but the store operation does not affect the content of the registers. The programmer must load data from main storage to change the register content. Store and load instructions provide the only control-register access method available to the programmer.

Store and load operations employ control storage microprogram concepts. When control registers read out (microprogram control) the data transfers to working register B for further manipulation.

Control address latches and registers select the desired control registers and gate the control register word (32 data bits and four parity bits) onto the control register bus. When loading, data enters the control register latches via the shifter in-bus.

The program status word (PSW) mode gates the control registers. Extended control mode (PSW bits 12 is one) allows system control through control register mask bits, while basic control mode (PSW bit 12 is zero) negates the content of the control registers. System programmer access to the control registers requires the following RS instructions: load control, LCL; store control, STCL.

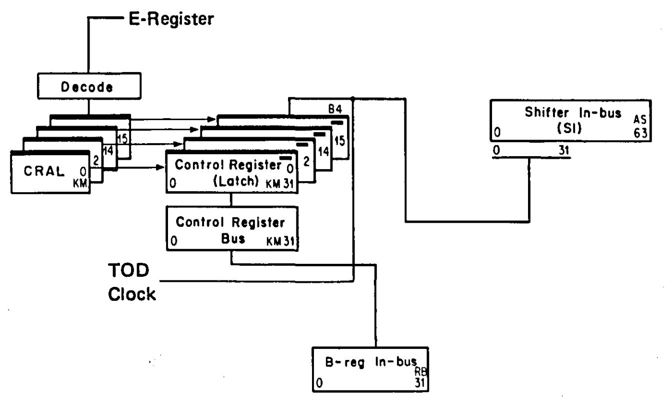


Figure 8-22. Control Registers

PROGRAM STATUS WORD LATCH

The PSW latch has 40 latches to hold the current PSW data. When a new PSW is set, bits 0 through 15 and 36 through 39 are set into the corresponding PSW latches. The PSW control data is then available for system control during the entire clock cycle for every clock cycle until the next PSW is set. When an interrupt occurs, the remaining PSW latch bits are set to reflect the interruption code, the instruction length code (ILC), and the condition code (CC).

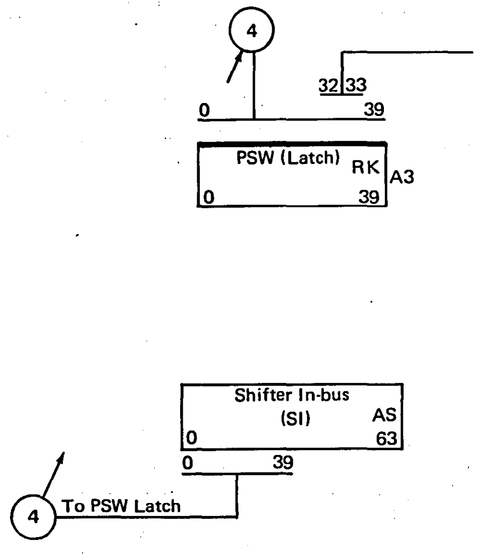


Figure 8-23. PSW Latch

E-REGISTER AND ADDER

The E-register adder increments or decrements bits 8-15 of the instruction in the E-register during execution of VFL, load multiple, and store multiple instructions. This adder can increment or decrement bits 8-15, which correspond to the R1/R2, R1/R3, L1/L2, or L fields of the instruction (depending upon the format), as two separate four-bit values (8-11 and 12-15) or as one value under microorder control. Bits 8-11 can be incremented or decremented by 1, 2, or 4; bits 12-15 can be decremented by 1 only; and, when considered as a single field, bits 8-15 can be decremented by 1 or 8. In addition, bits 8-15 can be decremented by the contents of ACAL or BCAL. Note that data that is gated into the E-register adder is always bits 8-15 of the instruction, whether the input is from the E-register or the instruction queue register.

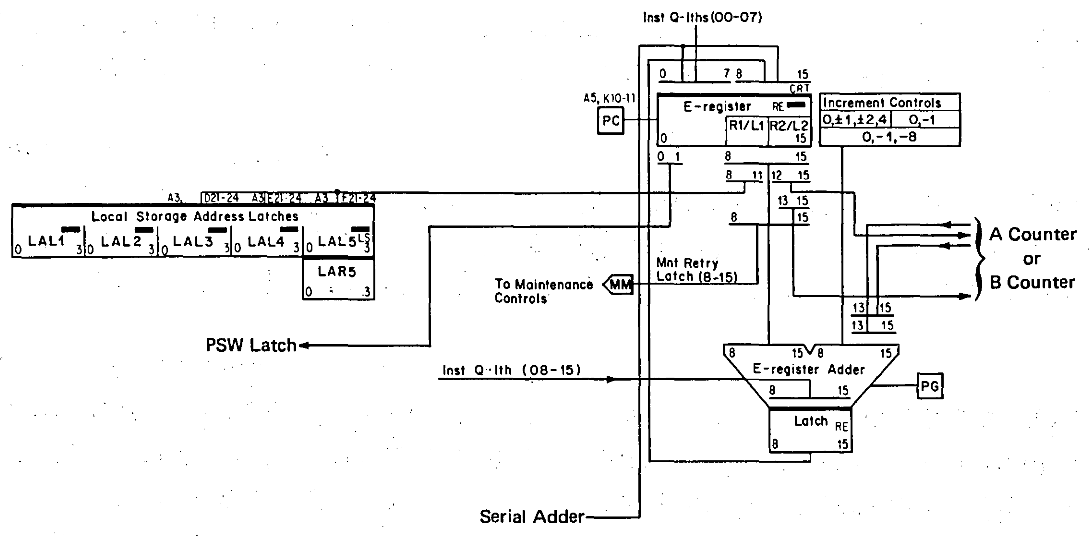
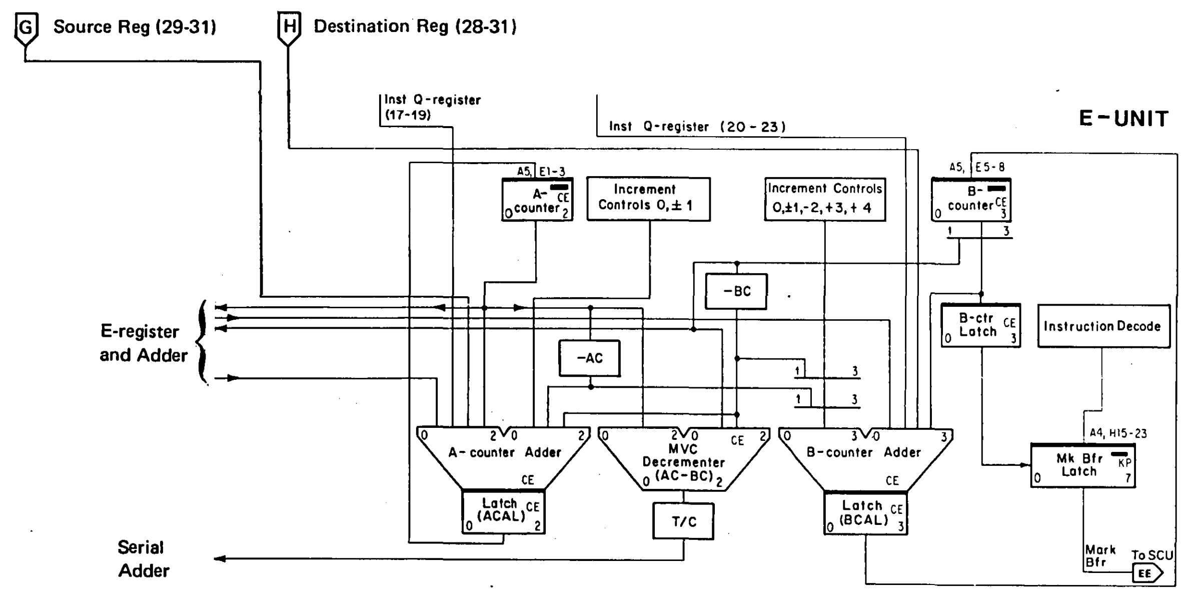


Figure 8-24. E-Register And Adder



A-COUNTER

The AC consists of a three-bit register, a binary adder (A-counter adder), and adder latches (ACAL). These components perform operations on values of 000 through 111, that can be incremented, decremented, complemented, or have other values added to them via the A-counter adder. In operation, these binary values (000 through 111) specify A-register bytes 0-7 for gating to the serial adder. The AC also serves as a general-purpose counter to hold the low-order three bits of source address values for byte locations during execution of VFL instructions and for specification tests during execution of RX format instructions.

Initial values are gated to the A-counter adder under control storage or I-unit control from E (13-15), Src (29-31), or IQ (17-19). In addition, initial values of 000, 001, or 111 can be set into the A-counter adder by means of the increment controls.

For control of the A-register bytes that are gated to the serial adder, the contents of ACAL are gated to a decoder which sets an A-register gate control trigger per the decoded ACAL value. This occurs every cycle; thus, when a control storage microorder is issued that gates an A-register byte to the serial adder, the gated byte is dependent upon which gate control trigger is set and, hence, on the value in ACAL.

Figure 8-25. AC And BC

B-COUNTER

The BC consists of two four-bit trigger registers (BC and BCAL triggers), a binary adder (B-counter adder), and two latch registers (BCAL and BCL). These components gate a byte of data from the B-register to the serial adder and ingate a byte of data from the serial adder to the B-register. The BC also serves as a general-purpose counter to hold the four low-order bits of the destination address. This latter usage enables the BC to be used to set marks for storage operations.

Initial values are gated to the B-counter adder under control storage or I-unit control from E (12-15), Dst (28-31), or IQ (20-23). In addition, initial values of 0, 1, 3, 4, E, and F (hex) can be set into the B-counter adder by means of the increment controls.

For control of the B-register bytes that are gated to the serial adder, the contents of BCL (01-03) are gated to a decoder which sets a B-register gate control trigger per the decoded BCL value. This occurs every cycle; thus, when a microorder is issued that gates a B-register byte to the serial adder, the gated byte is dependent upon which gate control trigger is set and, hence, on the value in BCL.

B-register ingating from the serial adder is under control of BCL (B-counter latch). The contents of BCL are decoded to select the specified byte of the B-register to which data from W is gated.

AC-BC DECREMENTER

The AC-BC decrementer is a three-position binary adder that sets up proper shift amounts in W during execution of the MVC instruction.

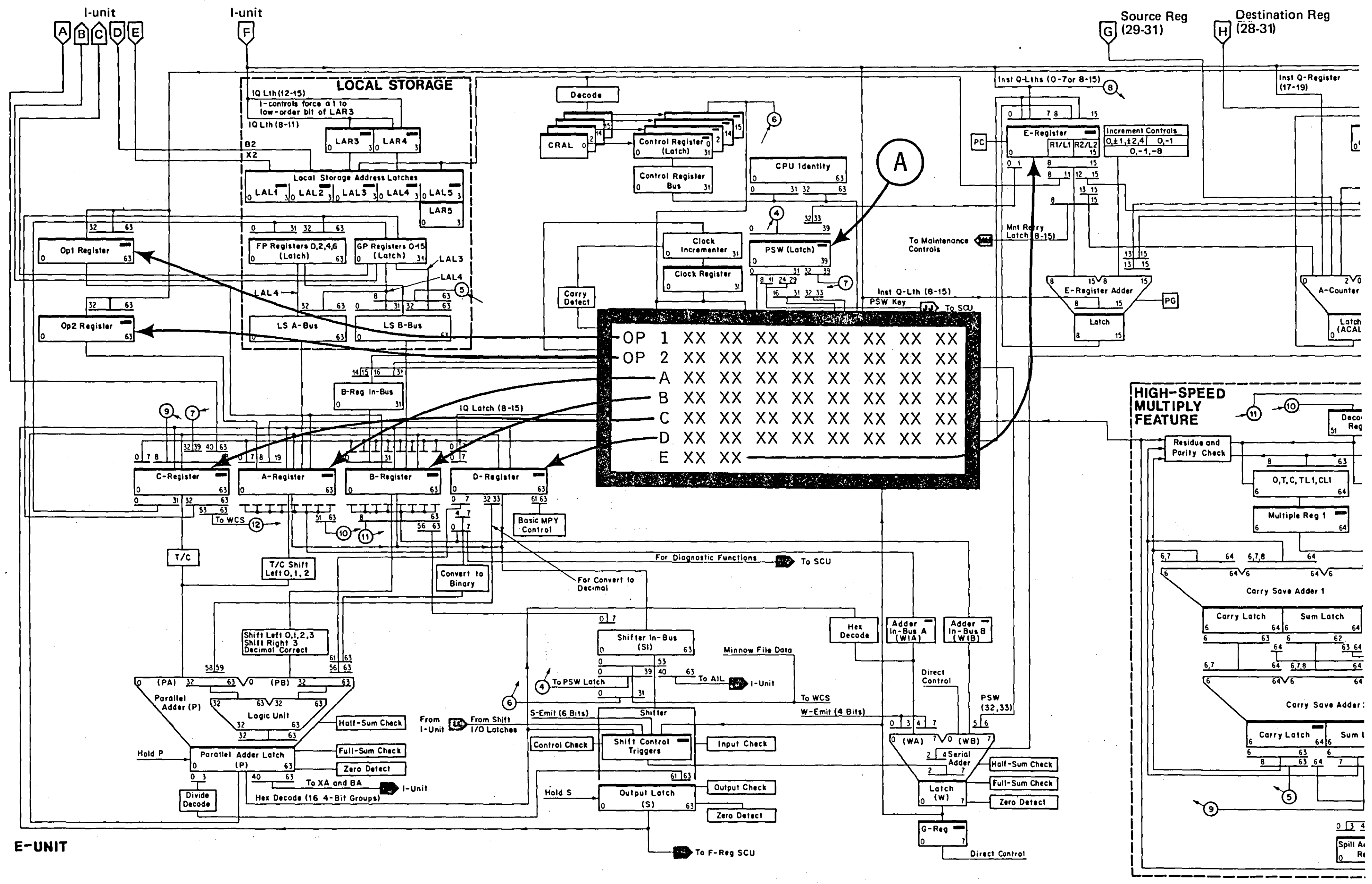


Figure 8-26. OP And Working Registers

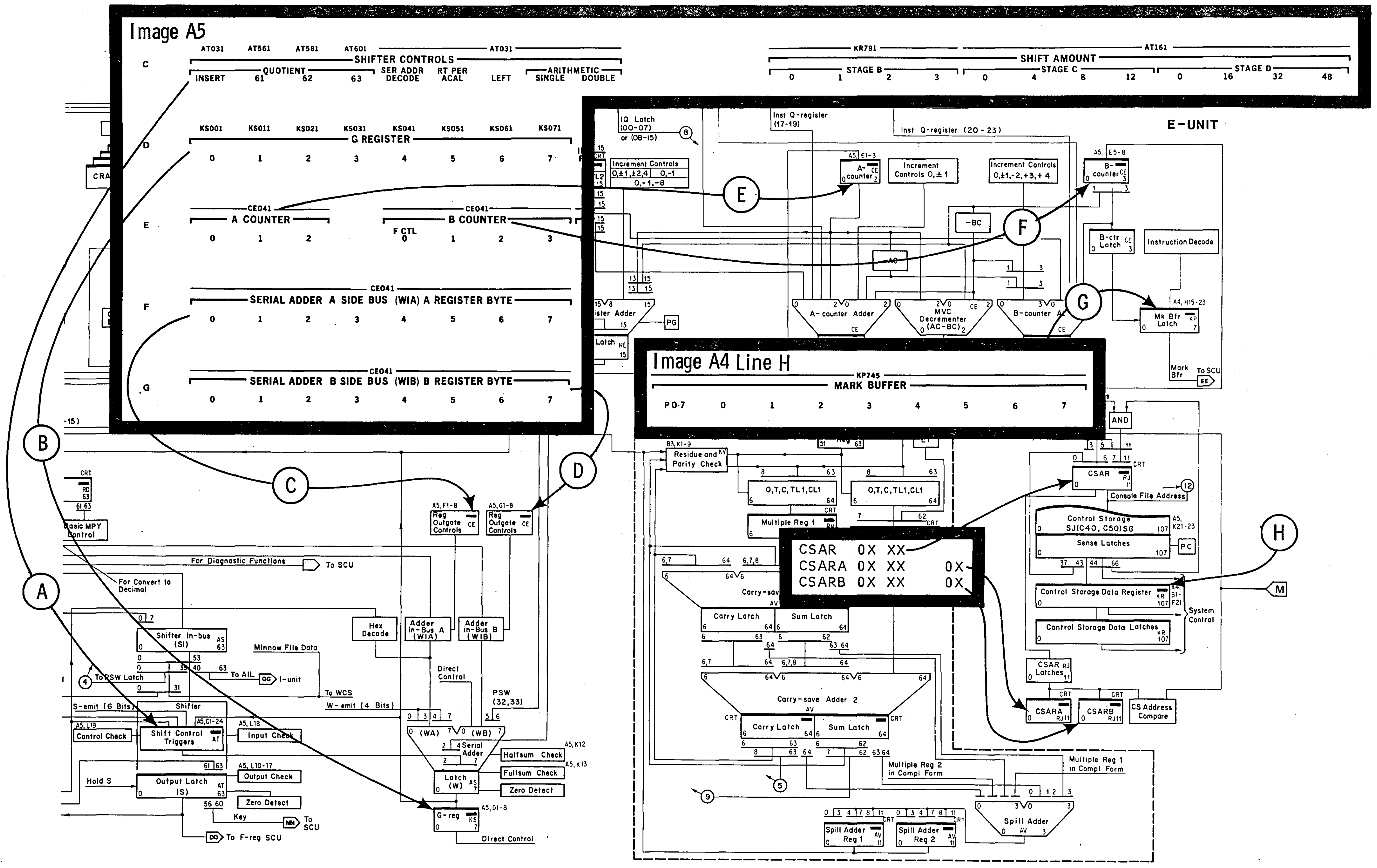


Figure 8-27. Misc Display

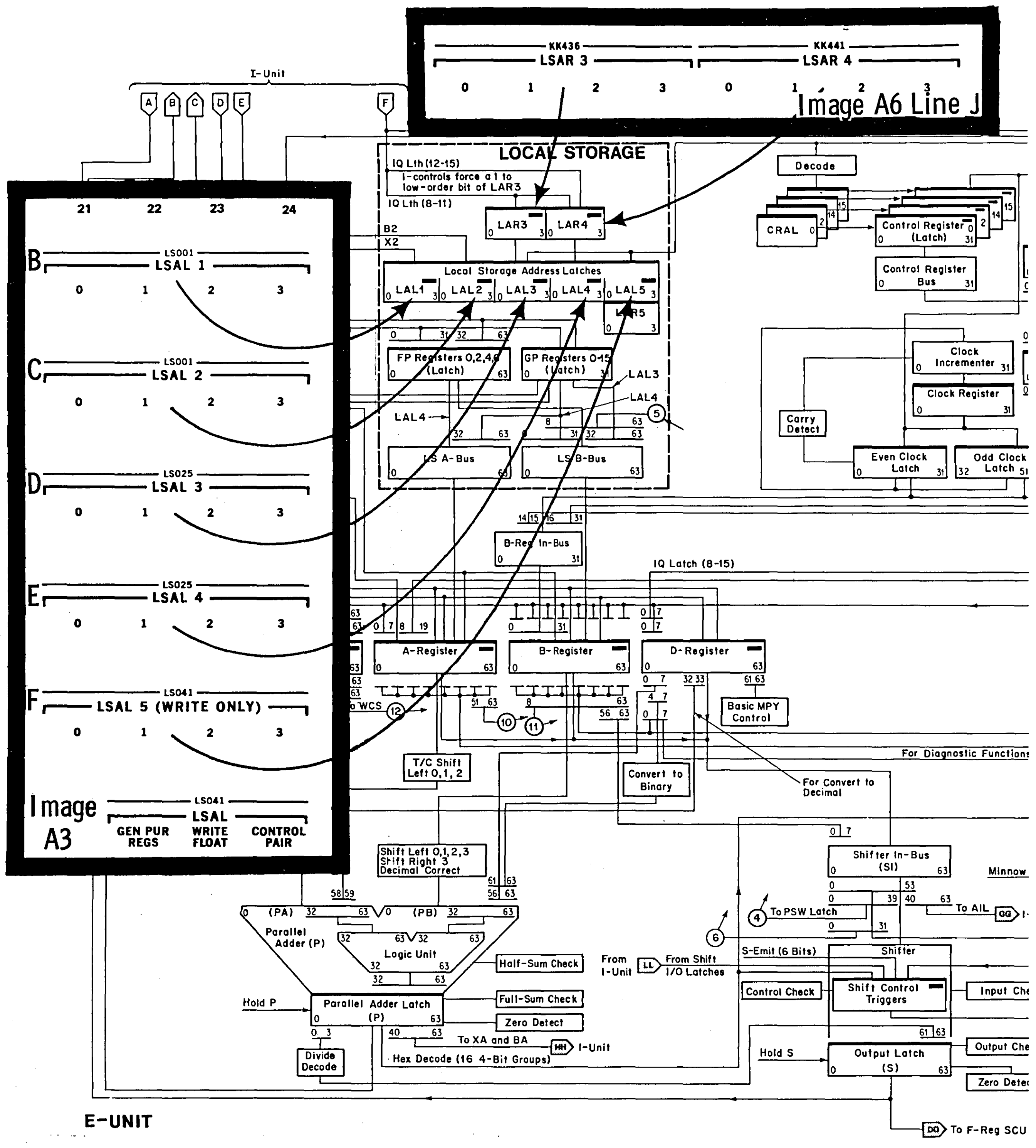


Figure 8-28. LS Display

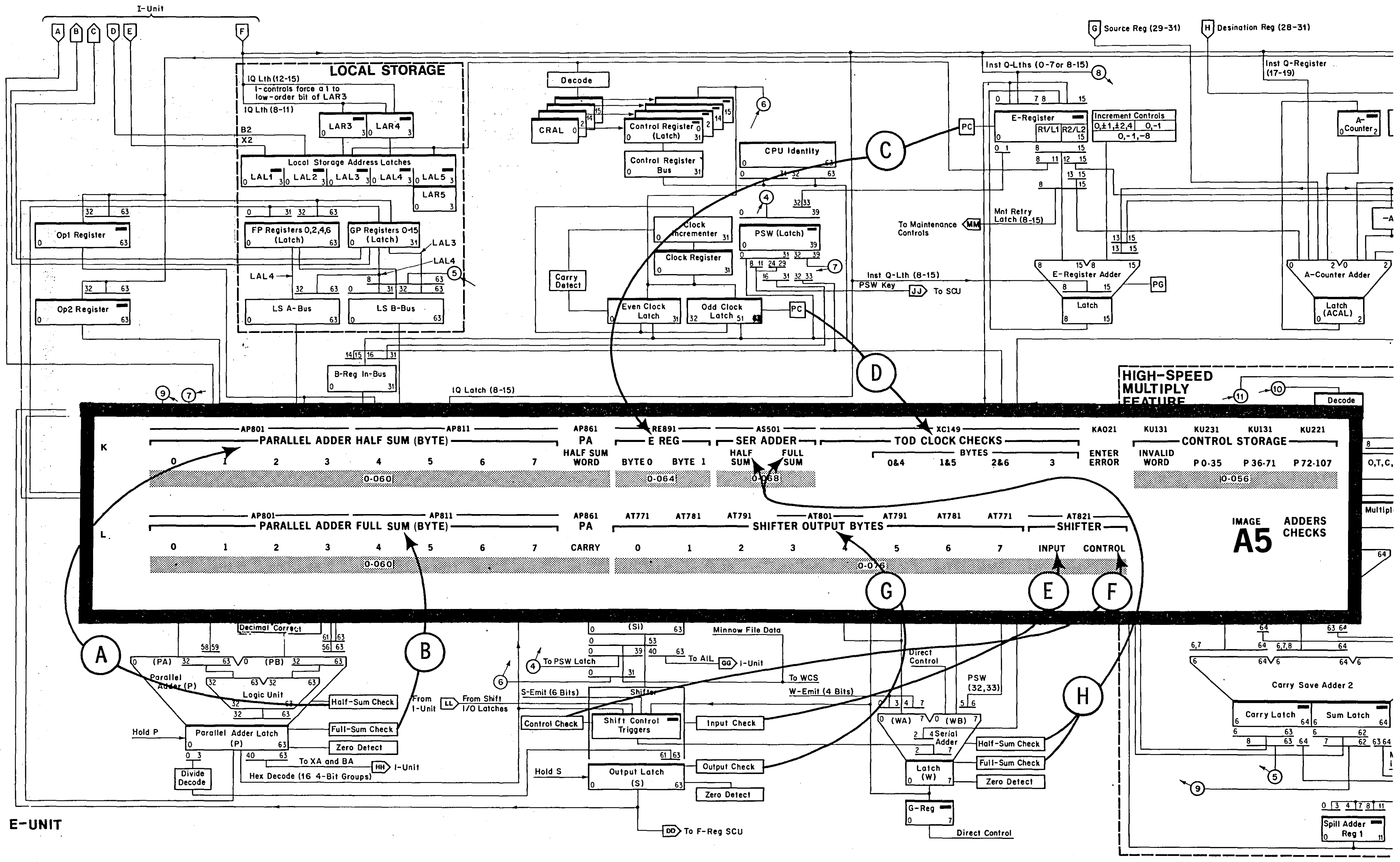
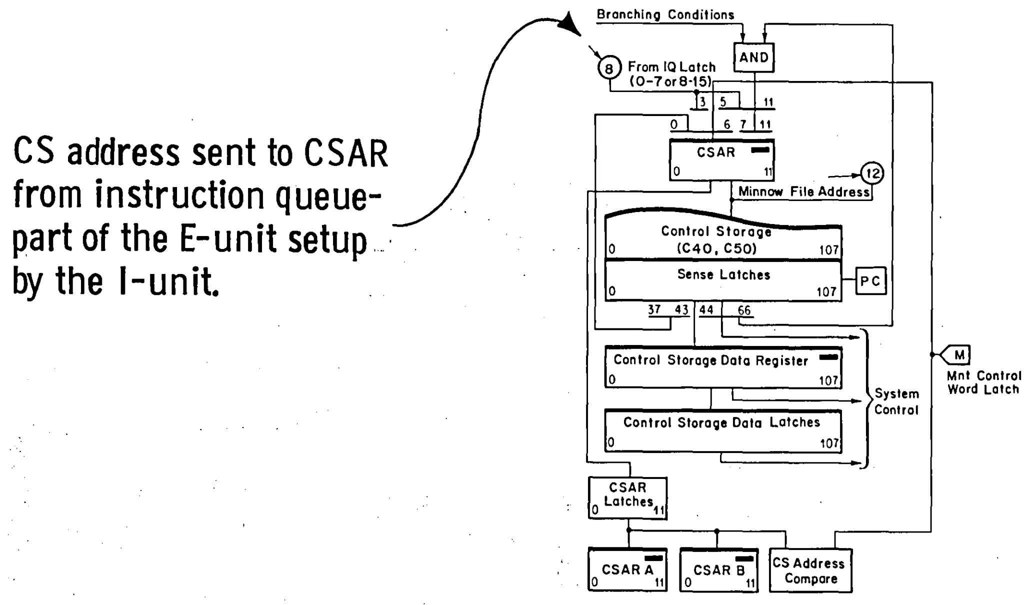


Figure 8-29. E-Unit Errors



CONTROL STORAGE (CS)

The CS logic units are shown above. Control storage holds the program (called the "microprogram") that provides the control of the E-unit as it executes instructions.

To understand the operations of control storage, it is helpful to note its relationship to conventional controls. Conventional controls are characterized by state/control triggers that activate control lines in accordance with the operation to be performed and existing machine conditions. Each cycle that the CPU may take represents a state of the CPU as defined by the control circuitry. Each state, in turn, specifies the control lines that are to be activated during that cycle and the state that is to follow next. The specified state causes the next state/control trigger to be set in the following cycle. In some cases, the next state may be contingent upon a branch condition which selects one of two or more state/control triggers.

Figure 8-30. Control Storage

In the E-unit, the state/control triggers are replaced by CS words that are sometimes called micro-instructions. Each CS word consists of a predetermined bit pattern that represents a state of the CPU and controls a portion of the CPU (the E-Unit) for one machine cycle. When decoded, the CS word defines all control lines that are to be activated during that machine cycle. Also contained in the CS word is the information required to address the next CS word.

Control storage for the E-unit consists of 2,048 108-bit words of read-only storage (C50) and 512 108-bit words of writeable control storage (C40). Each control storage word contains a unique bit pattern. When decoded, the bits control gates to route data through the E-unit.

For the C50, information can be read out as required, but a physical modification is necessary to change the stored information. For the C40 writeable control storage, information may be changed by using the load microprogram instruction.

In general, a control word is read out of the control storage (either the C40 or the C50) at the end of each machine cycle and controls the E-unit during the following machine cycle. Each control storage (CS) word contains the address of the CS word to control the E-unit during the following cycle. The number of control words (and machine cycles) required to perform a particular operation may vary because the individual functions and the address of the next CS word are modifiable by the operation in progress, the data or control bit configuration, and the detection of interrupts or exceptional conditions.

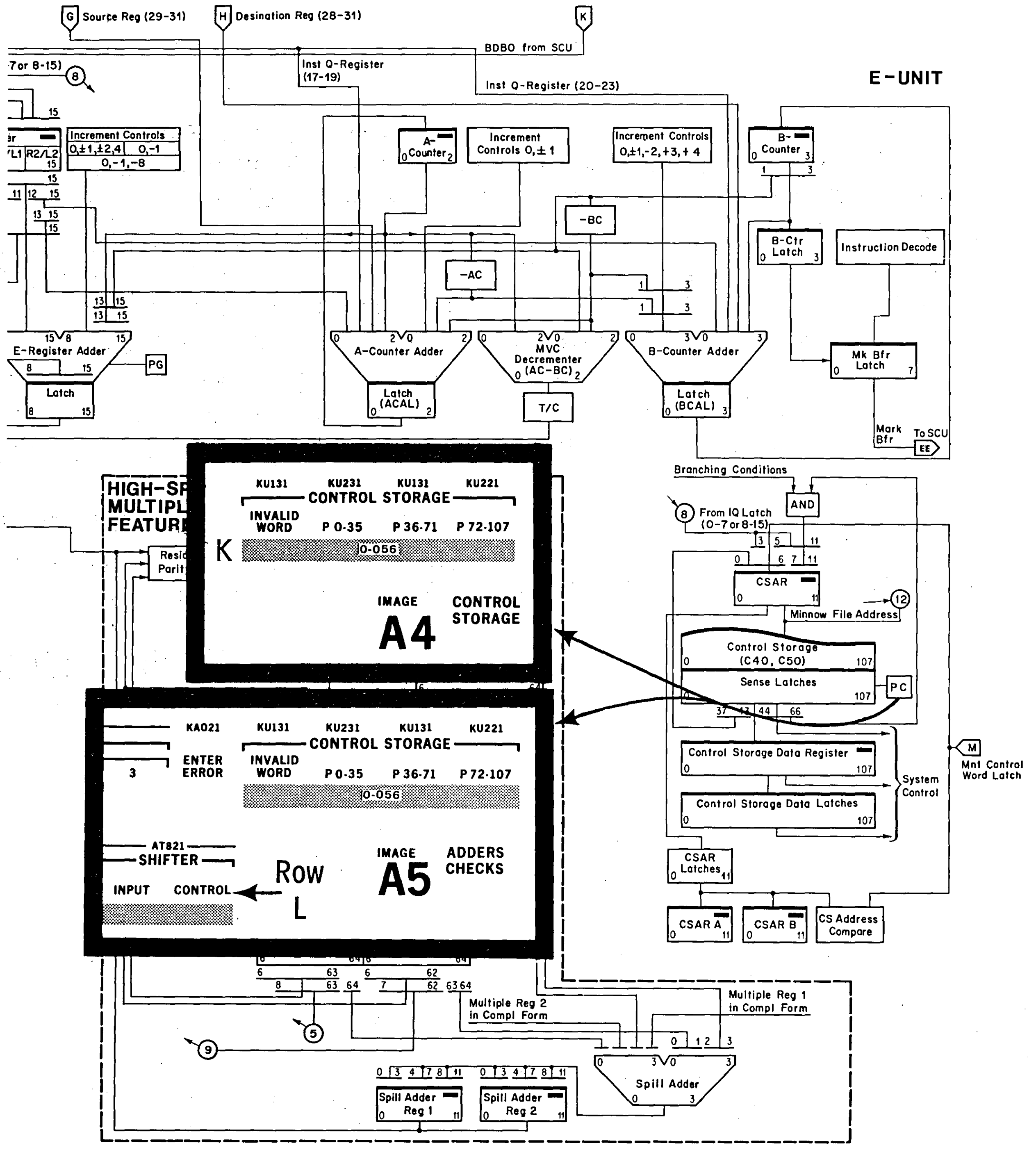


Figure 8-31. CS Error

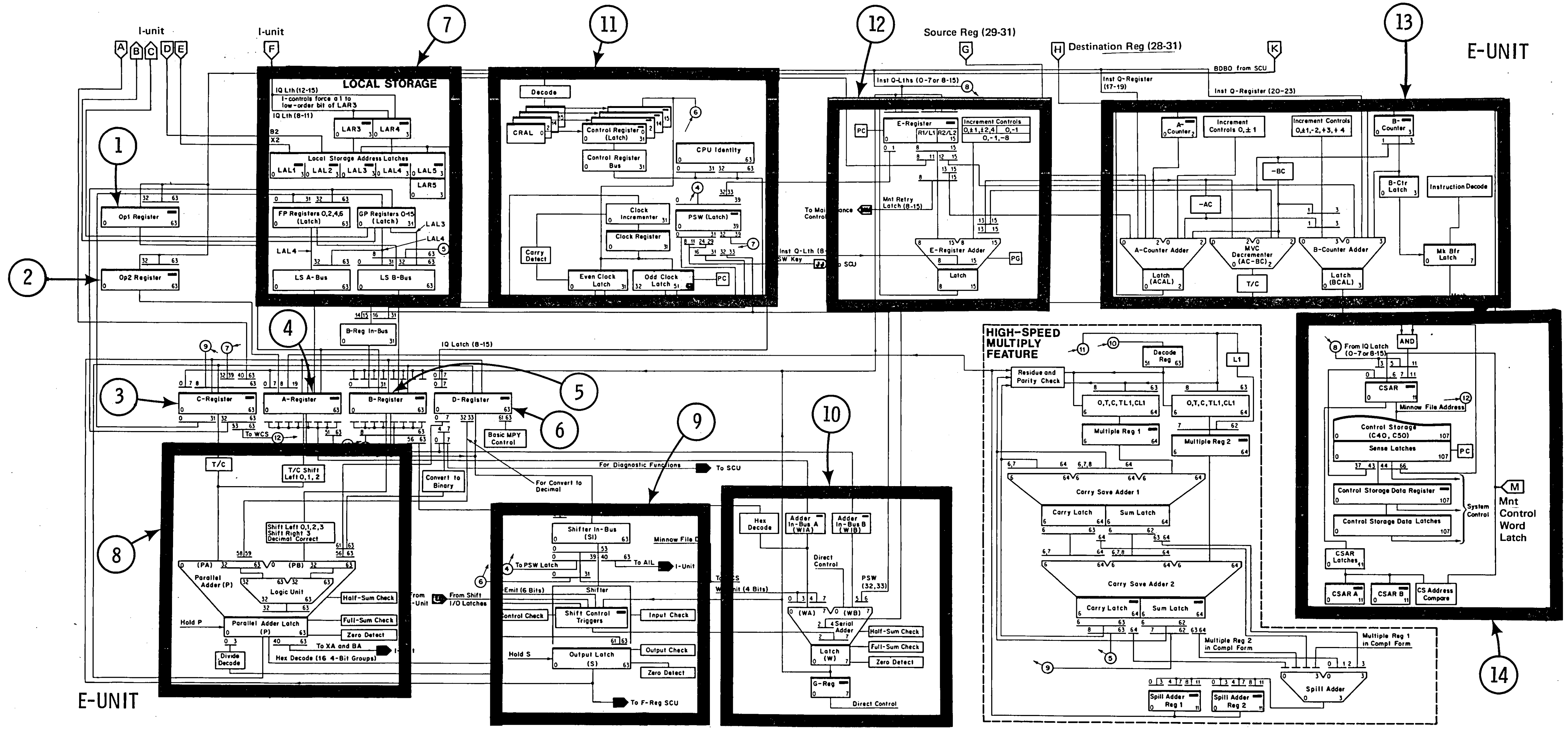


Figure 9-1. Session 9, Questions 6-10

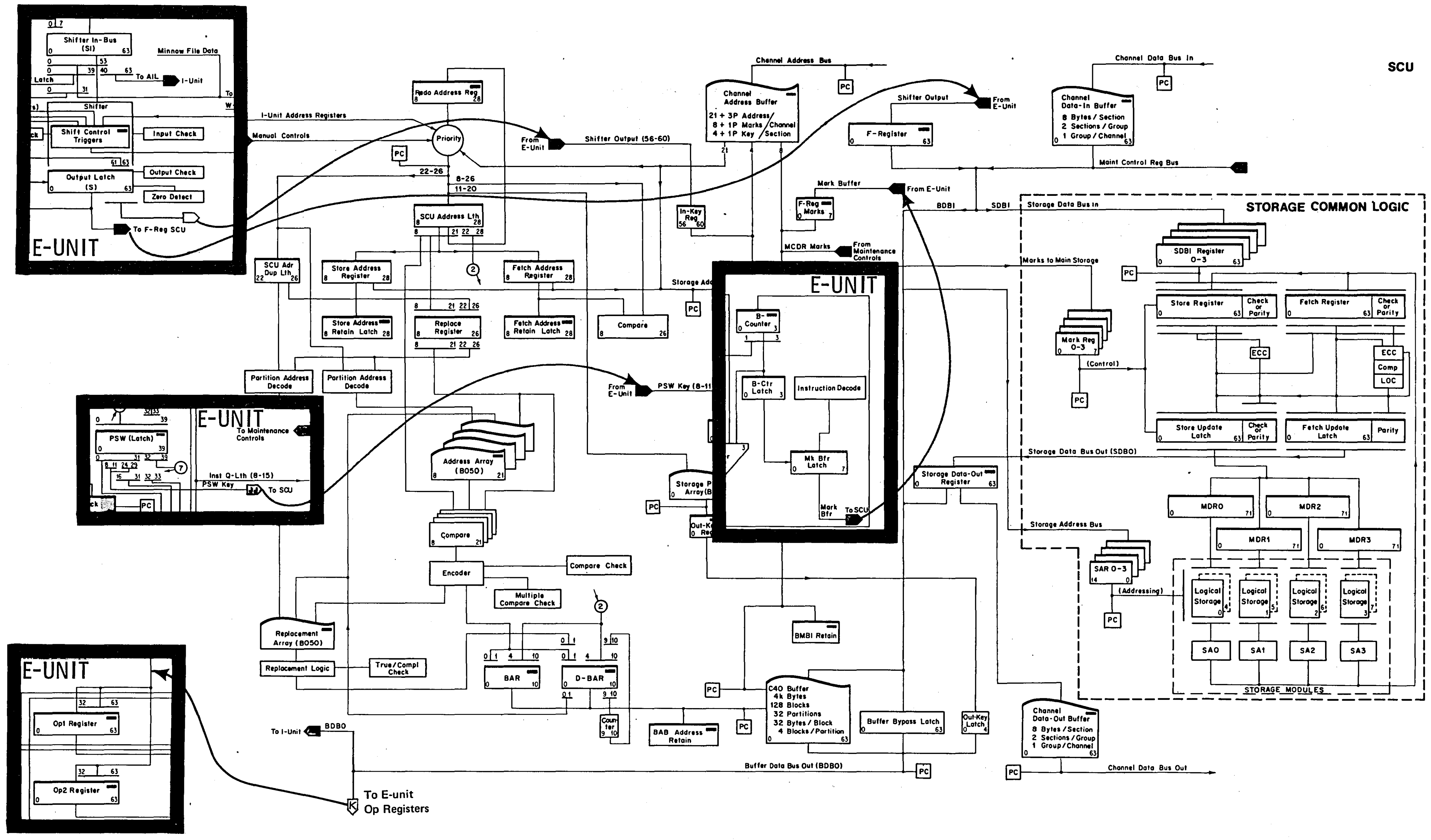


Figure 9-2. E-SCU Data Paths

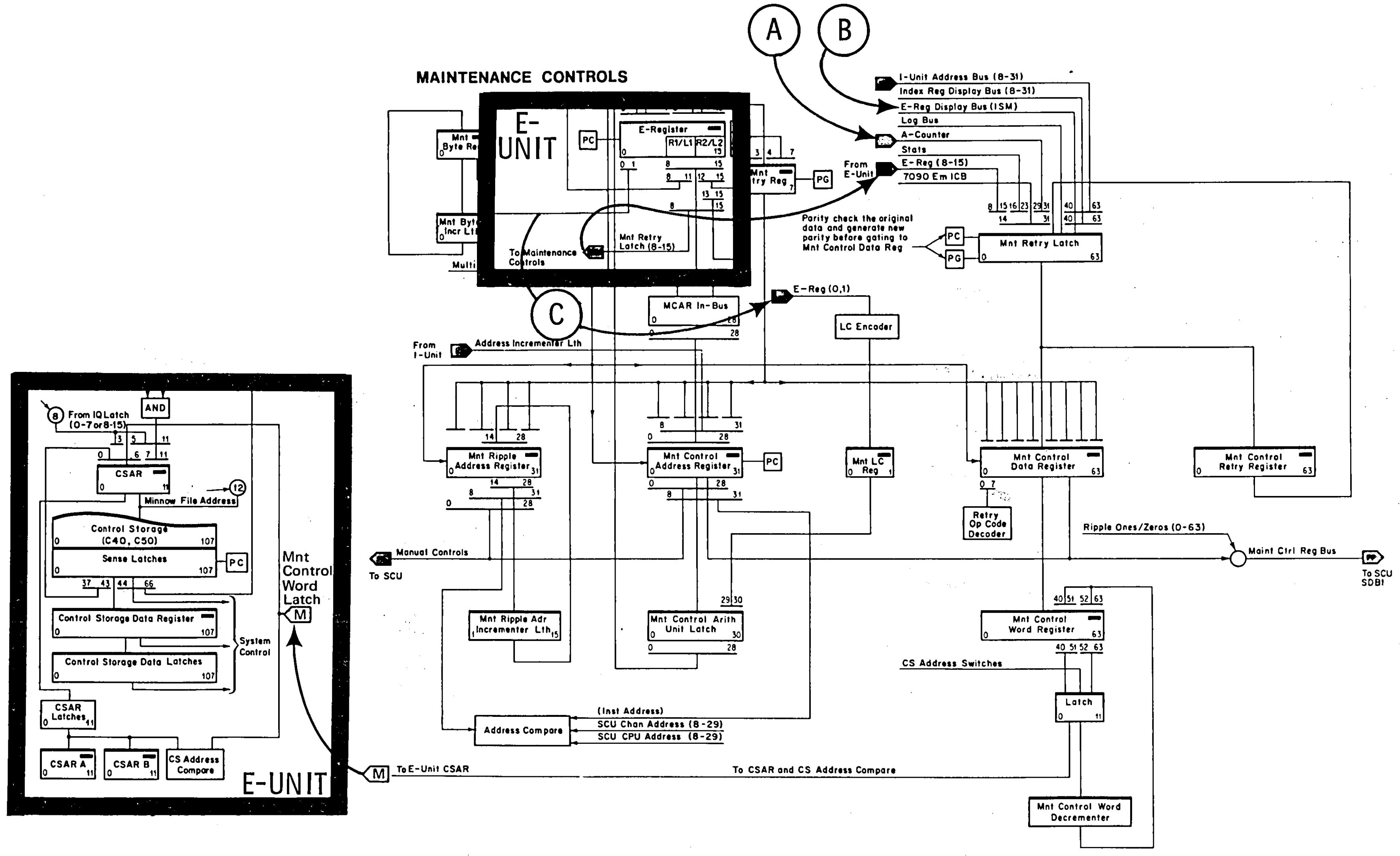


Figure 9-3. E-MC Data Paths

BASIC DATA FLOW

The basic data flow through the E-unit consists of gating a register (or two registers) through the parallel adder, the serial adder, or the shifter, to another register in a single E-unit cycle.

From the main storage or the high-speed buffer, data enters either the operand 1 (Op 1) or the operand 2 (Op 2) register, where it is held until the E-unit is ready to process it. When the E-unit is ready, data enters a working register (register A, B, C, or D), and the E-unit cycle begins by gating data from the working register(s) into the adder or shifter being used. After passing through the adder or shifter, the manipulated data is latched up in the adder or shifter output latch to wait for the next clock pulse, which gates it back either to a working register for additional processing on a subsequent cycle or to the F register in the SCU, where the data is placed back in main storage and the high-speed buffer.

Figure 9-4. E-Unit Basic Data Flow

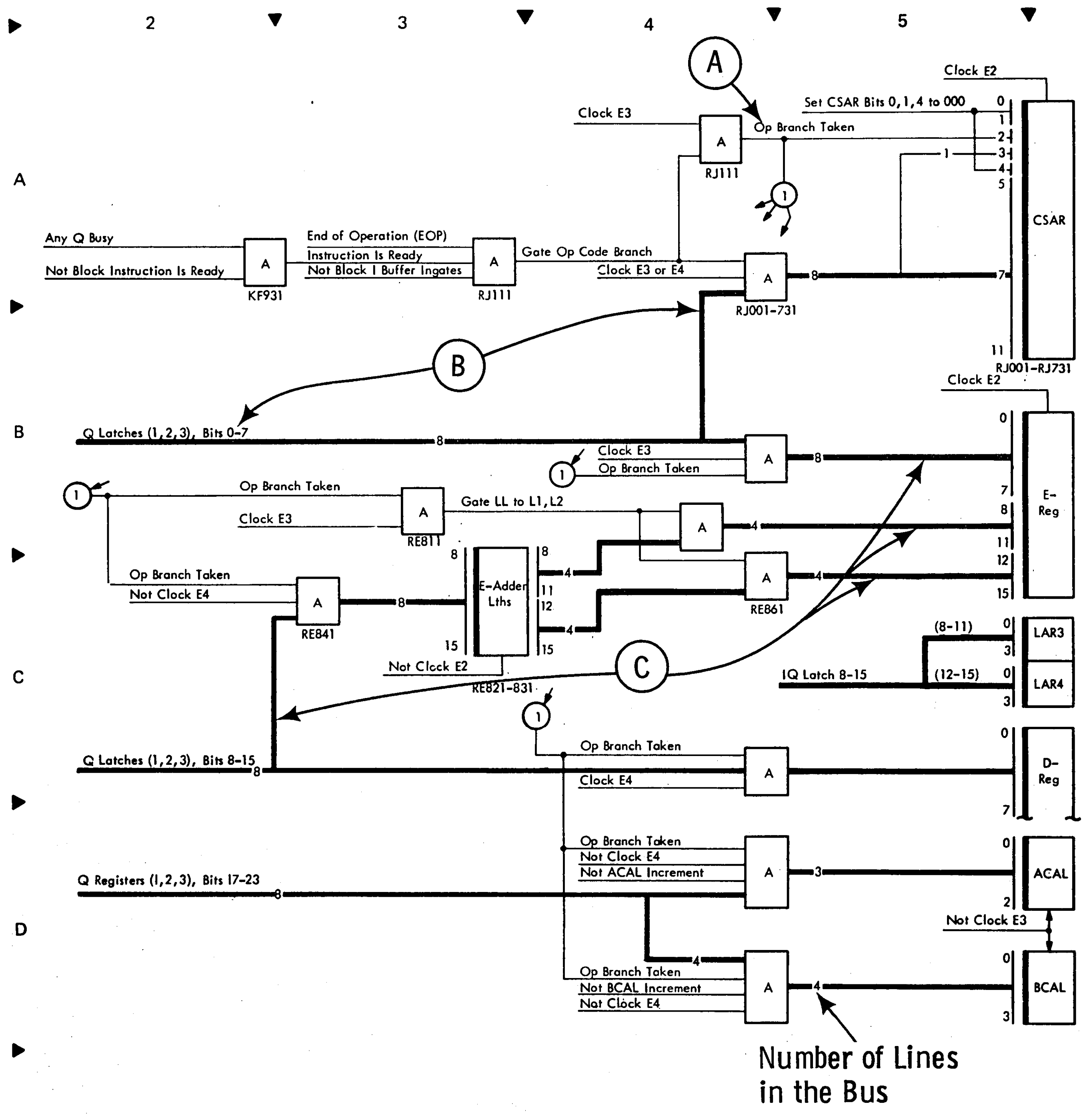


Figure 9-5. I-Unit Setup Of E-Unit

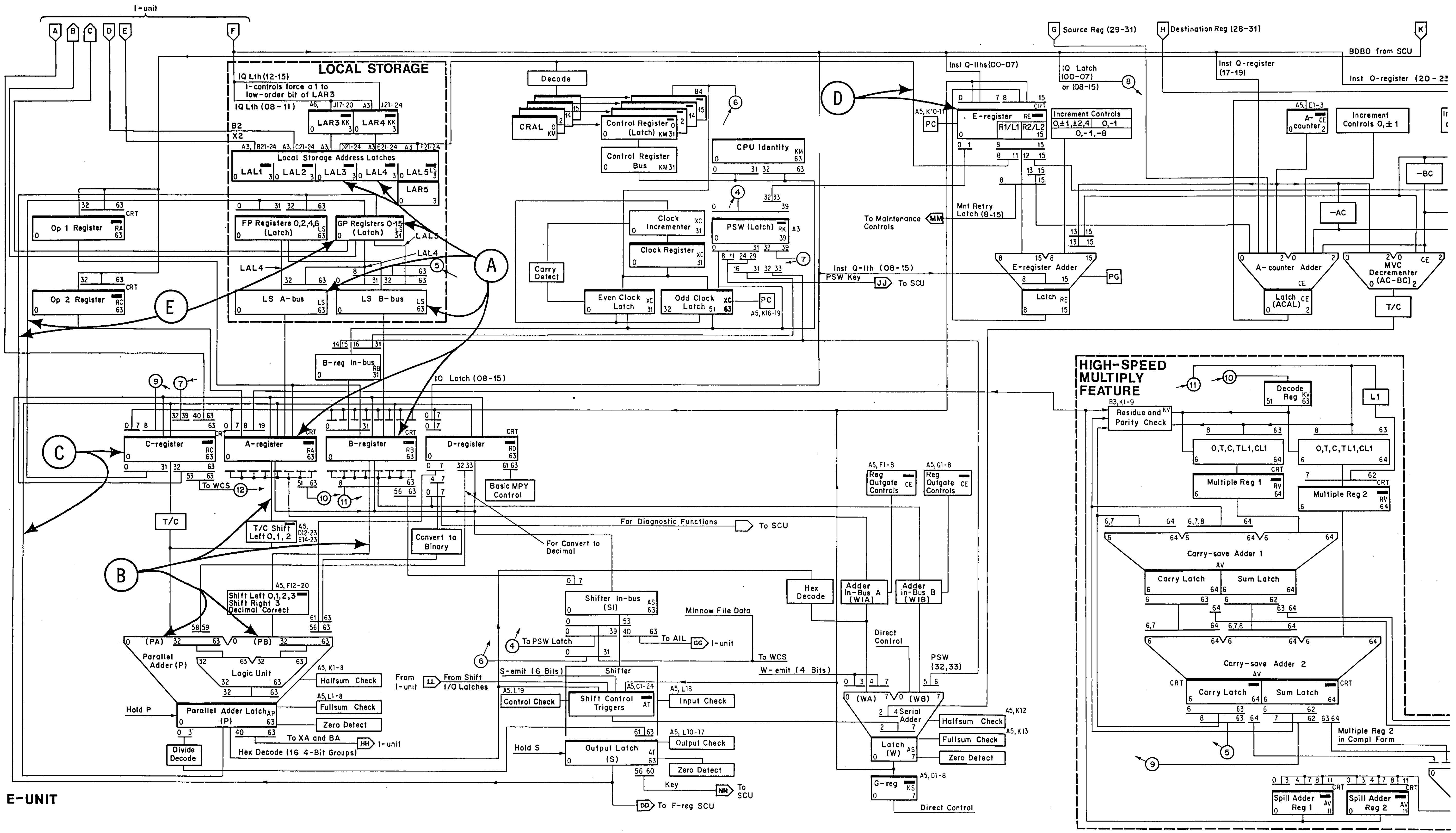


Figure 9-6. RR Add

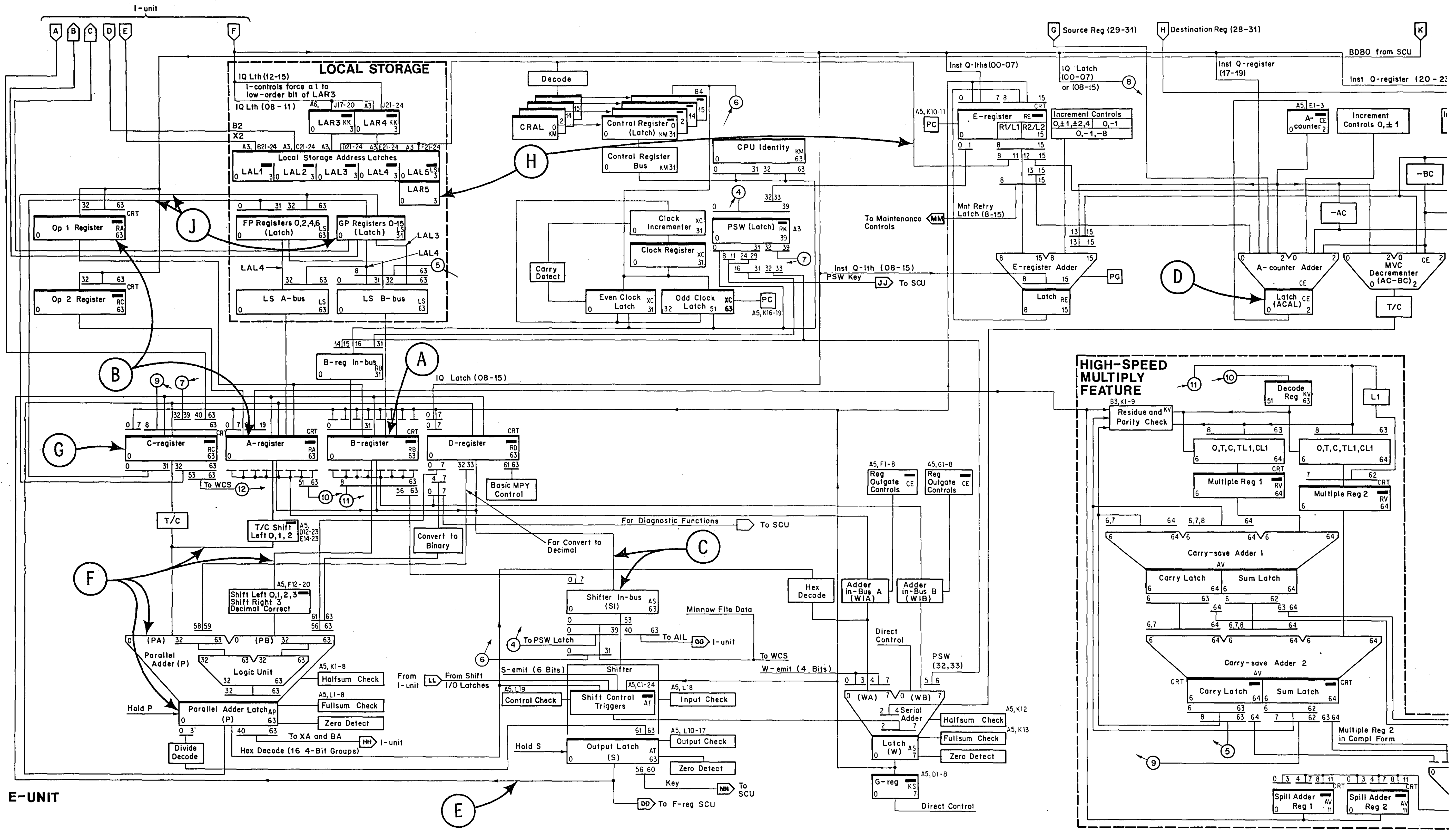


Figure 9-7. RX Add

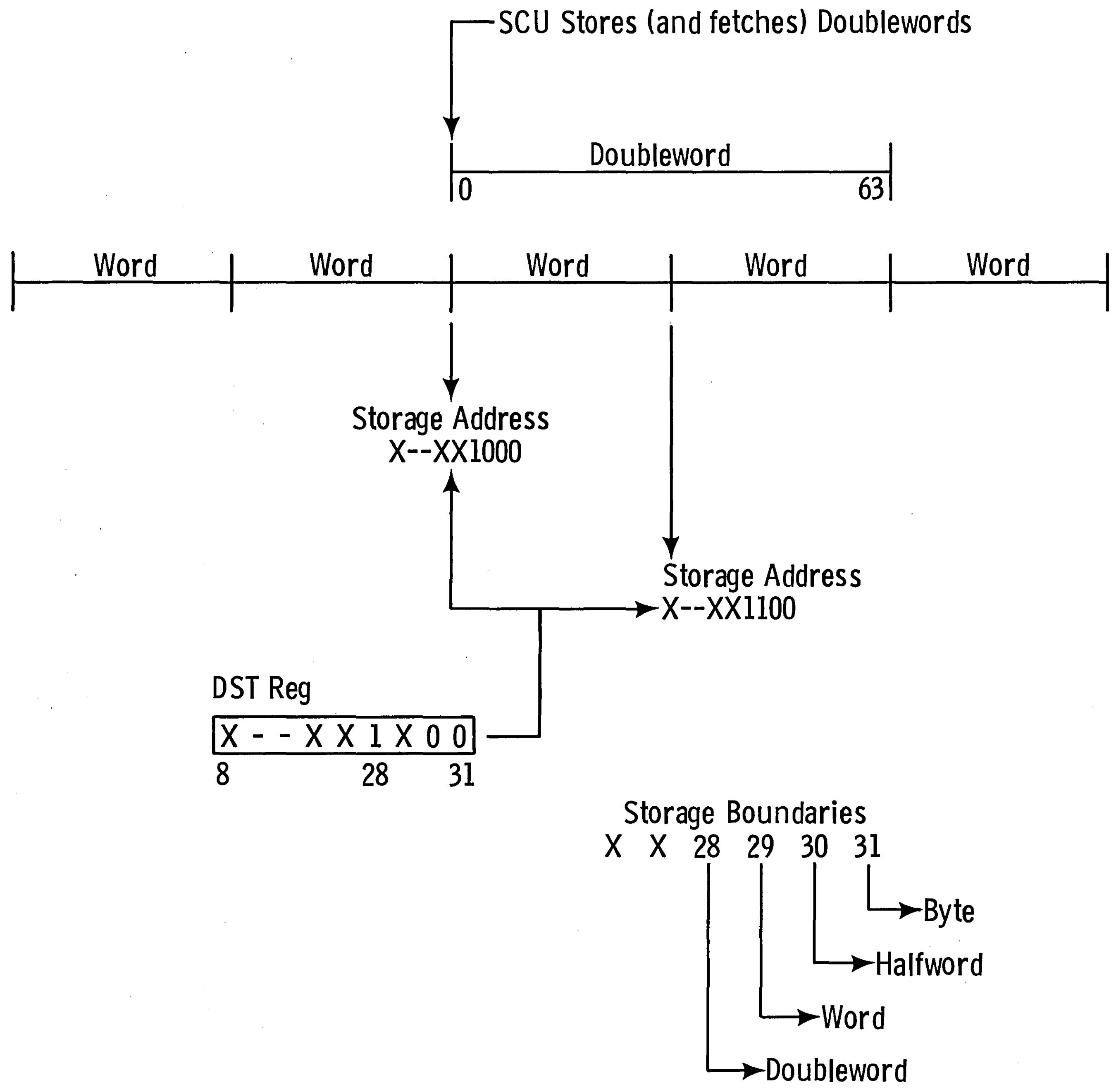


Figure 9-8. Store Concept 1

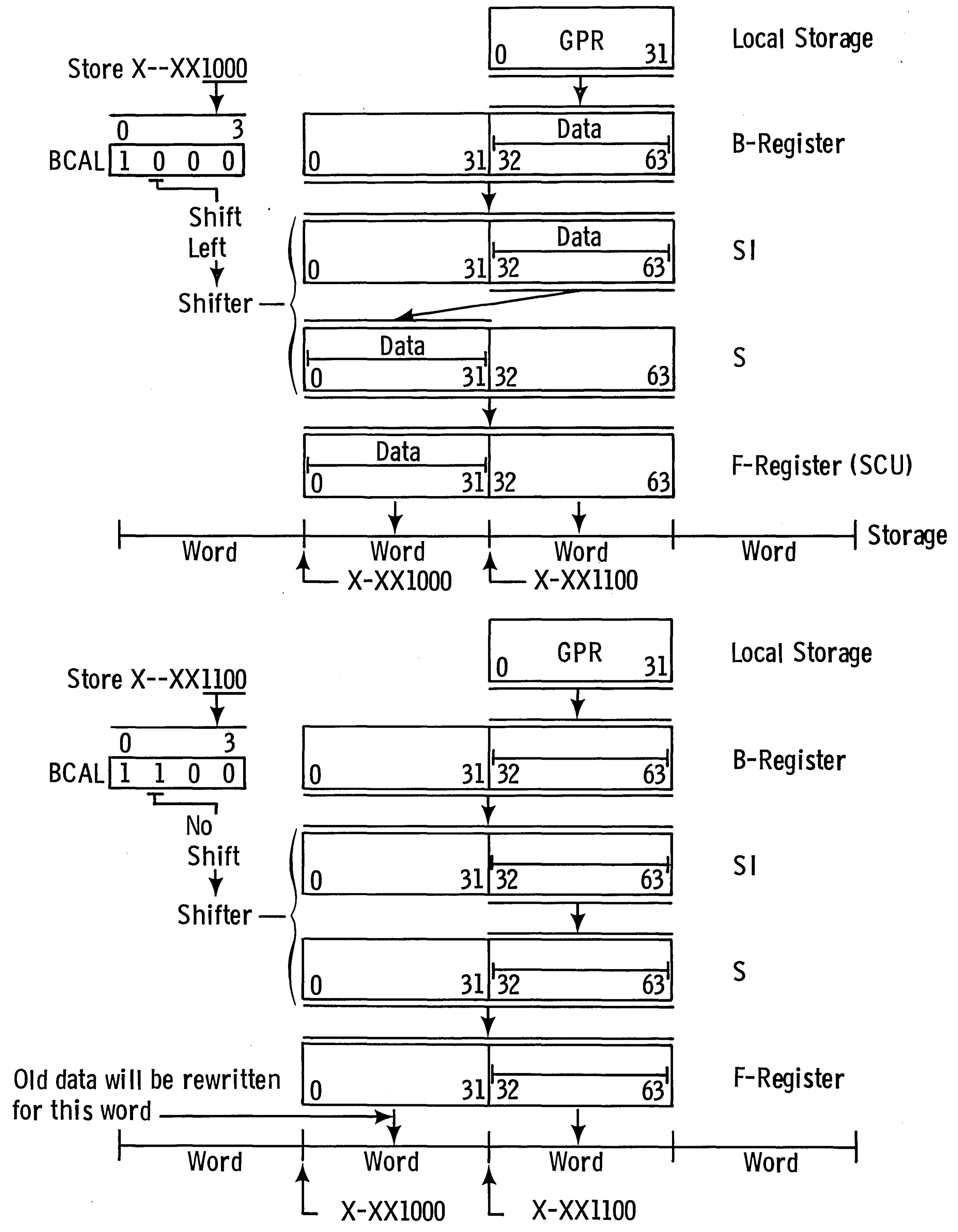


Figure 9-9. Store Concept 2

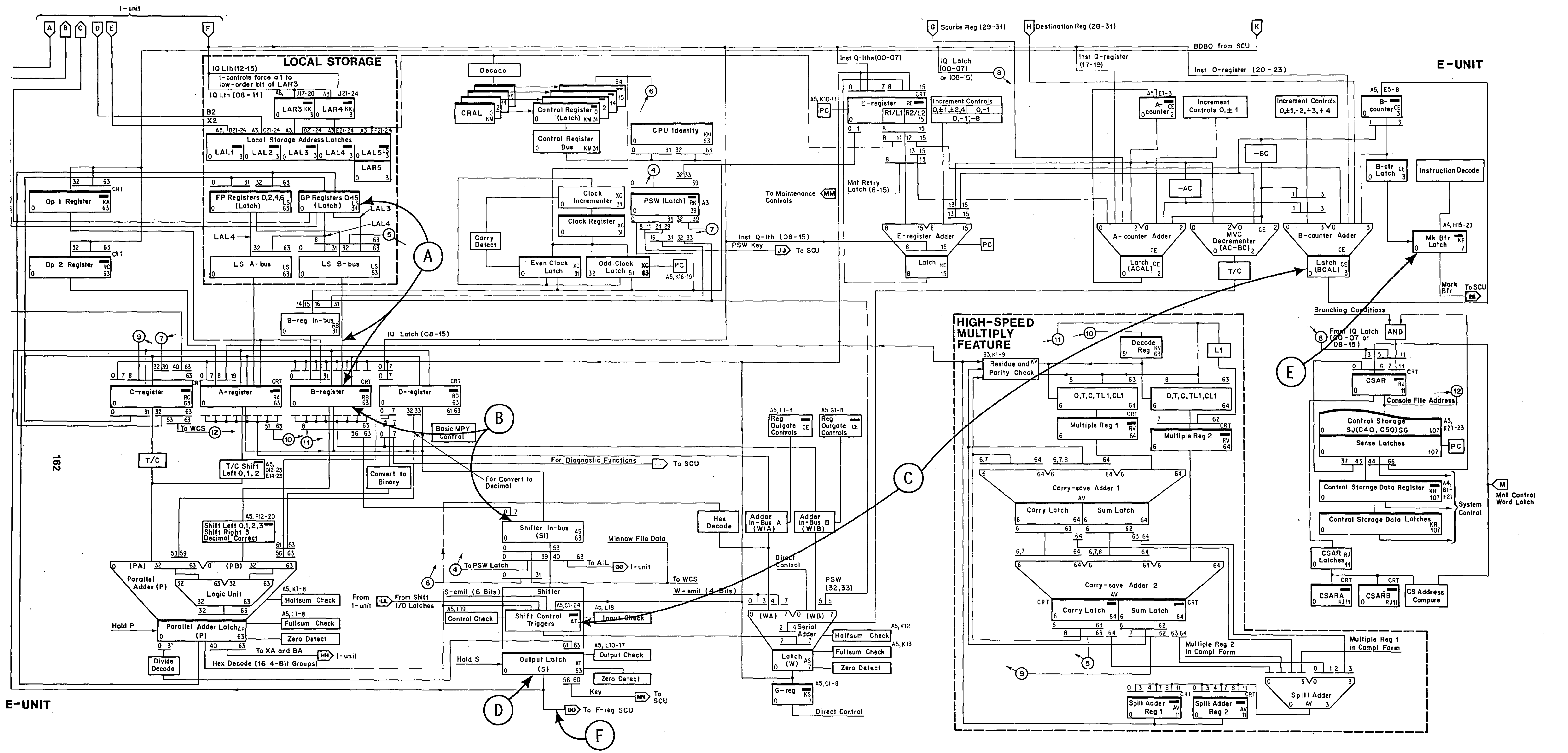


Figure 9-10. RX Store

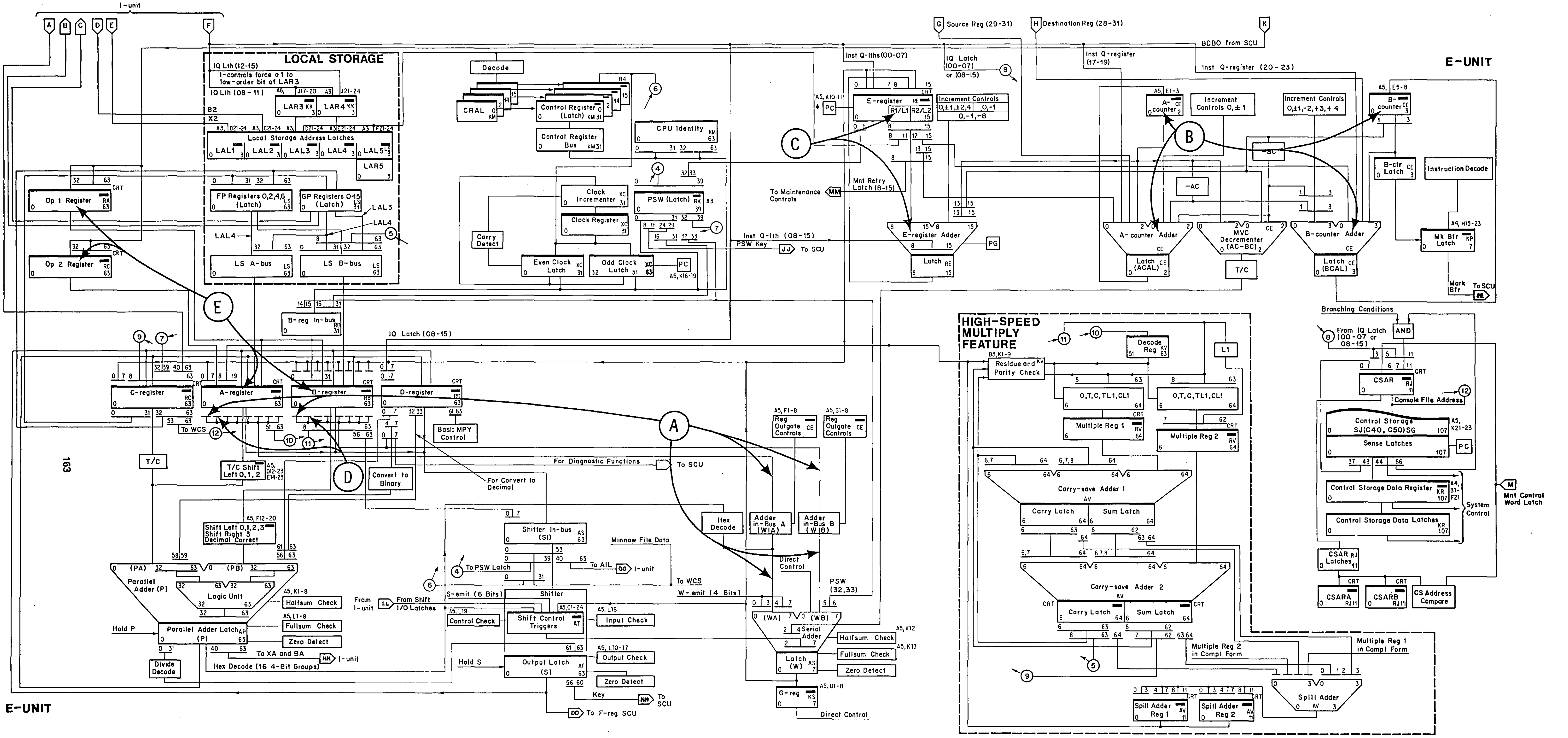


Figure 9-11. SS Compare Logical

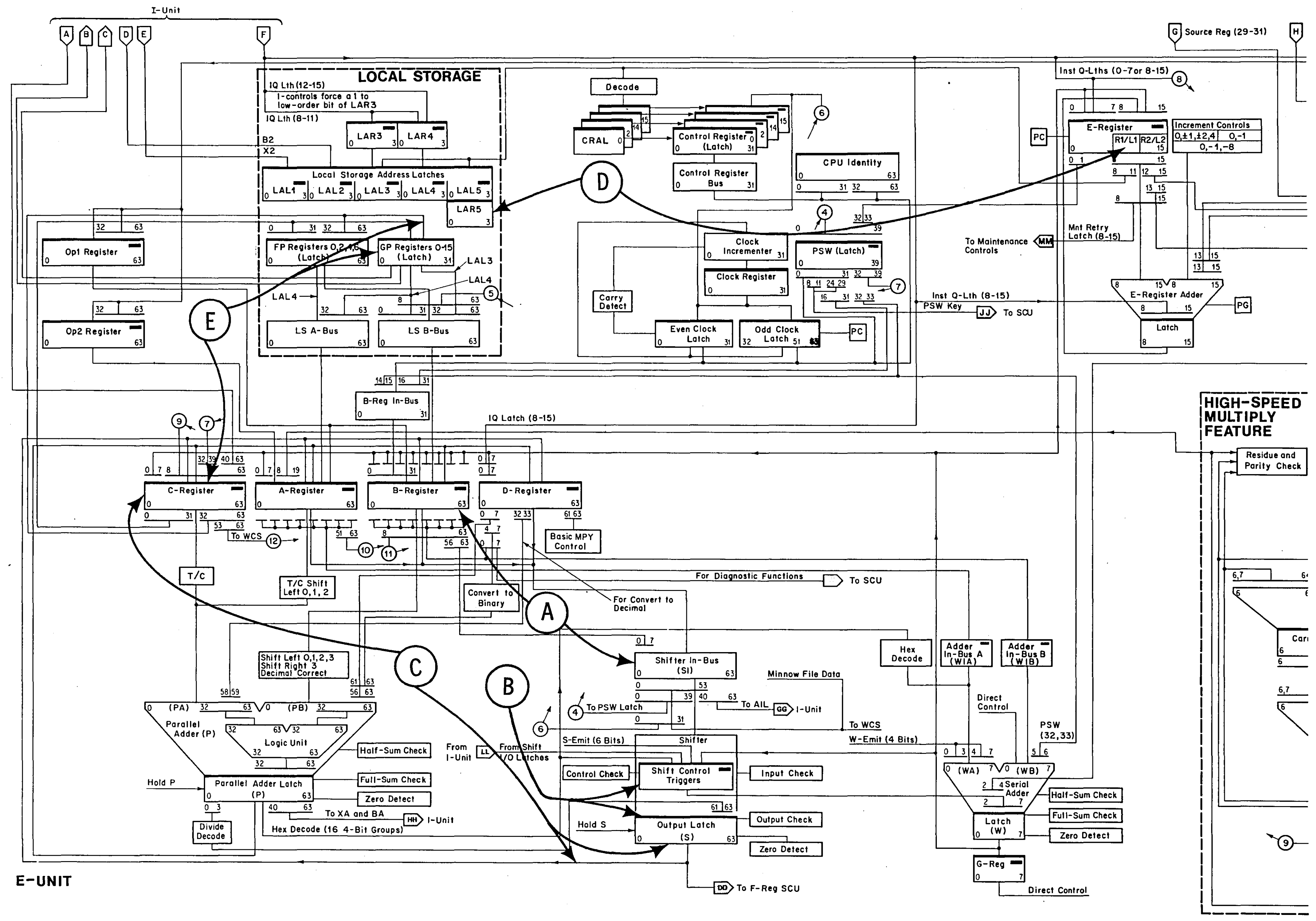


Figure 9-12. RS Shift Logical

START I/O

The start I/O (SIO) instruction selects a specified I/O device and initiates a channel command to that device. The following channel commands can be issued with the SIO instruction: read, read backward, write, sense, and control. The effective operand address (base plus displacement), determined during I-fetch of the SIO instruction, addresses the channel and I/O device. Bits 19-23 of the effective operand address are decoded to select the channel; bits 24-31 are sent to the channel as an eight-bit I/O device address, selecting the correct I/O device.

At the start of execution, the first 16 bits of the instruction are in the E-register and the second operand address is in the shift or I/O register; the first operand is not applicable. Because this instruction is an I/O instruction, the address in the shift or I/O register is the address of the channel and I/O device and is not to be interpreted as a main-storage address. Therefore, no data is requested from main storage. The SIO instruction can be executed only when the CPU is in the supervisor state. The first operation of the instruction, therefore, is to determine the state of the CPU. If the CPU is not in the supervisor state, a privileged-operation check occurs, causing a privileged-operation interrupt. If the CPU is in the supervisor state, execution of the SIO instruction begins by setting the select channel trigger. This trigger causes a select signal to be sent to the proper channel as determined by shift or I/O register bits 19-23.

The unit address in the shift or I/O register bits 24-31 also is sent to the channel. If at this point the selected channel is busy or in test mode, a CC of 2 or 3, respectively, is sent to the CPU. A release signal is also sent to the CPU, releasing it for execution of other instructions. If the channel is available, the unit address is gated to the unit address register in the channel. The channel then fetches the CAW from main-storage address 48 (hex). The CAW specifies the address of the first CCW and the storage protection key for all the channel commands associated with the SIO instruction. If any errors are discovered in the CAW or the unit address, a status byte is stored in the channel and a CC of 1 is sent to the CPU. A release signal is also sent to the CPU, releasing it for execution of other instructions.

Figure 9-13. Start I/O

Two operations, fetching the CCW and selecting the I/O unit per the unit address, are now started simultaneously. The channel starts a CCW fetch by raising a storage request to main storage. After the proper exchange of control signals, the command information (command code, data address, flags, and counts) is set into the proper registers in the channel. The CCW-valid trigger is set, if there were no errors, to show that the CCW has been received. The CCW information is then checked for correct parity.

The second operation, selection of the proper I/O devices, is started at the same time as the CCW fetch. To select the proper I/O device, the channel puts the unit address on the bus out to the control unit and sends an address-out signal followed 400 ns later by a select-out signal. The control unit responds with an operation-in signal, which causes the channel to drop the address-out signal, it puts the address of the selected device on the bus in and raises its address-in signal. The channel then compares the address it received from the control unit with the address it sent to the control unit to determine that proper selection has been made.

If the addresses are equal, the CCW-valid trigger is set, and if no errors occurred, the operation continues. The command code is placed on the bus out, and the command-out signal is sent to the control unit. The control unit responds with 0 status if it can accept the command. The channel then sends a condition code of 0 and a release signal to the CPU, releasing the CPU for further instruction execution. When the CPU receives the release signal, the select channel trigger is reset and an end-op cycle is taken, completing the operation.

Figure 9-13. Start I/O (Cont'd)

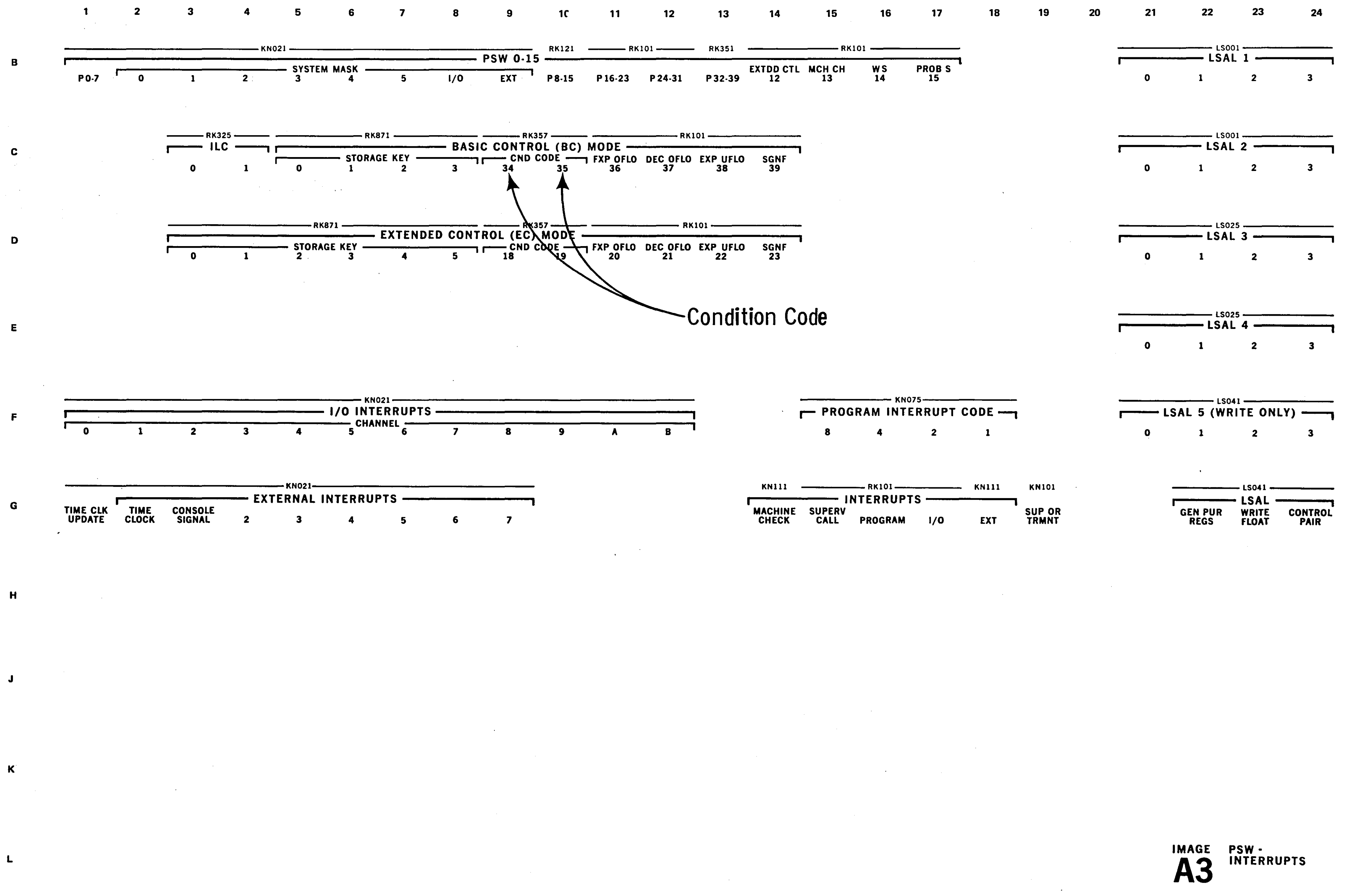
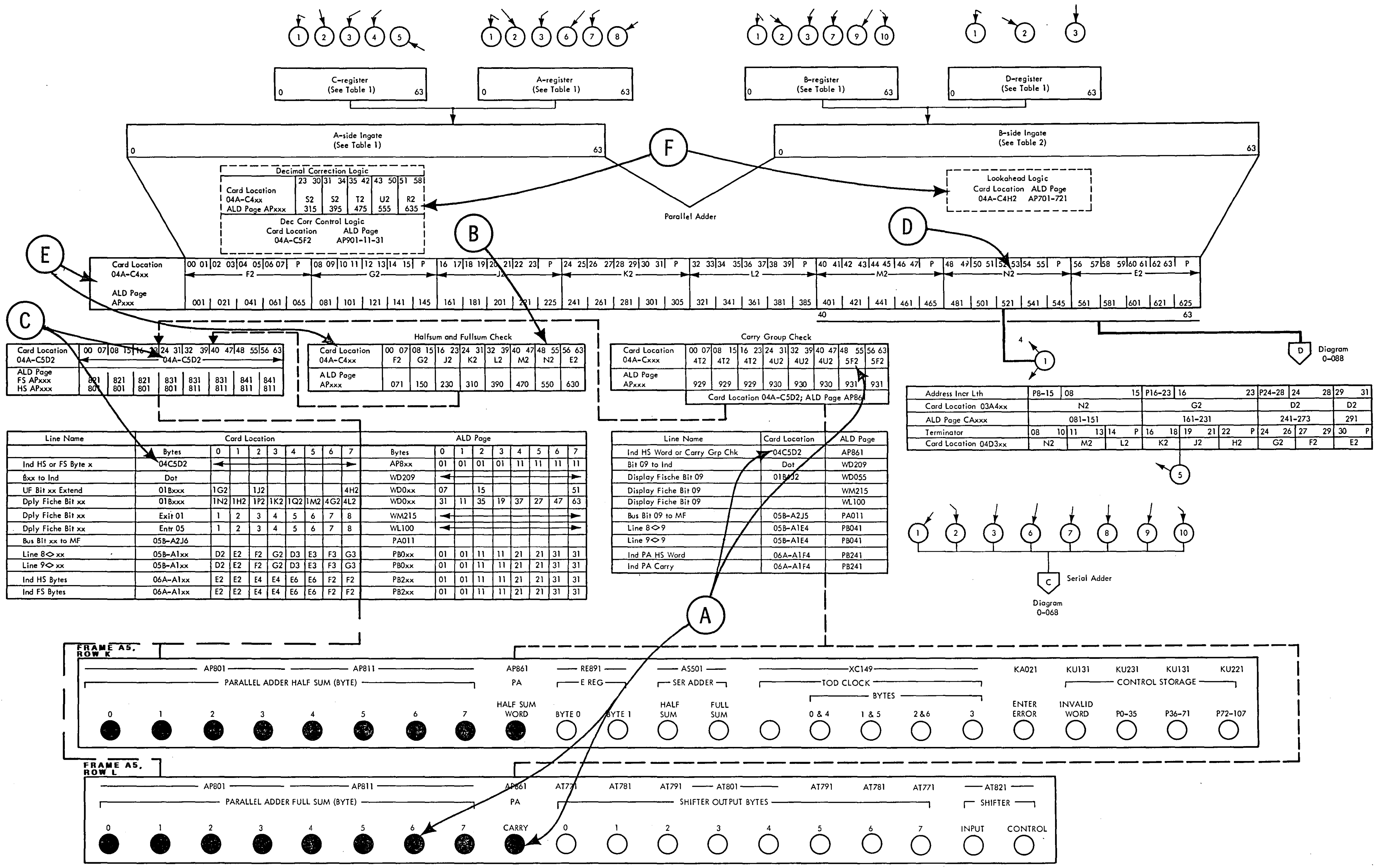


Figure 9-14. Condition Code



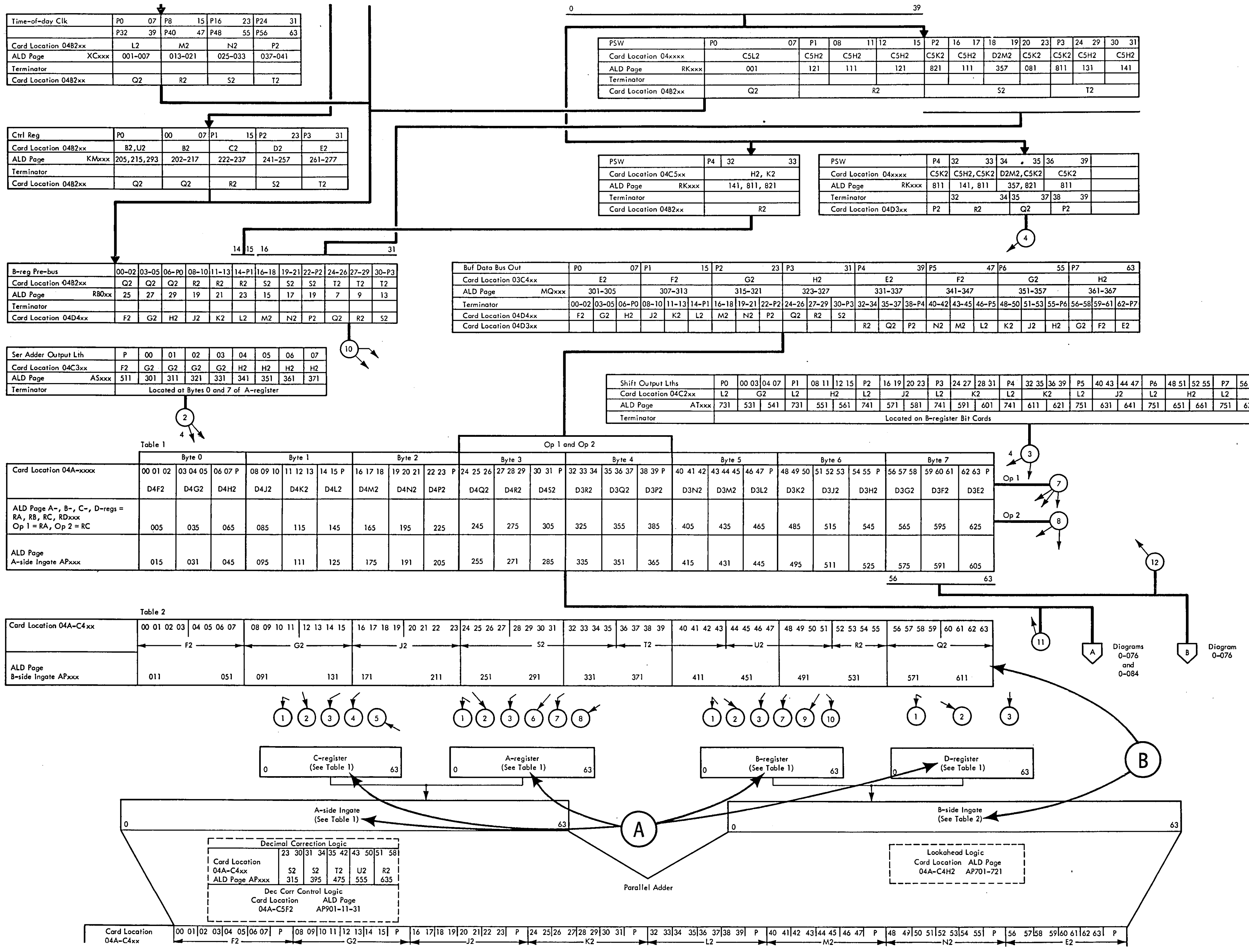
C-register (See Table 1) 0 63
A-register (See Table 1) 0 63
B-register (See Table 1) 0 63
D-register (See Table 1) 0 63
A-side Ingate (See Table 1) 0 63
B-side Ingate (See Table 2) 0 63
Decimal Correction Logic
Card Location 04A-C4xx
ALD Page APxxx
23 30 31 34 35 42 43 50 51 58
S2 S2 T2 U2 R2
315 395 475 555 635
Dec Corr Control Logic
Card Location ALD Page
04A-C5F2 AP901-11-31
Lookahead Logic
Card Location ALD Page
04A-C4H2 AP701-721
Parallel Adder
A B C D E F
Card Location 04A-C4xx
ALD Page APxxx
00 01 02 03 04 05 06 07 P F2 001 021 041 061 065
08 09 10 11 12 13 14 15 P G2 081 101 121 141 145
16 17 18 19 20 21 22 23 P J2 161 181 201 221 225
24 25 26 27 28 29 30 31 P K2 241 261 281 301 305
32 33 34 35 36 37 38 39 P L2 321 341 361 381 385
40 41 42 43 44 45 46 47 P M2 401 421 441 461 465
48 49 50 51 52 53 54 55 P N2 481 501 521 541 545
56 57 58 59 60 61 62 63 P E2 561 581 601 621 625
40
63
Card Location 04A-C5D2
00 07 08 15 16 23 24 31 32 39 40 47 48 55 56 63
04A-C5D2
ALD Page FS APxxx HS APxxx
821 801 821 801 821 801 831 801 831 811 831 811 841 811 841 811
Halfsum and Fullsum Check
Card Location 04A-C4xx
00 07 F2 08 15 G2 16 23 J2 24 31 K2 32 39 L2 40 47 M2 48 55 N2 56 63 E2
ALD Page APxxx 071 150 230 310 390 470 550 630
Carry Group Check
Card Location 04A-Cxxx
00 07 4T2 08 15 4T2 16 23 4T2 24 31 4U2 32 39 4U2 40 47 4U2 48 55 5F2 56 63 5F2
ALD Page APxxx 929 929 929 930 930 930 931 931
Card Location 04A-C5D2; ALD Page AP861
Diagram 0-088
Serial Adder
Diagram 0-068
FRAME A5, ROW K
AP801 AP811 AP861 RE891 AS501 XC149 KA021 KU131 KU231 KU131 KU221
PARALLEL ADDER HALF SUM (BYTE)
0 1 2 3 4 5 6 7
PA
HALF SUM WORD
E REG
BYTE 0 BYTE 1
SER ADDER
HALF SUM FULL SUM
TOD CLOCK
BYTES
0 & 4 1 & 5 2&6 3
ENTER ERROR
CONTROL STORAGE
INVALID WORD P0-35 P36-71 P72-107
FRAME A5, ROW L
AP801 AP811 AP861 AT771 AT781 AT791 AT801 AT791 AT781 AT771 AT821
PARALLEL ADDER FULL SUM (BYTE)
0 1 2 3 4 5 6 7
PA
CARRY
SHIFTER OUTPUT BYTES
0 1 2 3 4 5 6 7
SHIFTER
INPUT CONTROL

Line Name	Card Location									ALD Page								
	Bytes	0	1	2	3	4	5	6	7	Bytes	0	1	2	3	4	5	6	7
Ind HS or FS Byte x	04C5D2	←							→	AP8xx	01	01	01	01	11	11	11	11
Bxx to Ind	Dot									WD209	←							→
UF Bit xx Extend	01Bxxx	1G2		1J2					4H2	WD0xx	07		15					51
Dply Fiche Bit xx	01Bxxx	1N2	1H2	1P2	1K2	1Q2	1M2	4G2	4L2	WD0xx	31	11	35	19	37	27	47	63
Dply Fiche Bit xx	Exit 01	1	2	3	4	5	6	7	8	WM215	←							→
Dply Fiche Bit xx	Entr 05	1	2	3	4	5	6	7	8	WL100	←							→
Bus Bit xx to MF	05B-A2J6									PA011								
Line 8◇ xx	05B-A1xx	D2	E2	F2	G2	D3	E3	F3	G3	PB0xx	01	01	11	11	21	21	31	31
Line 9◇ xx	05B-A1xx	D2	E2	F2	G2	D3	E3	F3	G3	PB0xx	01	01	11	11	21	21	31	31
Ind HS Bytes	06A-A1xx	E2	E2	E4	E4	E6	E6	F2	F2	PB2xx	01	01	11	11	21	21	31	31
Ind FS Bytes	06A-A1xx	E2	E2	E4	E4	E6	E6	F2	F2	PB2xx	01	01	11	11	21	21	31	31

Line Name	Card Location	ALD Page
Ind HS Word or Carry Grp Chk	04C5D2	AP861
Bit 09 to Ind	Dot	WD209
Display Fische Bit 09	01B4J2	WD055
Display Fiche Bit 09		WM215
Display Fiche Bit 09		WL100
Bus Bit 09 to MF	05B-A2J5	PA011
Line 8◇9	05B-A1E4	PB041
Line 9◇9	05B-A1E4	PB041
Ind PA HS Word	06A-A1F4	PB241
Ind PA Carry	06A-A1F4	PB241

Address Incr Lth	P8-15	08 15		P16-23	16 23		P24-28	24 28	29 31
Card Location 03A4xx	N2			G2			D2		D2
ALD Page CAxxx	081-151			161-231			241-273		291
Terminator	08 10	11 13	14 P	16 18	19 21	22 P	24 26	27 29	30 P
Card Location 04D3xx	N2	M2	L2	K2	J2	H2	G2	F2	E2

Figure 10-1. Parallel Adder No 1



Time-of-day Clk	P0 07	P8 15	P16 23	P24 31
	P32 39	P40 47	P48 55	P56 63
Card Location 04B2xx	L2	M2	N2	P2
ALD Page XCxxx	001-007	013-021	025-033	037-041
Terminator				
Card Location 04B2xx	Q2	R2	S2	T2

Ctrl Reg	P0	00 07	P1 15	P2 23	P3 31
Card Location 04B2xx	B2,U2	B2	C2	D2	E2
ALD Page KMxxx	205,215,293	202-217	222-237	241-257	261-277
Terminator					
Card Location 04B2xx	Q2	Q2	R2	S2	T2

PSW	P0 07	P1	08 11	12 15	P2	16 17	18 19	20 23	P3	24 29	30 31
Card Location 04xxxx	C5L2	C5H2	C5H2	C5H2	C5K2	C5H2	D2M2	C5K2	C5K2	C5H2	C5H2
ALD Page RKxxx	001	121	111	121	821	111	357	081	811	131	141
Terminator											
Card Location 04B2xx	Q2	R2			S2				T2		

PSW	P4	32 33
Card Location 04C5xx	H2, K2	
ALD Page RKxxx	141, 811, 821	
Terminator		
Card Location 04B2xx	R2	

PSW	P4	32 33	34 35	36 39	
Card Location 04xxxx	C5K2	C5H2, C5K2	D2M2, C5K2	C5K2	
ALD Page RKxxx	811	141, 811	357, 821	811	
Terminator		32 34	35 37	38 39	
Card Location 04D3xx	P2	R2	Q2	P2	

B-reg Pre-bus	00-02	03-05	06-P0	08-10	11-13	14-P1	16-18	19-21	22-P2	24-26	27-29	30-P3
Card Location 04B2xx	Q2	Q2	Q2	R2	R2	R2	S2	S2	S2	T2	T2	T2
ALD Page RB0xx	25	27	29	19	21	23	15	17	19	7	9	13
Terminator												
Card Location 04D4xx	F2	G2	H2	J2	K2	L2	M2	N2	P2	Q2	R2	S2

Ser Adder Output Lth	P	00	01	02	03	04	05	06	07
Card Location 04C3xx	F2	G2	G2	G2	G2	H2	H2	H2	H2
ALD Page ASxxx	511	301	311	321	331	341	351	361	371
Terminator	Located at Bytes 0 and 7 of A-register								

Buf Data Bus Out	P0 07			P1 15			P2 23			P3 31			P4 39			P5 47			P6 55			P7 63		
Card Location 03C4xx	E2			F2			G2			H2			E2			F2			G2			H2		
ALD Page MQxxx	301-305			307-313			315-321			323-327			331-337			341-347			351-357			361-367		
Terminator	00-02	03-05	06-P0	08-10	11-13	14-P1	16-18	19-21	22-P2	24-26	27-29	30-P3	32-34	35-37	38-P4	40-42	43-45	46-P5	48-50	51-53	55-P6	56-58	59-61	62-P7
Card Location 04D4xx	F2	G2	H2	J2	K2	L2	M2	N2	P2	Q2	R2	S2												
Card Location 04D3xx													R2	Q2	P2	N2	M2	L2	K2	J2	H2	G2	F2	E2

Shift Output Lths	P0	00 03	04 07	P1	08 11	12 15	P2	16 19	20 23	P3	24 27	28 31	P4	32 35	36 39	P5	40 43	44 47	P6	48 51	52 55	P7	56 59	60 63
Card Location 04C2xx	L2	G2		L2	H2		L2	J2		L2	K2		L2	K2		L2	J2		L2	H2		L2	G2	
ALD Page ATxxx	731	531	541	731	551	561	741	571	581	741	591	601	741	611	621	751	631	641	751	651	661	751	671	681
Terminator	Located on B-register Bit Cards																							

Table 1

										Op 1 and Op 2														
	Byte 0			Byte 1			Byte 2			Byte 3			Byte 4			Byte 5			Byte 6			Byte 7		
Card Location 04A-xxxx	00 01 02 D4F2	03 04 05 D4G2	06 07 P D4H2	08 09 10 D4J2	11 12 13 D4K2	14 15 P D4L2	16 17 18 D4M2	19 20 21 D4N2	22 23 P D4P2	24 25 26 D4Q2	27 28 29 D4R2	30 31 P D4S2	32 33 34 D3R2	35 36 37 D3Q2	38 39 P D3P2	40 41 42 D3N2	43 44 45 D3M2	46 47 P D3L2	48 49 50 D3K2	51 52 53 D3J2	54 55 P D3H2	56 57 58 D3G2	59 60 61 D3F2	62 63 P D3E2
ALD Page A-, B-, C-, D-regs = RA, RB, RC, RDxxx Op 1 = RA, Op 2 = RC	005	035	065	085	115	145	165	195	225	245	275	305	325	355	385	405	435	465	485	515	545	565	595	625
ALD Page A-side Ingate APxxx	015	031	045	095	111	125	175	191	205	255	271	285	335	351	365	415	431	445	495	511	525	575	591	605

Table 2

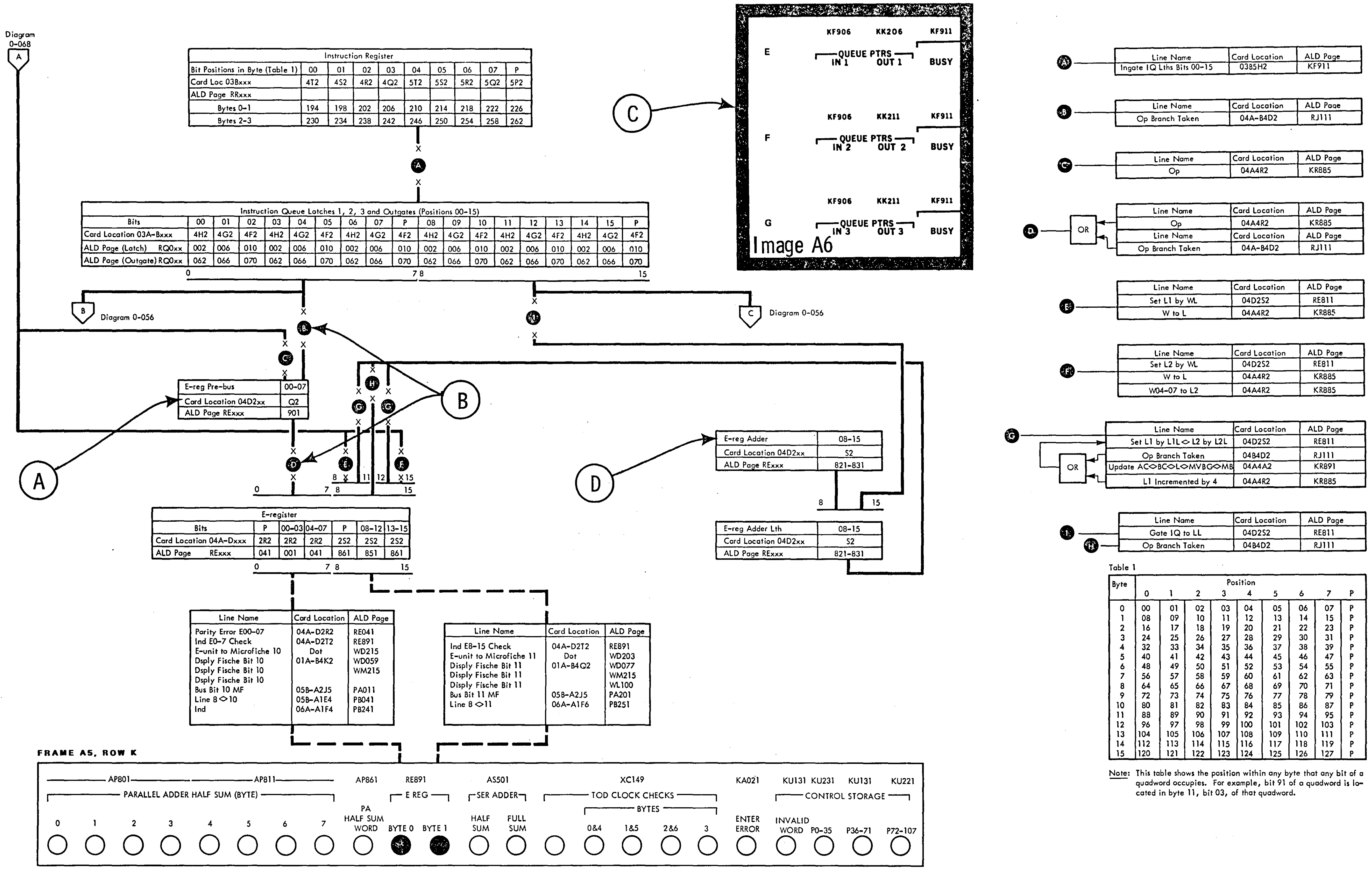
Card Location 04A-C4xx	00 01 02 03	04 05 06 07	08 09 10 11	12 13 14 15	16 17 18 19	20 21 22 23	24 25 26 27	28 29 30 31	32 33 34 35	36 37 38 39	40 41 42 43	44 45 46 47	48 49 50 51	52 53 54 55	56 57 58 59	60 61 62 63
	F2		G2		J2		S2			T2		U2		R2	Q2	
ALD Page B-side Ingate APxxx	011	051	091	131	171	211	251	291	331	371	411	451	491	531	571	611

Decimal Correction Logic

	23 30	31 34	35 42	43 50	51 58
Card Location 04A-C4xx ALD Page APxxx	S2 315	S2 395	T2 475	U2 555	R2 635

Card Location 04A-C4xx	00 01	02 03	04 05	06 07	P	08 09	10 11	12 13	14 15	P	16 17	18 19	20 21	22 23	P	24 25	26 27	28 29	30 31	P	32 33	34 35	36 37	38 39	P	40 41	42 43	44 45	46 47	P	48 49	50 51	52 53	54 55	P	56 57	58 59	60 61	62 63	P
	F2					G2					J2					K2					L2					M2					N2					E2				

Figure 10-2. Parallel Adder No 2



Instruction Register									
Bit Positions in Byte (Table 1)	00	01	02	03	04	05	06	07	P
Card Loc 03Bxxx	4T2	4S2	4R2	4Q2	5T2	5S2	5R2	5Q2	5P2
ALD Page RRxxx									
Bytes 0-1	194	198	202	206	210	214	218	222	226
Bytes 2-3	230	234	238	242	246	250	254	258	262

Instruction Queue Latches 1, 2, 3 and Outgates (Positions 00-15)																		
Bits	00	01	02	03	04	05	06	07	P	08	09	10	11	12	13	14	15	P
Card Location 03A-Bxxx	4H2	4G2	4F2	4H2	4G2	4F2	4H2	4G2	4F2	4H2	4G2	4F2	4H2	4G2	4F2	4H2	4G2	4F2
ALD Page (Latch) RQ0xx	002	006	010	002	006	010	002	006	010	002	006	010	002	006	010	002	006	010
ALD Page (Outgate) RQ0xx	062	066	070	062	066	070	062	066	070	062	066	070	062	066	070	062	066	070

E-reg Pre-bus	00-07
Card Location 04D2xx	Q2
ALD Page RExxx	901

E-register						
Bits	P	00-03	04-07	P	08-12	13-15
Card Location 04A-Dxxx	2R2	2R2	2R2	2S2	2S2	2S2
ALD Page RExxx	041	001	041	861	851	861

E-reg Adder	08-15
Card Location 04D2xx	S2
ALD Page RExxx	821-831

E-reg Adder Lth	08-15
Card Location 04D2xx	S2
ALD Page RExxx	821-831

Line Name	Card Location	ALD Page
Parity Error E00-07	04A-D2R2	RE041
Ind E0-7 Check	04A-D2T2	RE891
E-unit to Microfiche 10	Dot	WD215
Dsply Fische Bit 10	01A-B4K2	WD059
Dsply Fische Bit 10		WM215
Dsply Fische Bit 10		
Bus Bit 10 MF	05B-A2J5	PA011
Line 8 ◇ 10	05B-A1E4	PB041
Ind	06A-A1F4	PB241

Line Name	Card Location	ALD Page
Ind E8-15 Check	04A-D2T2	RE891
E-unit to Microfiche 11	Dot	WD203
Disply Fische Bit 11	01A-B4Q2	WD077
Disply Fische Bit 11		WM215
Disply Fische Bit 11		WL100
Bus Bit 11 MF	05B-A2J5	PA201
Line 8 ◇ 11	06A-A1F6	PB251

A

Line Name	Card Location	ALD Page
Ingate IQ Lths Bits 00-15	03B5H2	KF911

B

Line Name	Card Location	ALD Page
Op Branch Taken	04A-B4D2	RJ111

C

Line Name	Card Location	ALD Page
Op	04A4R2	KR885

D

Line Name	Card Location	ALD Page
Op	04A4R2	KR885
Line Name	Card Location	ALD Page
Op Branch Taken	04A-B4D2	RJ111

E

Line Name	Card Location	ALD Page
Set L1 by WL	04D2S2	RE811
W to L	04A4R2	KR885

F

Line Name	Card Location	ALD Page
Set L2 by WL	04D2S2	RE811
W to L	04A4R2	KR885
W04-07 to L2	04A4R2	KR885

G

Line Name	Card Location	ALD Page
Set L1 by L1L ◇ L2 by L2L	04D2S2	RE811
Op Branch Taken	04B4D2	RJ111
Update AC◇BC◇L◇MVBG◇MB	04A4A2	KR891
L1 Incremented by 4	04A4R2	KR885

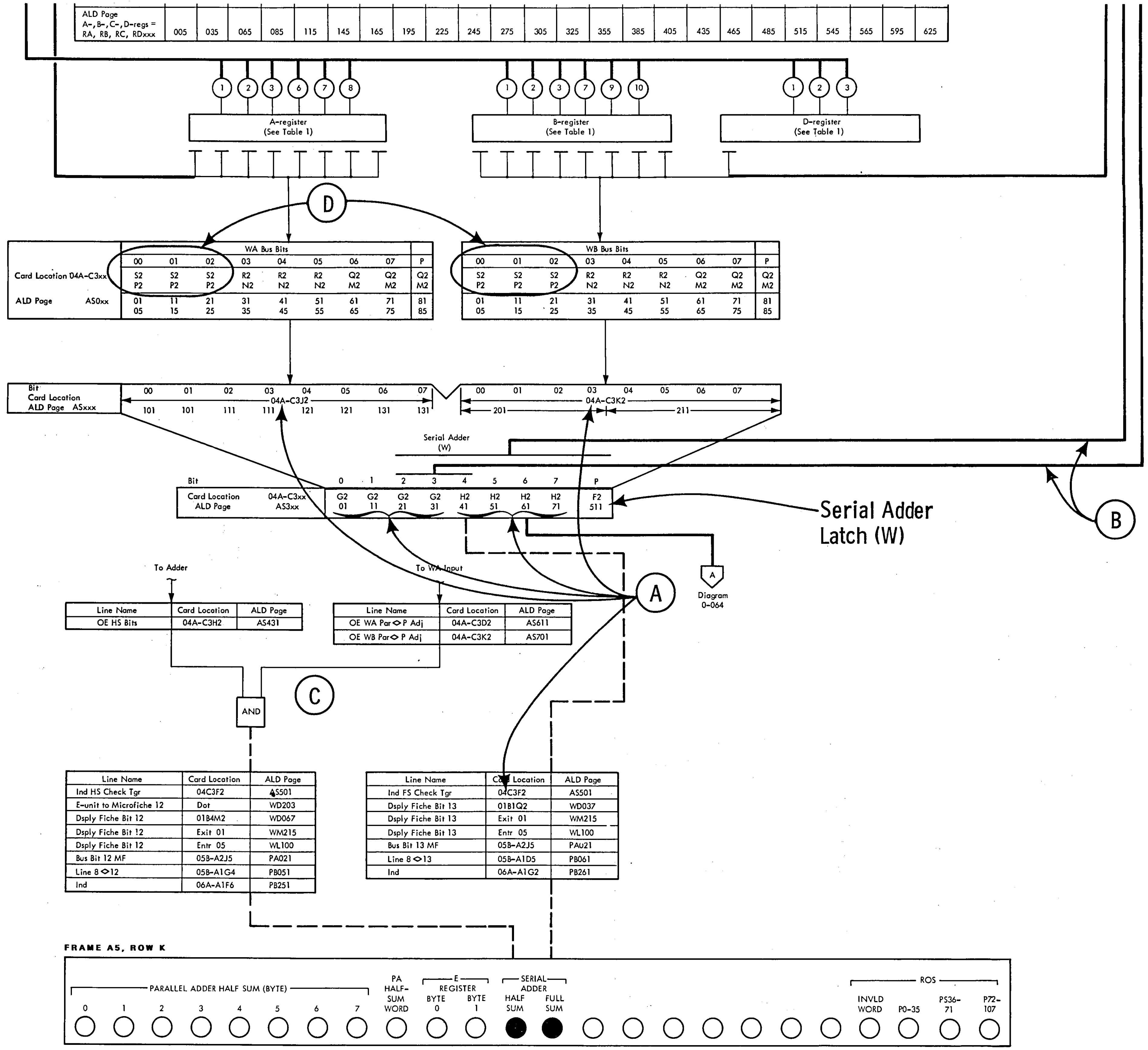
Line Name	Card Location	ALD Page
Gate IQ to LL	04D2S2	RE811
Op Branch Taken	04B4D2	RJ111

Table 1

Byte	Position								
	0	1	2	3	4	5	6	7	P
0	00	01	02	03	04	05	06	07	P
1	08	09	10	11	12	13	14	15	P
2	16	17	18	19	20	21	22	23	P
3	24	25	26	27	28	29	30	31	P
4	32	33	34	35	36	37	38	39	P
5	40	41	42	43	44	45	46	47	P
6	48	49	50	51	52	53	54	55	P
7	56	57	58	59	60	61	62	63	P
8	64	65	66	67	68	69	70	71	P
9	72	73	74	75	76	77	78	79	P
10	80	81	82	83	84	85	86	87	P
11	88	89	90	91	92	93	94	95	P
12	96	97	98	99	100	101	102	103	P
13	104	105	106	107	108	109	110	111	P
14	112	113	114	115	116	117	118	119	P
15	120	121	122	123	124	125	126	127	P

Note: This table shows the position within any byte that any bit of a quadword occupies. For example, bit 91 of a quadword is located in byte 11, bit 03, of that quadword.

Figure 10-3. E-Register ECAD



Line Name	Card Location	ALD Page
OE HS Bits	04A-C3H2	AS431

Line Name	Card Location	ALD Page
OE WA Par<>P Adj	04A-C3D2	AS611
OE WB Par<>P Adj	04A-C3K2	AS701

Line Name	Card Location	ALD Page
Ind HS Check Tgr	04C3F2	AS501
E-unit to Microfiche 12	Dot	WD203
Dsply Fiche Bit 12	01B4M2	WD067
Dsply Fiche Bit 12	Exit 01	WM215
Dsply Fiche Bit 12	Entr 05	WL100
Bus Bit 12 MF	05B-A2J5	PA021
Line 8 <>12	05B-A1G4	PB051
Ind	06A-A1F6	PB251

Line Name	Card Location	ALD Page
Ind FS Check Tgr	04C3F2	AS501
Dsply Fiche Bit 13	01B1Q2	WD037
Dsply Fiche Bit 13	Exit 01	WM215
Dsply Fiche Bit 13	Entr 05	WL100
Bus Bit 13 MF	05B-A2J5	PA021
Line 8 <>13	05B-A1D5	PB061
Ind	06A-A1G2	PB261

Figure 10-4. Serial Adder

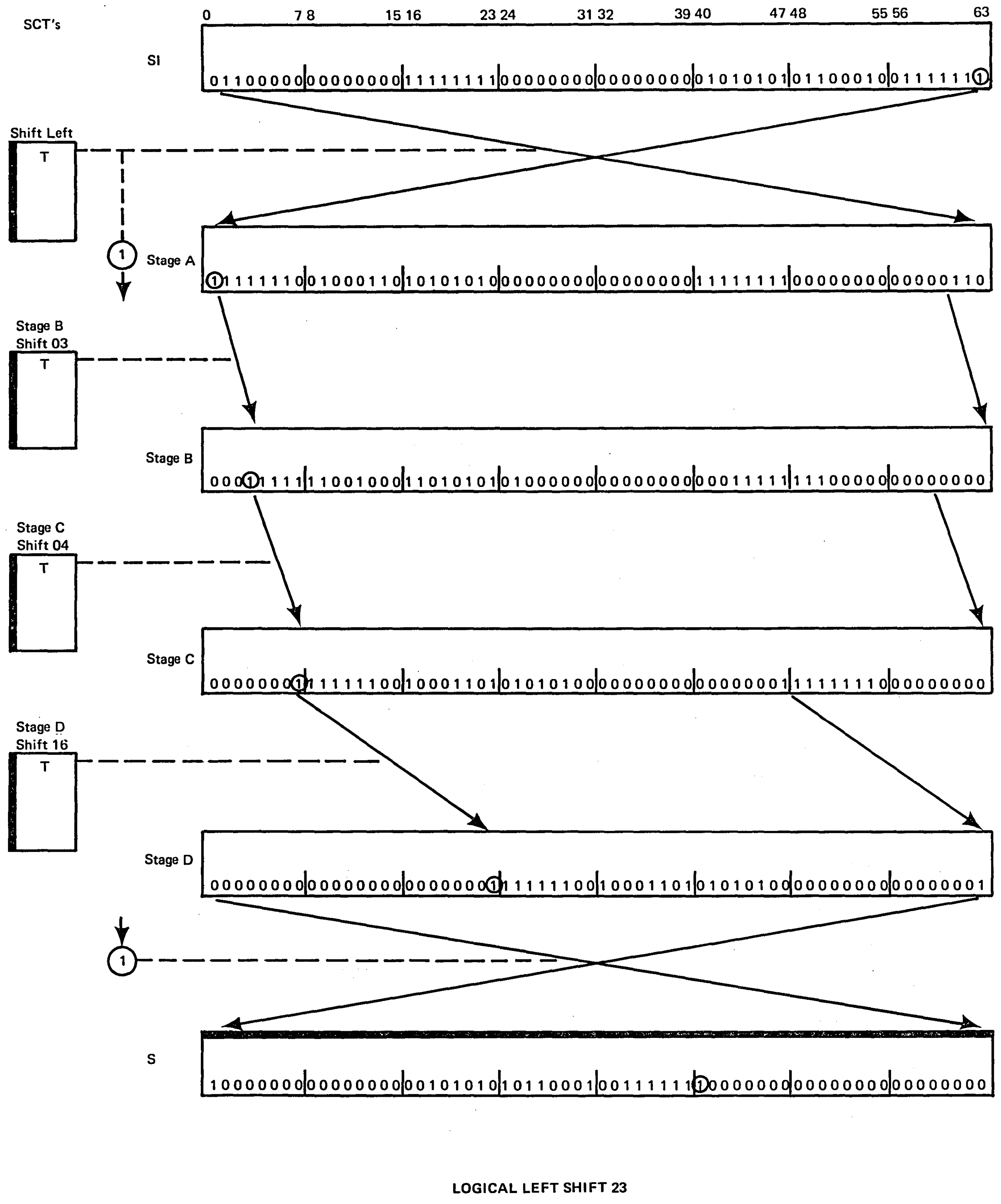


Figure 10-5. Logical Left Shift 23

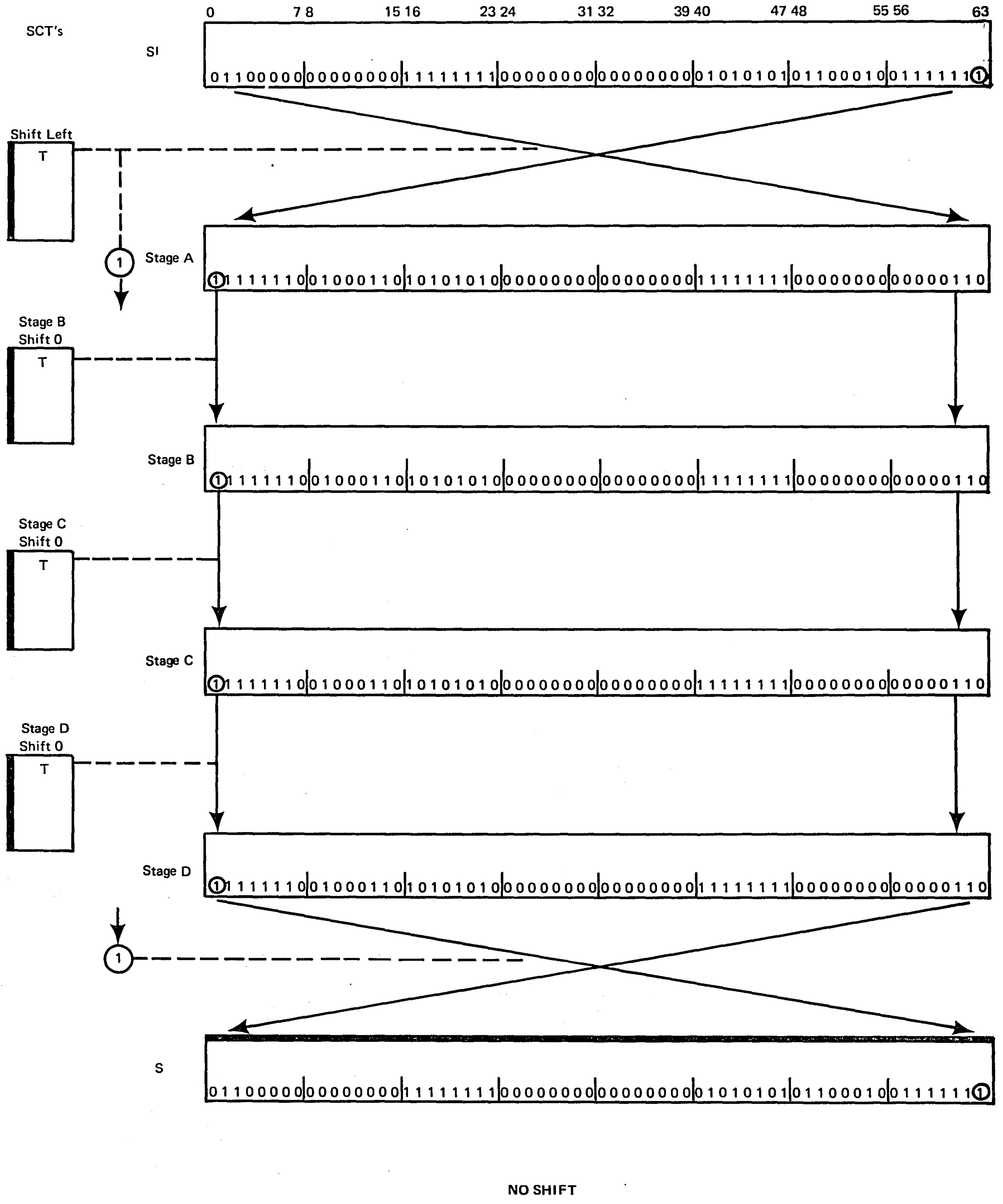


Figure 10-6. No Shift

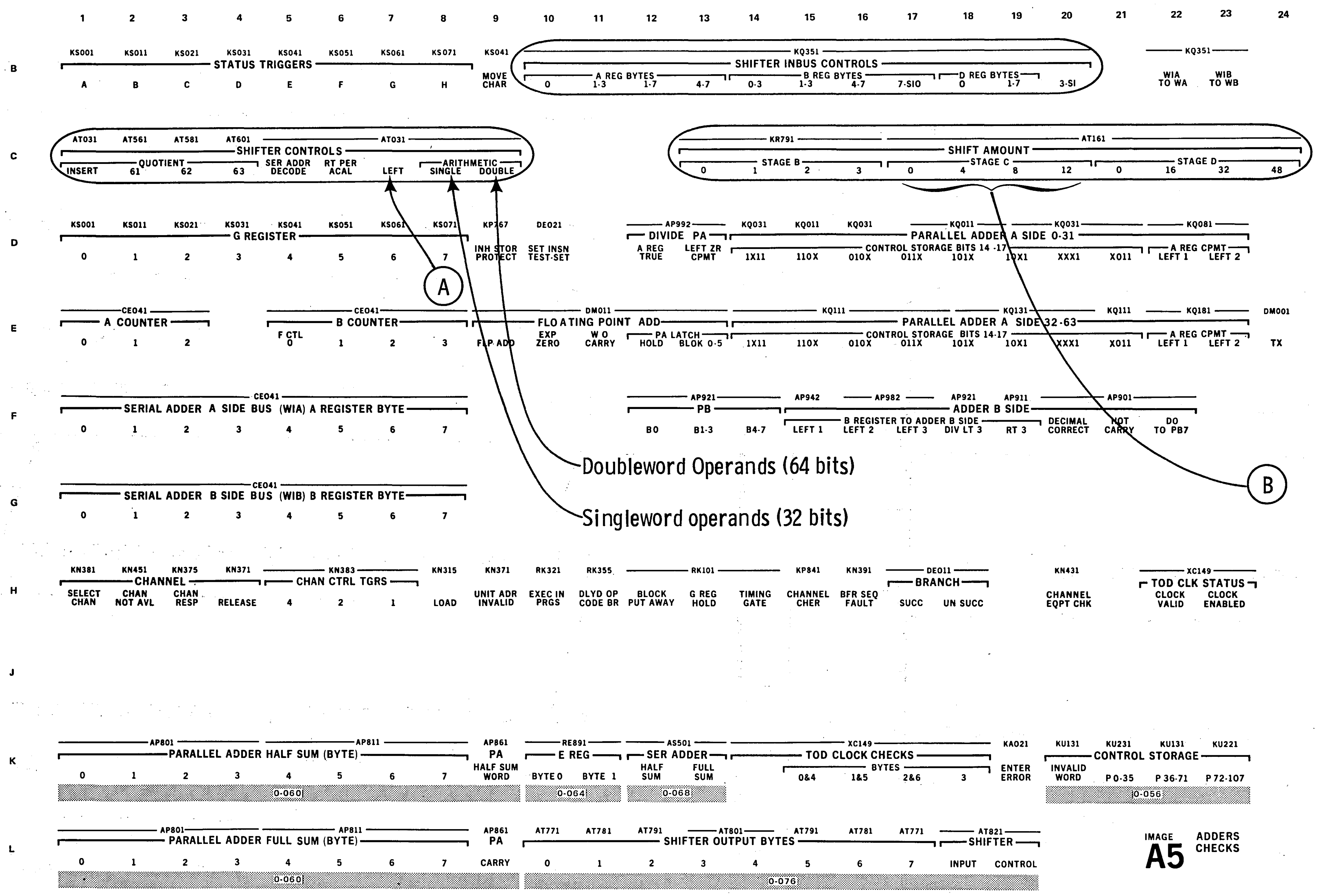

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
B
KS001 KS011 KS021 KS031 KS041 KS051 KS061 KS071 STATUS TRIGGERS A B C D E F G H
KS041 MOVE CHAR
KQ351 SHIFTER INBUS CONTROLS
A REG BYTES 0 1-3 1-7 4-7
B REG BYTES 0-3 1-3 4-7 7-SIO
D REG BYTES 0 1-7 3-SI
KQ351 WIA TO WA WIB TO WB
C
AT031 AT561 AT581 AT601 AT031 SHIFTER CONTROLS
INSERT QUOTIENT 61 62 63 SER ADDR DECODE RT PER ACAL LEFT ARITHMETIC SINGLE DOUBLE
KR791 AT161 SHIFT AMOUNT
STAGE B 0 1 2 3
STAGE C 0 4 8 12
STAGE D 0 16 32 48
D
KS001 KS011 KS021 KS031 KS041 KS051 KS061 KS071 G REGISTER 0 1 2 3 4 5 6 7
KP767 INH STOR PROTECT
DE021 SET INSN TEST-SET
AP992 DIVIDE PA A REG TRUE LEFT ZR CPMT
KQ031 KQ011 KQ031 KQ011 KQ031 PARALLEL ADDER A SIDE 0-31
CONTROL STORAGE BITS 14-17 1X11 110X 010X 011X 101X 10X1 XXX1 X011
KQ081 A REG CPMT LEFT 1 LEFT 2
A
E
CE041 A COUNTER 0 1 2
CE041 B COUNTER F CTL 0 1 2 3
DM011 FLOATING POINT ADD FLP ADD EXP ZERO W O CARRY
PA LATCH HOLD BLOK 0-5
KQ111 KQ131 KQ111 PARALLEL ADDER A SIDE 32-63
CONTROL STORAGE BITS 14-17 1X11 110X 010X 011X 101X 10X1 XXX1 X011
KQ181 A REG CPMT LEFT 1 LEFT 2
DM001 TX
F
CE041 SERIAL ADDER A SIDE BUS (WIA) A REGISTER BYTE 0 1 2 3 4 5 6 7
AP921 PB B0 B1-3 B4-7
AP942 AP982 AP921 AP911 AP901 ADDER B SIDE
B REGISTER TO ADDER B SIDE LEFT 1 LEFT 2 LEFT 3 DIV LT 3 RT 3
DECIMAL CORRECT NOT CARRY DO TO PB7
Doubleword Operands (64 bits)
Singleword operands (32 bits)
B
G
CE041 SERIAL ADDER B SIDE BUS (WIB) B REGISTER BYTE 0 1 2 3 4 5 6 7
H
KN381 KN451 KN375 KN371 CHANNEL SELECT CHAN CHAN NOT AVL CHAN RESP RELEASE
KN383 CHAN CTRL TGRS 4 2 1
KN315 LOAD
KN371 UNIT ADR INVALID
RK321 EXEC IN PRGS
RK355 DLYD OP CODE BR
RK101 BLOCK PUT AWAY G REG HOLD TIMING GATE
KP841 CHANNEL CHER
KN391 BFR SEQ FAULT
DE011 BRANCH SUCC UN SUCC
KN431 CHANNEL EQPT CHK
XC149 TOD CLK STATUS CLOCK VALID CLOCK ENABLED
J
K
AP801 AP811 PARALLEL ADDER HALF SUM (BYTE) 0 1 2 3 4 5 6 7
AP861 PA HALF SUM WORD
RE891 E REG BYTE 0 BYTE 1
AS501 SER ADDER HALF SUM FULL SUM
XC149 TOD CLOCK CHECKS BYTES 0&4 1&5 2&6 3
KA021 ENTER ERROR
KU131 KU231 KU131 KU221 CONTROL STORAGE INVALID WORD P 0-35 P 36-71 P 72-107
0-060 0-064 0-068 0-056
L
AP801 AP811 PARALLEL ADDER FULL SUM (BYTE) 0 1 2 3 4 5 6 7
AP861 PA CARRY
AT771 AT781 AT791 AT801 AT791 AT781 AT771 SHIFTER OUTPUT BYTES 0 1 2 3 4 5 6 7
AT821 SHIFTER INPUT CONTROL
0-060 0-076
IMAGE A5 ADDERS CHECKS

Figure 10-7. Shifter Control Indicators

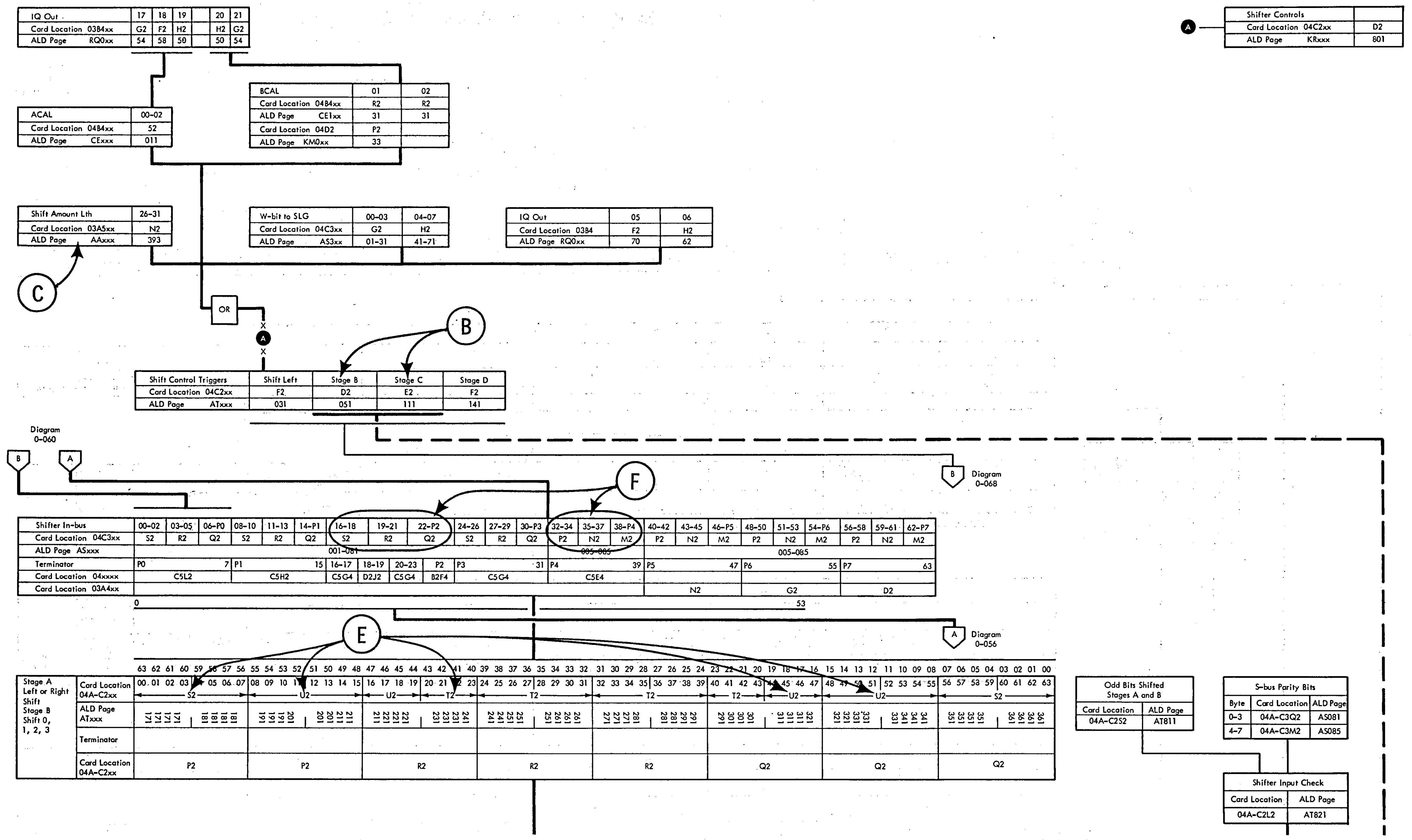
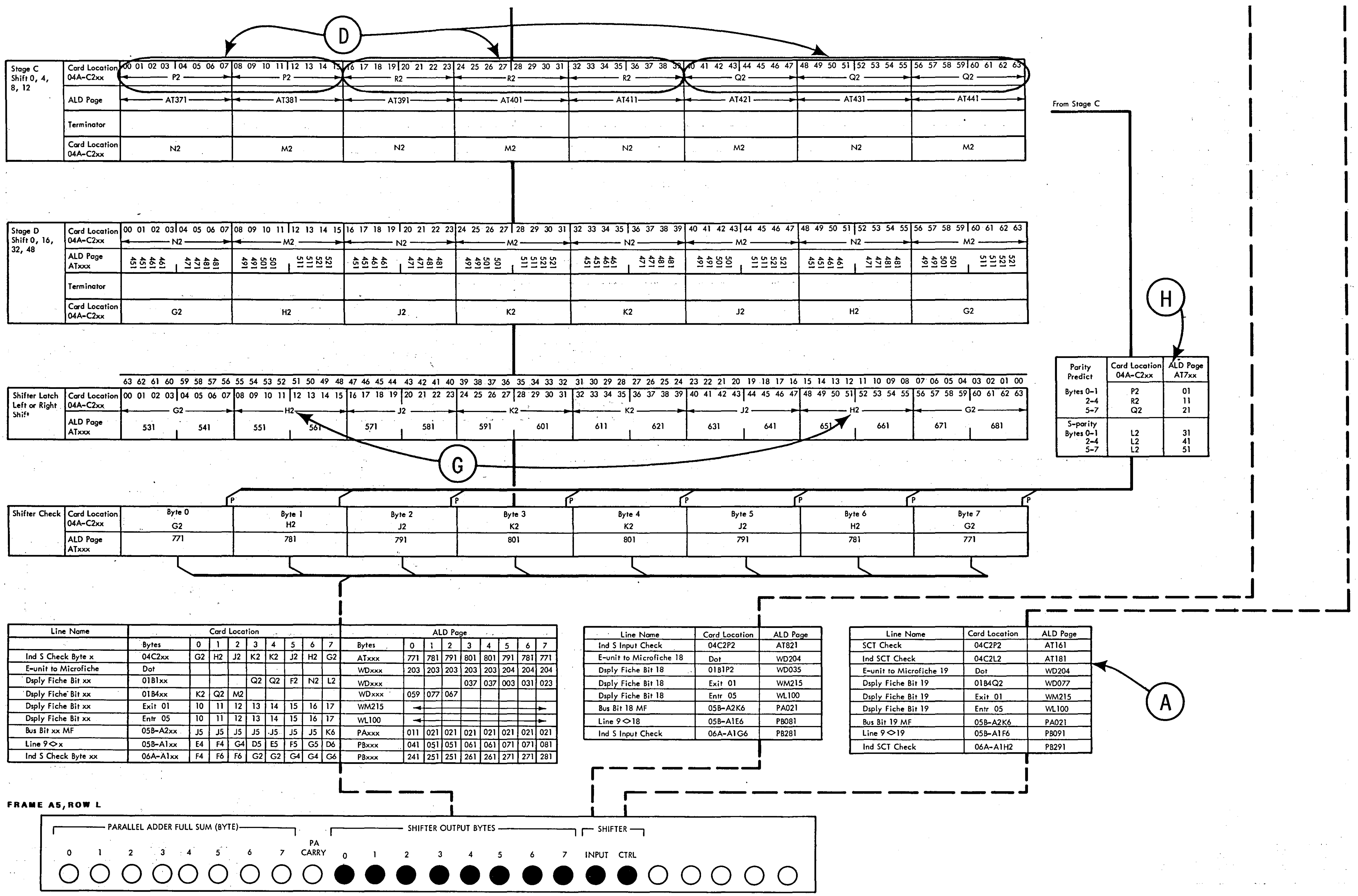


Figure 10-8a. Shifter ECAD



Parity Predict	Card Location 04A-C2xx	ALD Page AT7xx
Bytes 0–1	P2	01
2–4	R2	11
5–7	Q2	21
S-parity Bytes 0–1	L2	31
2–4	L2	41
5–7	L2	51

Line Name	Card Location									ALD Page								
	Bytes	0	1	2	3	4	5	6	7	Bytes	0	1	2	3	4	5	6	7
Ind S Check Byte x	04C2xx	G2	H2	J2	K2	K2	J2	H2	G2	ATxxx	771	781	791	801	801	791	781	771
E-unit to Microfiche	Dot									WDxxx	203	203	203	203	203	204	204	204
Dsply Fiche Bit xx	01B1xx				Q2	Q2	F2	N2	L2	WDxxx				037	037	003	031	023
Dsply Fiche Bit xx	01B4xx	K2	Q2	M2						WDxxx	059	077	067					
Dsply Fiche Bit xx	Exit 01	10	11	12	13	14	15	16	17	WM215	←							→
Dsply Fiche Bit xx	Entr 05	10	11	12	13	14	15	16	17	WL100	←							→
Bus Bit xx MF	05B-A2xx	J5	J5	J5	J5	J5	J5	J5	K6	PAxxx	011	021	021	021	021	021	021	021
Line 9<>x	05B-A1xx	E4	F4	G4	D5	E5	F5	G5	D6	PBxxx	041	051	051	061	061	071	071	081
Ind S Check Byte xx	06A-A1xx	F4	F6	F6	G2	G2	G4	G4	G6	PBxxx	241	251	251	261	261	271	271	281

Line Name	Card Location	ALD Page
Ind S Input Check	04C2P2	AT821
E-unit to Microfiche 18	Dot	WD204
Dsply Fiche Bit 18	01B1P2	WD035
Dsply Fiche Bit 18	Exit 01	WM215
Dsply Fiche Bit 18	Entr 05	WL100
Bus Bit 18 MF	05B-A2K6	PA021
Line 9<>18	05B-A1E6	PB081
Ind S Input Check	06A-A1G6	PB281

Line Name	Card Location	ALD Page
SCT Check	04C2P2	AT161
Ind SCT Check	04C2L2	AT181
E-unit to Microfiche 19	Dot	WD204
Dsply Fiche Bit 19	01B4Q2	WD077
Dsply Fiche Bit 19	Exit 01	WM215
Dsply Fiche Bit 19	Entr 05	WL100
Bus Bit 19 MF	05B-A2K6	PA021
Line 9<>19	05B-A1F6	PB091
Ind SCT Check	06A-A1H2	PB291

Figure 10-8b. Shifter ECAD

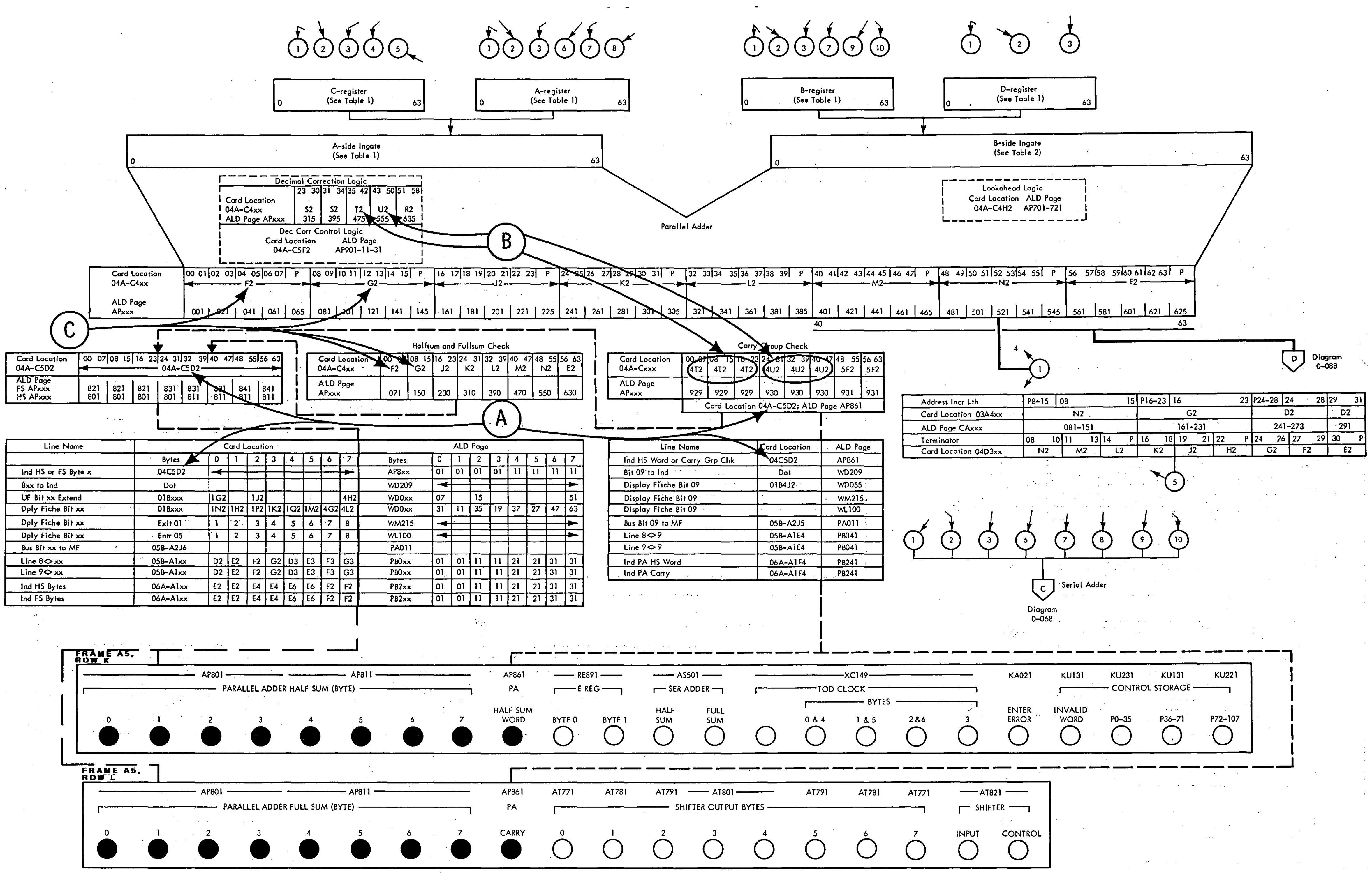


Figure 10-9. Parallel Adder Functional Packaging

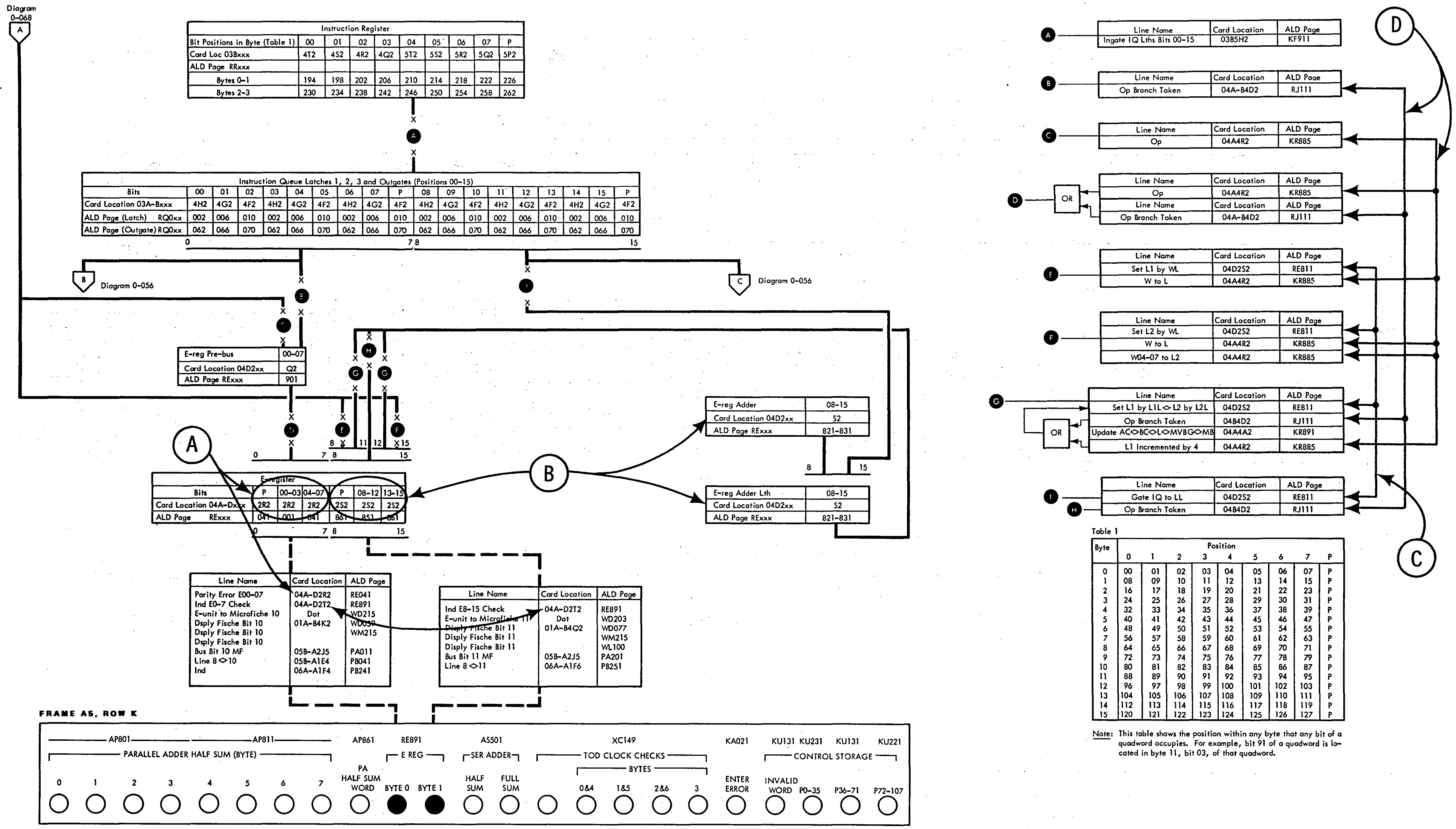


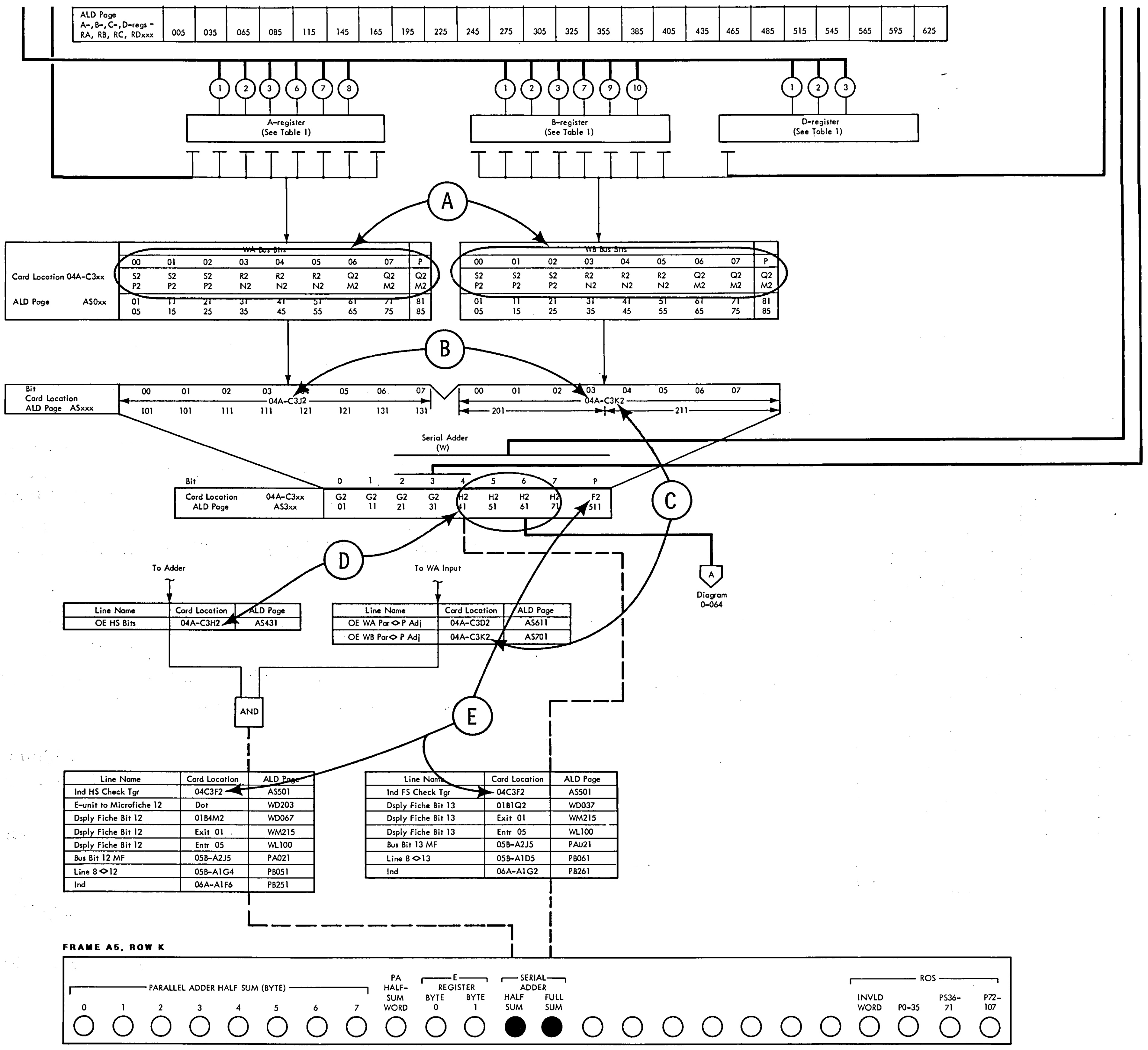
Diagram 0-068
A
Instruction Register
Bit Positions in Byte (Table 1) | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | P
Card Loc 03Bxxx | 4T2 | 4S2 | 4R2 | 4Q2 | 5T2 | 5S2 | 5R2 | 5Q2 | 5P2
ALD Page RRxxx
Bytes 0-1 | 194 | 198 | 202 | 206 | 210 | 214 | 218 | 222 | 226
Bytes 2-3 | 230 | 234 | 238 | 242 | 246 | 250 | 254 | 258 | 262
Instruction Queue Latches 1, 2, 3 and Outgates (Positions 00-15)
Bits | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | P | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 15 | P
Card Location 03A-Bxxx | 4H2 | 4G2 | 4F2 | 4H2 | 4G2 | 4F2 | 4H2 | 4G2 | 4F2 | 4H2 | 4G2 | 4F2 | 4H2 | 4G2 | 4F2 | 4H2 | 4G2 | 4F2
ALD Page (Latch) RQ0xx | 002 | 006 | 010 | 002 | 006 | 010 | 002 | 006 | 010 | 002 | 006 | 010 | 002 | 006 | 010 | 002 | 006 | 010
ALD Page (Outgate) RQ0xx | 062 | 066 | 070 | 062 | 066 | 070 | 062 | 066 | 070 | 062 | 066 | 070 | 062 | 066 | 070 | 062 | 066 | 070
B Diagram 0-056
C Diagram 0-056
E-reg Pre-bus | 00-07
Card Location 04D2xx | Q2
ALD Page RExxx | 901
E-register
Bits | P | 00-03 | 04-07 | P | 08-12 | 13-15
Card Location 04A-Dxxx | 2R2 | 2R2 | 2R2 | 2S2 | 2S2 | 2S2
ALD Page RExxx | 041 | 001 | 041 | 861 | 851 | 861
E-reg Adder | 08-15
Card Location 04D2xx | S2
ALD Page RExxx | 821-831
E-reg Adder Lth | 08-15
Card Location 04D2xx | S2
ALD Page RExxx | 821-831
A
B
Line Name | Card Location | ALD Page
Parity Error E00-07 | 04A-D2R2 | RE041
Ind E0-7 Check | 04A-D2T2 | RE891
E-unit to Microfiche 10 | Dot | WD215
Dsply Fische Bit 10 | 01A-B4K2 | WD059
Dsply Fische Bit 10 | | WM215
Dsply Fische Bit 10
Bus Bit 10 MF | 05B-A2J5 | PA011
Line 8 <> 10 | 05B-A1E4 | PB041
Ind | 06A-A1F4 | PB241
Line Name | Card Location | ALD Page
Ind E8-15 Check | 04A-D2T2 | RE891
E-unit to Microfiche 11 | Dot | WD203
Disply Fische Bit 11 | 01A-B4Q2 | WD077
Disply Fische Bit 11 | | WM215
Disply Fische Bit 11 | | WL100
Bus Bit 11 MF | 05B-A2J5 | PA201
Line 8 <> 11 | 06A-A1F6 | PB251
FRAME A5, ROW K
AP801 — AP811
PARALLEL ADDER HALF SUM (BYTE)
0 1 2 3 4 5 6 7
AP861
PA HALF SUM WORD
RE891
E REG
BYTE 0 BYTE 1
AS501
SER ADDER
HALF SUM
FULL SUM
XC149
TOD CLOCK CHECKS
BYTES
0&4 1&5 2&6 3
KA021
ENTER ERROR
KU131 KU231 KU131 KU221
CONTROL STORAGE
INVALID WORD P0-35 P36-71 P72-107
A — Line Name | Card Location | ALD Page — Ingate IQ Lths Bits 00-15 | 03B5H2 | KF911
B — Line Name | Card Location | ALD Page — Op Branch Taken | 04A-B4D2 | RJ111
C — Line Name | Card Location | ALD Page — Op | 04A4R2 | KR885
D — OR — Line Name | Card Location | ALD Page — Op | 04A4R2 | KR885 — Line Name | Card Location | ALD Page — Op Branch Taken | 04A-B4D2 | RJ111
E — Line Name | Card Location | ALD Page — Set L1 by WL | 04D2S2 | RE811 — W to L | 04A4R2 | KR885
F — Line Name | Card Location | ALD Page — Set L2 by WL | 04D2S2 | RE811 — W to L | 04A4R2 | KR885 — W04-07 to L2 | 04A4R2 | KR885
G — OR — Line Name | Card Location | ALD Page — Set L1 by L1L <> L2 by L2L | 04D2S2 | RE811 — Op Branch Taken | 04B4D2 | RJ111 — Update AC<>BC<>L<>MVBG<>MB | 04A4A2 | KR891 — L1 Incremented by 4 | 04A4R2 | KR885
I — H — Line Name | Card Location | ALD Page — Gate IQ to LL | 04D2S2 | RE811 — Op Branch Taken | 04B4D2 | RJ111
D
C

Table 1

Byte	Position								
	0	1	2	3	4	5	6	7	P
0	00	01	02	03	04	05	06	07	P
1	08	09	10	11	12	13	14	15	P
2	16	17	18	19	20	21	22	23	P
3	24	25	26	27	28	29	30	31	P
4	32	33	34	35	36	37	38	39	P
5	40	41	42	43	44	45	46	47	P
6	48	49	50	51	52	53	54	55	P
7	56	57	58	59	60	61	62	63	P
8	64	65	66	67	68	69	70	71	P
9	72	73	74	75	76	77	78	79	P
10	80	81	82	83	84	85	86	87	P
11	88	89	90	91	92	93	94	95	P
12	96	97	98	99	100	101	102	103	P
13	104	105	106	107	108	109	110	111	P
14	112	113	114	115	116	117	118	119	P
15	120	121	122	123	124	125	126	127	P

Note: This table shows the position within any byte that any bit of a quadword occupies. For example, bit 91 of a quadword is located in byte 11, bit 03, of that quadword.

Figure 10-10. E-Register Functional Packaging



ALD Page A-, B-, C-, D-regs = RA, RB, RC, RDxxx	005	035	065	085	115	145	165	195	225	245	275	305	325	355	385	405	435	465	485	515	545	565	595	625

	WA Bus Bits								
	00	01	02	03	04	05	06	07	P
Card Location 04A-C3xx	S2 P2	S2 P2	S2 P2	R2 N2	R2 N2	R2 N2	Q2 M2	Q2 M2	Q2 M2
ALD Page AS0xx	01 05	11 15	21 25	31 35	41 45	51 55	61 65	71 75	81 85

WB Bus Bits								
00	01	02	03	04	05	06	07	P
S2 P2	S2 P2	S2 P2	R2 N2	R2 N2	R2 N2	Q2 M2	Q2 M2	Q2 M2
01 05	11 15	21 25	31 35	41 45	51 55	61 65	71 75	81 85

Bit	00	01	02	03	04	05	06	07	00	01	02	03	04	05	06	07
Card Location	04A-C3J2								04A-C3K2							
ALD Page ASxxx	101	101	111	111	121	121	131	131	201				211			

Bit		0	1	2	3	4	5	6	7	P
Card Location	04A-C3xx	G2	G2	G2	G2	H2	H2	H2	H2	F2
ALD Page	AS3xx	01	11	21	31	41	51	61	71	511

Line Name	Card Location	ALD Page
OE HS Bits	04A-C3H2	AS431

Line Name	Card Location	ALD Page
OE WA Par<>P Adj	04A-C3D2	AS611
OE WB Par<>P Adj	04A-C3K2	AS701

Line Name	Card Location	ALD Page
Ind HS Check Tgr	04C3F2	AS501
E-unit to Microfiche 12	Dot	WD203
Dsply Fiche Bit 12	01B4M2	WD067
Dsply Fiche Bit 12	Exit 01	WM215
Dsply Fiche Bit 12	Entr 05	WL100
Bus Bit 12 MF	05B-A2J5	PA021
Line 8<>12	05B-A1G4	PB051
Ind	06A-A1F6	PB251

Line Name	Card Location	ALD Page
Ind FS Check Tgr	04C3F2	AS501
Dsply Fiche Bit 13	01B1Q2	WD037
Dsply Fiche Bit 13	Exit 01	WM215
Dsply Fiche Bit 13	Entr 05	WL100
Bus Bit 13 MF	05B-A2J5	PAU21
Line 8<>13	05B-A1D5	PB061
Ind	06A-A1G2	PB261

Figure 10-11. Serial Adder Checks

SCU TASKS

The tasks assigned to SCU are divided into seven main variations.

1. CPU fetch - data not in buffer
2. CPU fetch - data in buffer
3. CPU store - data not in buffer
4. CPU store - data in buffer
5. Channel fetch
6. Channel store - data not in buffer
7. Channel store - data in buffer

Each of the above seven main tasks starts with a request to SCU controls. The requesting unit starts the SCU by sending its request signal and by placing an address on the SCU address bus. Requests can come to SCU from either CPU or channels and can arrive at SCU in any sequence or any number of simultaneous requests.

CPU requests come from the I-unit address registers - IARA, IARB, source and destination. Each of these registers has its own request line to SCU which turns on a corresponding request latch. These latches in SCU start the sequence of operation and generate an identifying code so that SCU can know during the processing of a request where it came from.

Channel requests are identified in a similar way, but because of the speed requirements of the faster I/O devices, channels have the highest priority and channel requests are handled by SCU slightly differently from CPU requests.

Four of the seven SCU tasks are necessary to fetch and store from the two users of main storage, CPU and channel. The use of the high-speed buffer makes the additional three tasks necessary. Because of the way the high-speed buffer is used, CPU requests always check the buffer to see if the addressed main storage location is in the buffer, but the buffer is checked only on a store for channel requests.

Figure 11-1. SCU Tasks

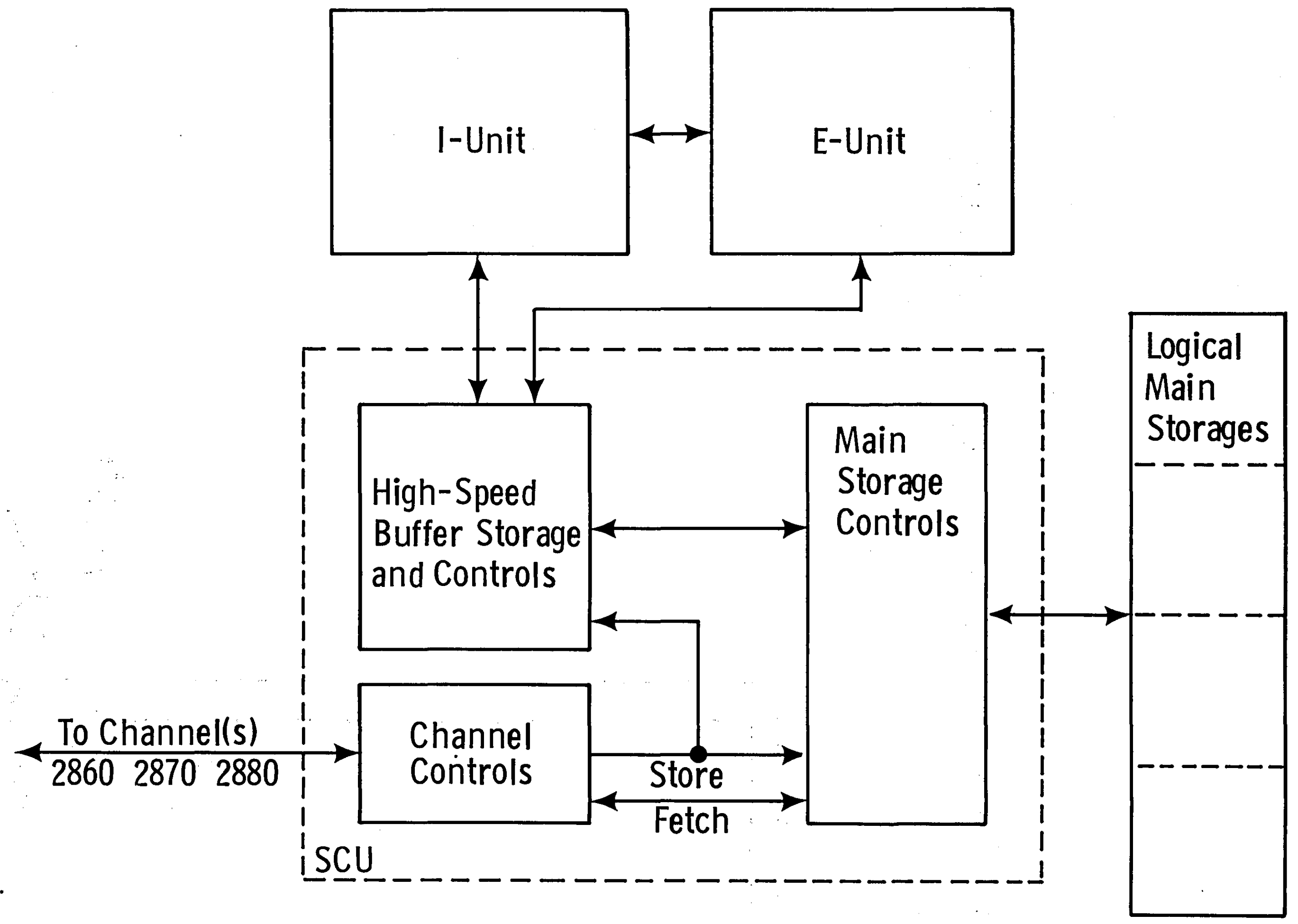


Figure 11-2. Storage Control Unit (SCU)

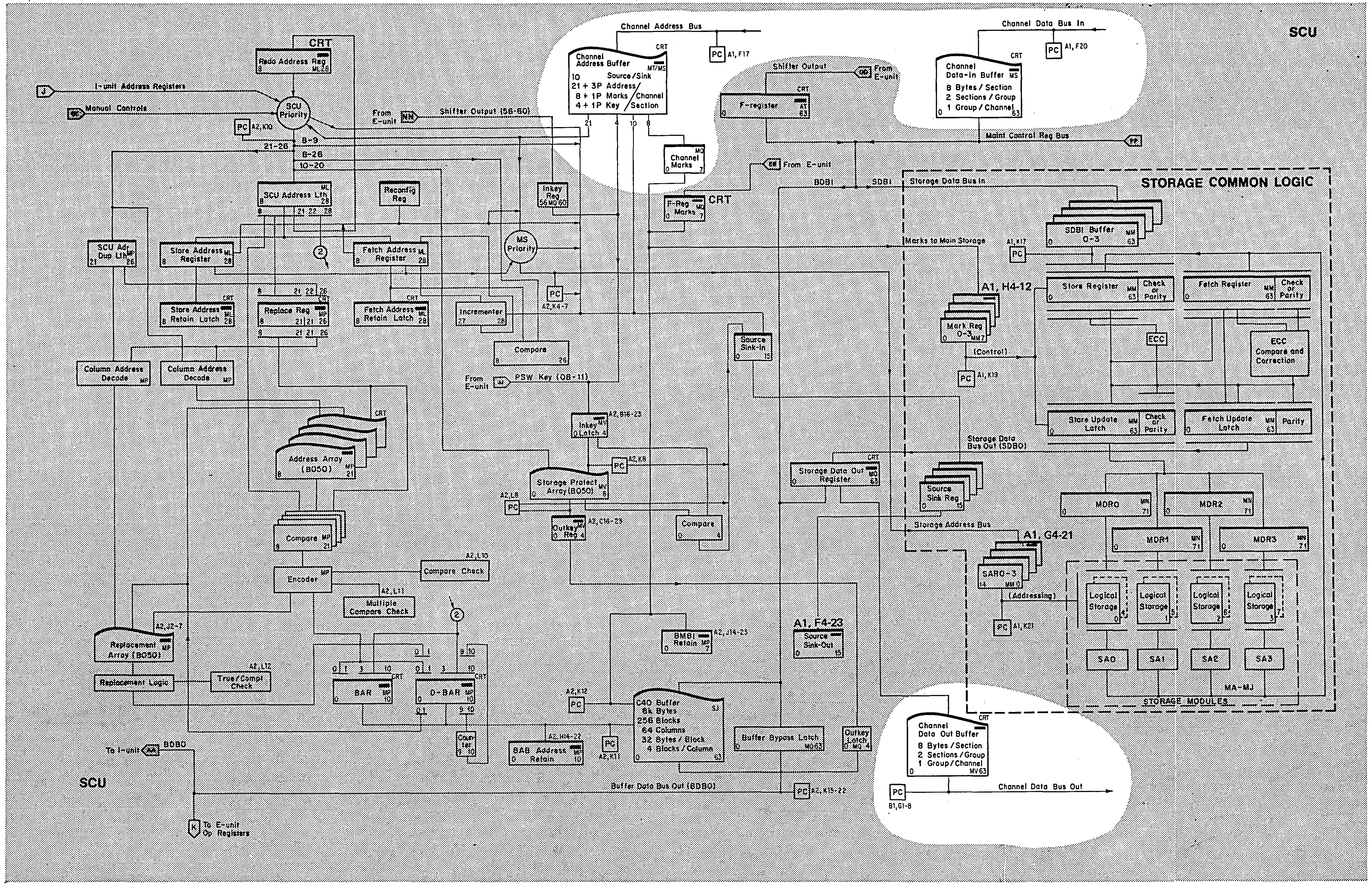


Figure 11-3. Channel Control Area

HIGH-SPEED BUFFER CONTROL AREA

The storage control unit (SCU) contains the high-speed buffer and controls all buffer and main storage references made by the CPU, the channels, and manual controls. The buffer storage control portion of the SCU handles CPU to main storage references, both fetches and stores. Parity checking is used for data verification in the buffer.

When a data fetch request is made by the CPU, buffer storage control determines whether or not the requested data is in the high-speed buffer in interrogating its address array of the buffer's contents. If the data requested is present in the buffer, it is sent directly to the CPU without a main storage reference. If the requested data is not currently in the buffer, a main storage fetch is made. The data obtained is sent to the CPU. The data is also assigned a buffer location and stored in the buffer. When data is stored by the CPU, both the buffer and main storage are updated if the main storage location being altered is one whose contents are currently being maintained in the buffer.

The channels never access the buffer directly. They read into and write from main storage only. When a channel stores data in main storage, the address array is interrogated. If data from the affected main storage address is being maintained in the buffer, the data is placed in the buffer as well as in main storage.

The entire buffer can be disabled manually by a system console switch or via execution of a DIAGNOSE instruction. When the buffer is disabled, all CPU fetches are made directly to main storage and effective system execution speed is reduced.

Figure 11-4. High-Speed Buffer Control Area

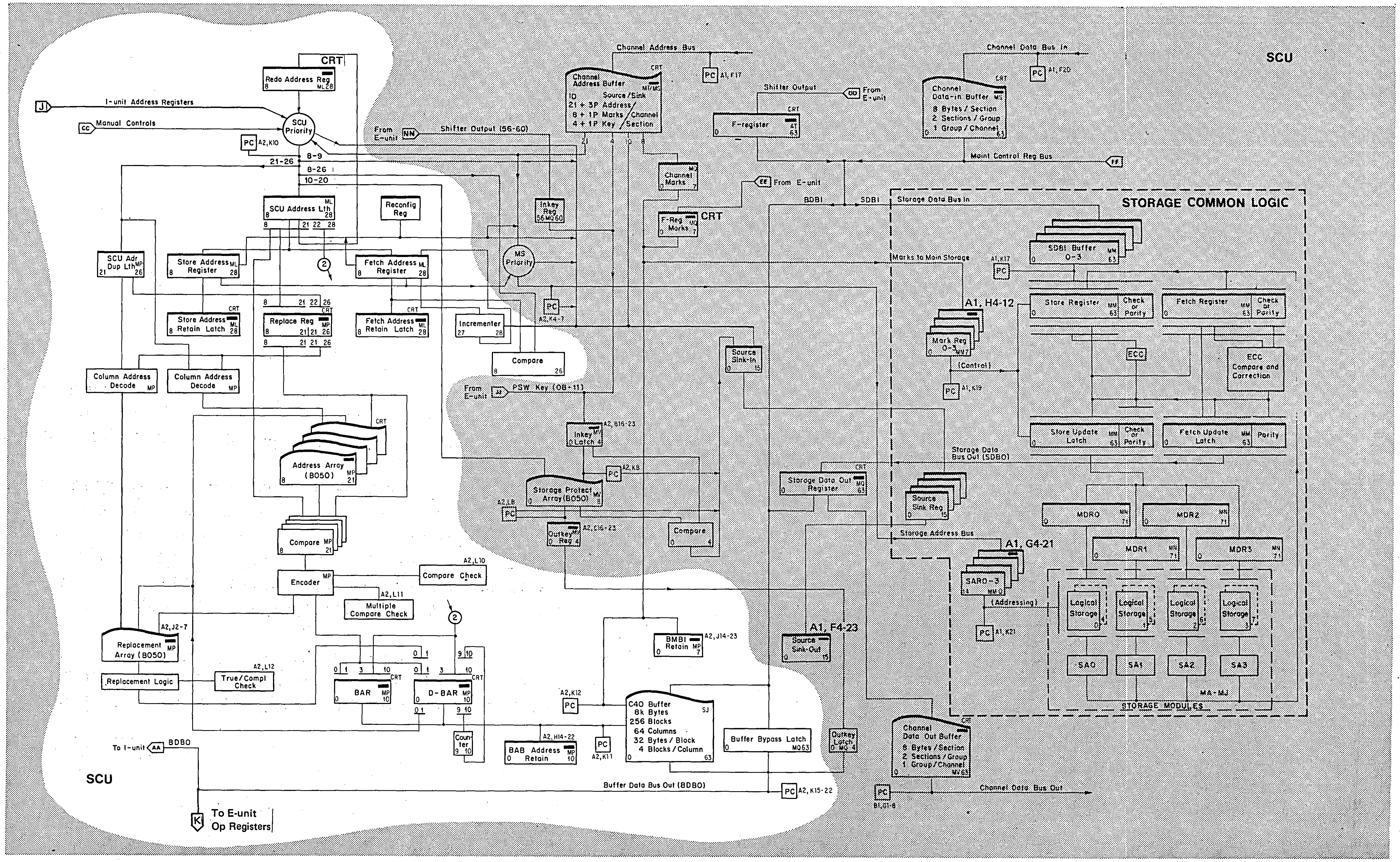


Figure 11-5. High-Speed Buffer Control Area.

MAIN STORAGE CONTROLS

All control signals and data pass through the main storage control section of SCU. Circuits to assign priority between requests for channel and the various parts of CPU are located in this section of SCU.

The main storage control section receives requests from the high-speed buffer controls or the channel controls. The request is then processed first through the priority circuits. Then a select is sent to main storage along with an address, source-sink bits, and data and marks if a store. At advance time, main storage controls receive the source-sink bits as well as the data for a fetch and sends the information to its proper destination.

The main storage control section also has the circuits that permit pluggable control in interleaving addresses. Under control from the console, serial, two-way, or four-way interleaving is possible.

Figure 11-6. Main Storage Controls

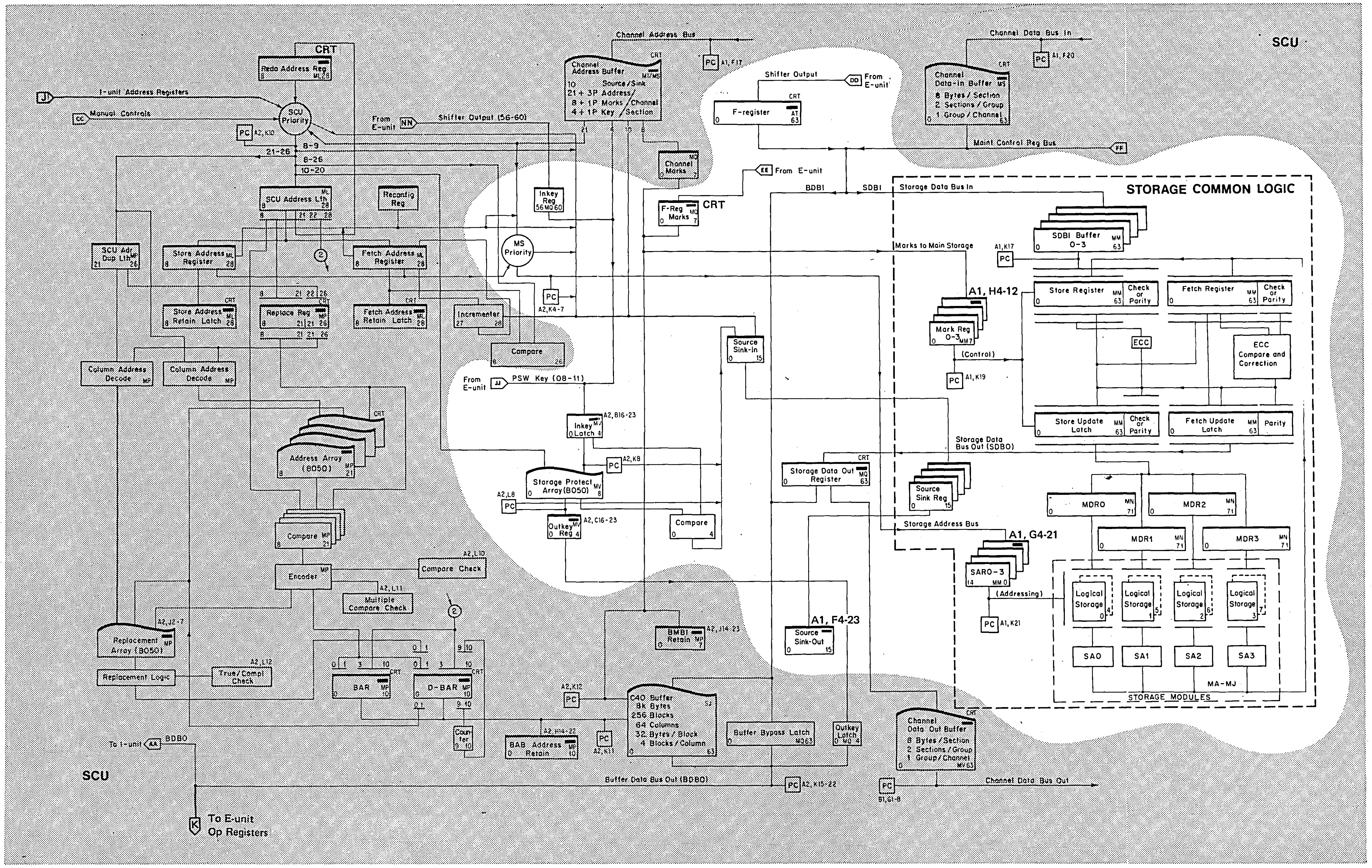


Figure 11-7. Main Storage Control Area

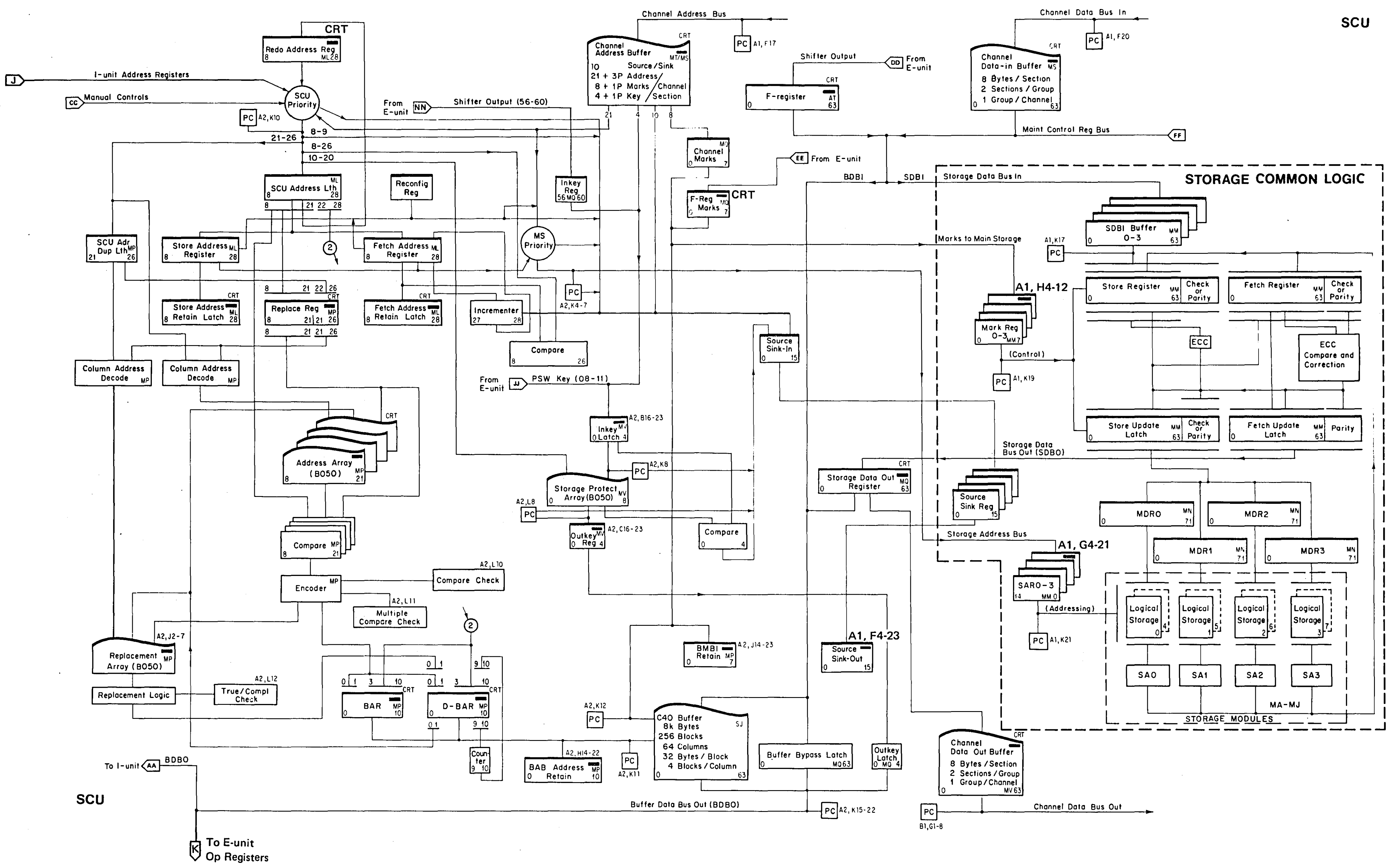


Figure 11-8. SCU Data Flow

CHANNEL BUFFERS

The basic machine includes three channel buffers which are divided into fourteen sections; two sections per channel. The sections are numbered 1-0, 1-1, 2-0, 2-1, etc., 7-0, and 7-1. When the extended channel feature is added, the buffers are divided into 24 sections numbered from 1-0 through D-1 but skipping 8-0 and 8-1. The three channel buffers are channel address-in, channel data-in, and channel data-out. The channel address in buffer has 48 data bits which are organized as follows: ten bits for source-sink information, 24 bits for address, nine bits for store marks, and five bits for protect key. Each of the data buffers include 72 bits for a doubleword of data.

Although two sections are provided for each channel, the only channel that makes use of both sections is the 2880. The other channels will always be using the first section or x-0 (x = the channel buffer group assigned to the channel). Channel buffer sections are assigned to the channels by group numbers. The group numbers are not necessarily synonymous with the digit of the channel address.

Figure 11-9. Channel Buffers

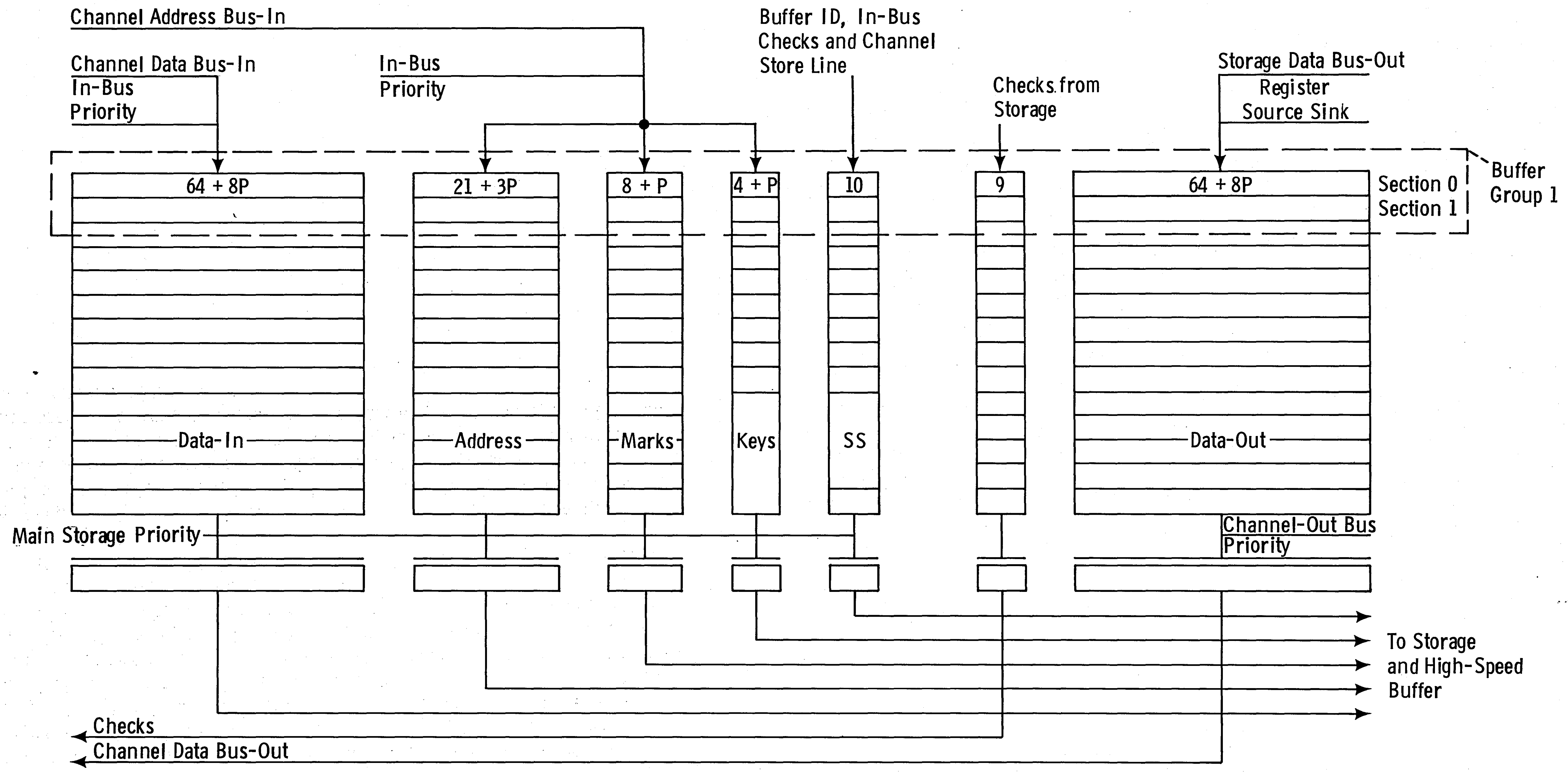


Figure 11-10. Channel Buffers

The purpose of the high-speed buffer is to speed up CPU request for data. The buffer produces the effect of a faster main storage by storing selected sections of main storage that are likely to be used next by the CPU. When the required information is in the buffer a request for data can be filled quickly. This leaves the times that CPU must wait for the longer main storage access time to the requests that do not find the needed data in the buffer. The buffer is constructed and used in such a way that better than 90% of the time, the buffer will contain the requested data.

The overall effect of the buffer and the way it is used is to make main storage appear to have an 80-nsec cycle time instead of its actual two usec cycle time. To use the buffer, the SCU processes all request from the CPU for data by first checking to see if the buffer has the requested data. If the buffer has the data, the request is filled by cycling the buffer and sending the data to the requesting section of the CPU.

If the buffer does not have the requested data, a fetch is made to main storage and the CPU must wait the full main storage access time.

When the SCU receives a request to store data into main storage from the CPU, the buffer again is checked to see if the addressed location is in the buffer. If the buffer contains the addressed location on a CPU store, both the buffer and main storage are stored into. If the buffer does not have the addressed location, the store is made to main storage only.

A fetch request for main storage data from a channel does not involve the buffer. Main storage is addressed and the data is sent to the requesting channel. When a channel wants to store informatoin into main storage, the buffer is checked to see if the store location is in the buffer. If the store location is in the buffer, SCU stores the information into both the buffer and main storage.

Figure 11-11.

ERROR CORRECTION

- Error correction detects and corrects single-bit errors.
- Error correction uses four data registers.
- Error correction converts parity bits to ECC bits.

Each main storage access moves a doubleword (72 bits--64 data and eight error correction) through storage common logic. The error correction bits provide a means of automatically detecting and correcting single-bit errors. However, multiple-bit errors (though detected) are uncorrectable and must be stored as is. Either type of error is indicated to allow the CPU to determine any necessary action.

Before considering the method used to generate error correction bits, consider parity checking. In parity checking (used when ECC is disabled), one position of a byte is reserved for a parity bit. The parity-check logic sets this bit to a 1 or 0, as required, to make the total number of 1s contained in the byte an odd number (odd parity). If any bit position changes status during the transmission of data, the byte parity becomes even and the parity-check logic detects the error. No method of correction is provided by parity checking, and even multiples of errors in a byte will go undetected.

The error correction bits constitute a special type of parity checking. In a normal parity scheme, as described above, each data bit contributes to the status of the associated parity bit. Note that each data bit exhibits an equal amount of influence on the status of the parity bit. With error correction code (ECC) logic, the parity bits are replaced by a group of seven bits which record a parity sum. Each data bit contributes to this sum. No two data bits contribute the same amount; however, the amount contributed by any one bit is equivalent to the position of that bit in the doubleword. Thus, bit position 1 contributes to ECC bit 1, position 2 contributes to ECC bit 2, and position 3 contributes to both ECC bits 1 and 2. Additionally, bits 0-32 contribute to ECC bit 0, and bit 0 also contributes to ECC bit 32.

Eight check bits are identified as C0, C1, C2, C4, C8, C16, C32, and Ct. The last check bit, Ct, acts as a parity bit for the other check bits. Odd parity is always maintained.

Figure 11-12-1.

When data is stored, the data bits in the store register are used to generate the stored ECC check bits. When data is fetched, the data bits in the fetch register are used to generate new ECC check bits. These new ECC check bits are then compared with the check bits in the fetch register that had been stored and were fetched with the data bits. An equal comparison of the fetched check bits and the newly generated check bits indicates no error. An unequal comparison indicates an error, and bits known as syndromes are generated from the two sets of check bits.

The syndrome bits, identified as S0, S1, S2, S4, S8, S16, S32, and St are decoded to indicate and correct the changed bit (even a change of the check bit) if only one bit has changed. If two or more bits have changed the syndrome bits decode and signal an uncorrectable error.

The error correction circuit uses four registers to hold data before, during, and after error detection and correction. Two registers (RS and RF) hold the data before and during detection and two registers (SUL and FUL) hold the data after detection and correction. The correction circuit always operates regardless of error occurrance. Error correction circuits also prepare new ECC bits for store operations.

The four ECC registers, RS, SUL, RF, and FUL, are identical; each consists of 64 data bits and eight parity or check bits for the left half and 64 data bits and eight parity or check bits for the right half.

STORE REGISTER

The store register (RS) receives data from one of two sources, the storage data buffer-in registers for new data or the storage bus-out (SBO) from the LSU's.

Gating into RS is on a byte basis under mark bit or "cancel" signal control. Each mark bit is associated with a byte; if the mark bit is set, the byte is transferred from the buffer register, and, if the mark bit is clear, the byte is transferred from the LSU. The "cancel" signal causes all bytes to be transferred from the LSU.

The setting of RS occurs approximately 1200 nsec into the storage cycle.

The ECC logic generates check bits from the data bits in RS; the check bits, as well as the data bits, are available for transfer to SUL.

Figure 11-12-2.

STORE UPDATE LATCHES

The store update latches (SUL) register receives data from one of two places: RS (during a normal fetch, store, or TAS cycle) or from RF if an uncorrectable error has occurred.

If, during a fetch cycle or a partial store cycle, a correctable had been detected, during the transfer from RS to SUL, the faulty bit and the associated ECC bit would have been corrected. However, if a correctable error had occured in a marked byte, the correction would have been blocked since the marked byte contains new data.

The transfer from RF to SUL occurs if an uncorrectable error occurs; this transfer overrides the transfer from RS since the data is to be regenerated exactly as it was received from the LSU.

The data in SUL is made immediately available to the applicable LSU.

FETCH REGISTER

The fetch register (RF) unconditionally receives data from the LSU at about 1200 nsec into the cycle.

The ECC logic generates new check bits from the data bits in RF. These new check bits are compared with the check bits that are received from the LSU (and stored in RF with the data bits). If an equal comparison is found (indicating no error), the data bits in RF are transferred to FUL and the check bits are replaced with newly generated parity bits. If a correctable error had been detected, the failing bit and associated parity bit would have been corrected during the transfer from RF to FUL. Also, if a fetch or a partial store cycle had been in process, the failing bit and the associated check bit would have been corrected during the transfer from RS to SUL.

If an uncorrectable error occurs, the data and the check bits in RF are transferred to SUL, bypassing the ECC logic associated with RS.

FETCH UPDATE LATCHES

The fetch update latches (FUL) register always receives data from RF. The content of FUL is gated to the CPU about 20 nsec after the setting of FUL.

Figure 11-12-3.

CHANNEL BUFFERING

Each channel has a fixed assignment of one buffer group and each group consists of two sections. A buffer section stores all the information for one main storage access. Each buffer section stores data in, data out, marks, key, address, and various check and identification information.

Channel requests for input or output information must first vie for priority on the channel-in bus. The order of in-bus priority is established by inserting jumpers on a matrix card located in Frame 03. The wiring of the jumpers for a particular system results in a fixed-buffer assignment to each channel which is displayed on the microfiche viewer.

When in-bus priority is resolved, the appropriate channel is signalled. The channel, in turn, responds with address, marks, key and, if a store operation, a doubleword of data and a store signal. Upon the receipt of an address valid signal, this information is loaded into one of the channel's assigned buffer sections. If both buffer sections are empty (X 0 and X 1 where X is the buffer group number), the channel transfer is loaded into section X0. If a buffer section is not available for use (empty), that channel is blocked from receiving in-bus priority.

Once a buffer section is loaded, it is set to a busy state and entered as a contender into main storage priority. After main storage priority has been received, the storage unit is selected. Identification of the channel buffer section is placed on the source-sink bus-in, and the buffer section contents are gated to main storage. Main storage control receives the address, marks, and key. In the case of a store, the buffer section contents are also gated to the high-speed buffer control section to update the buffer if the address block is in the buffer.

When the channel request processing is completed by main storage, a storage signal followed by data, source-sink identification, and error information is sent to the main storage control which assembles and sends the information to the initiating channel buffer section.

Figure 11-13-1.

When the information from storage is loaded into a channel buffer section, use of the channel out-bus is vied for by entering channel-out bus priority. When priority is resolved in favor of a buffer section the contents of that buffer section are gated onto the channel-out bus. At this point that buffer section is made not busy. All address and data type checks, associated with this channel request, are sent to the channel at the same time as the channel buffer data.

Figure 11-13-2.

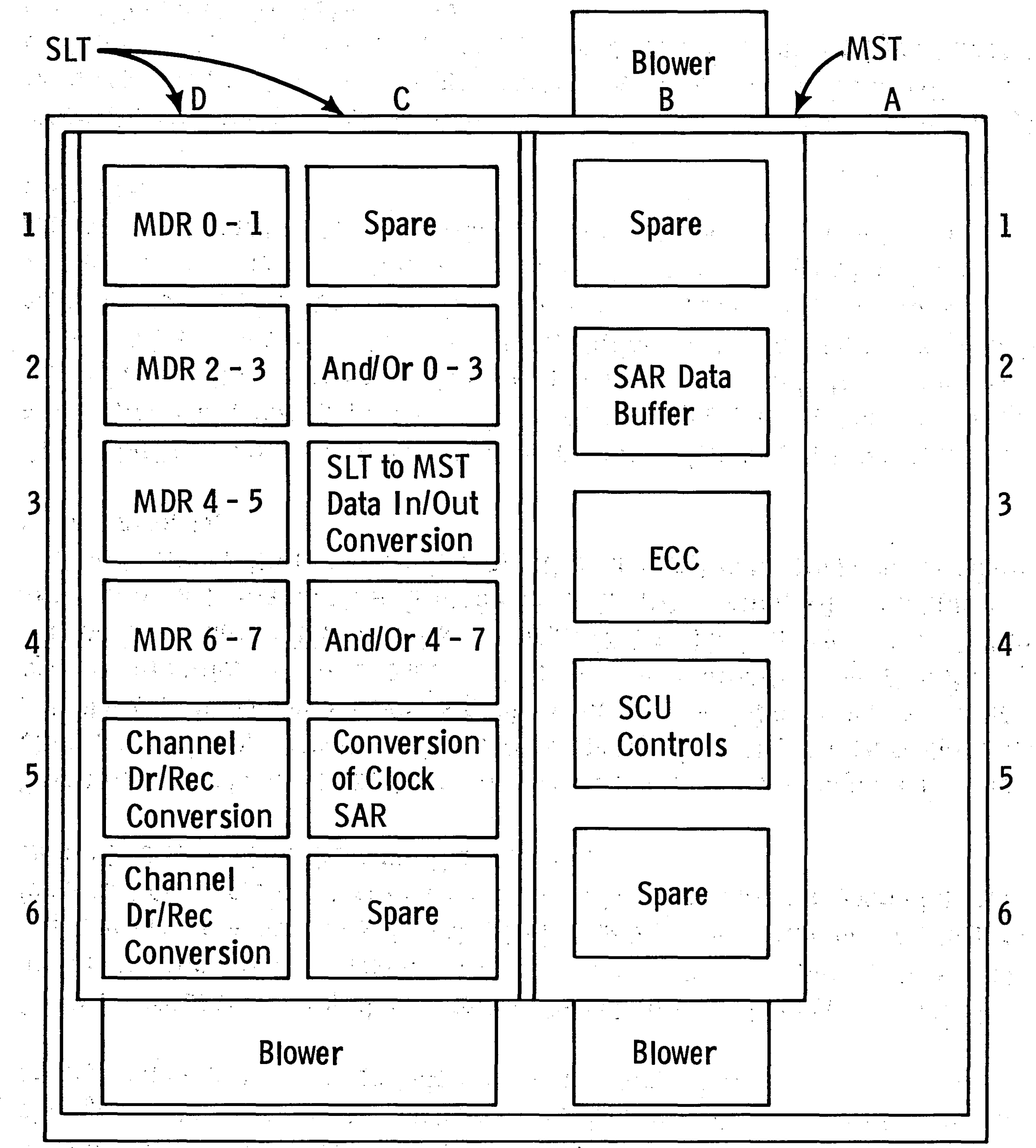


Figure 11-14. CPU Frame 02

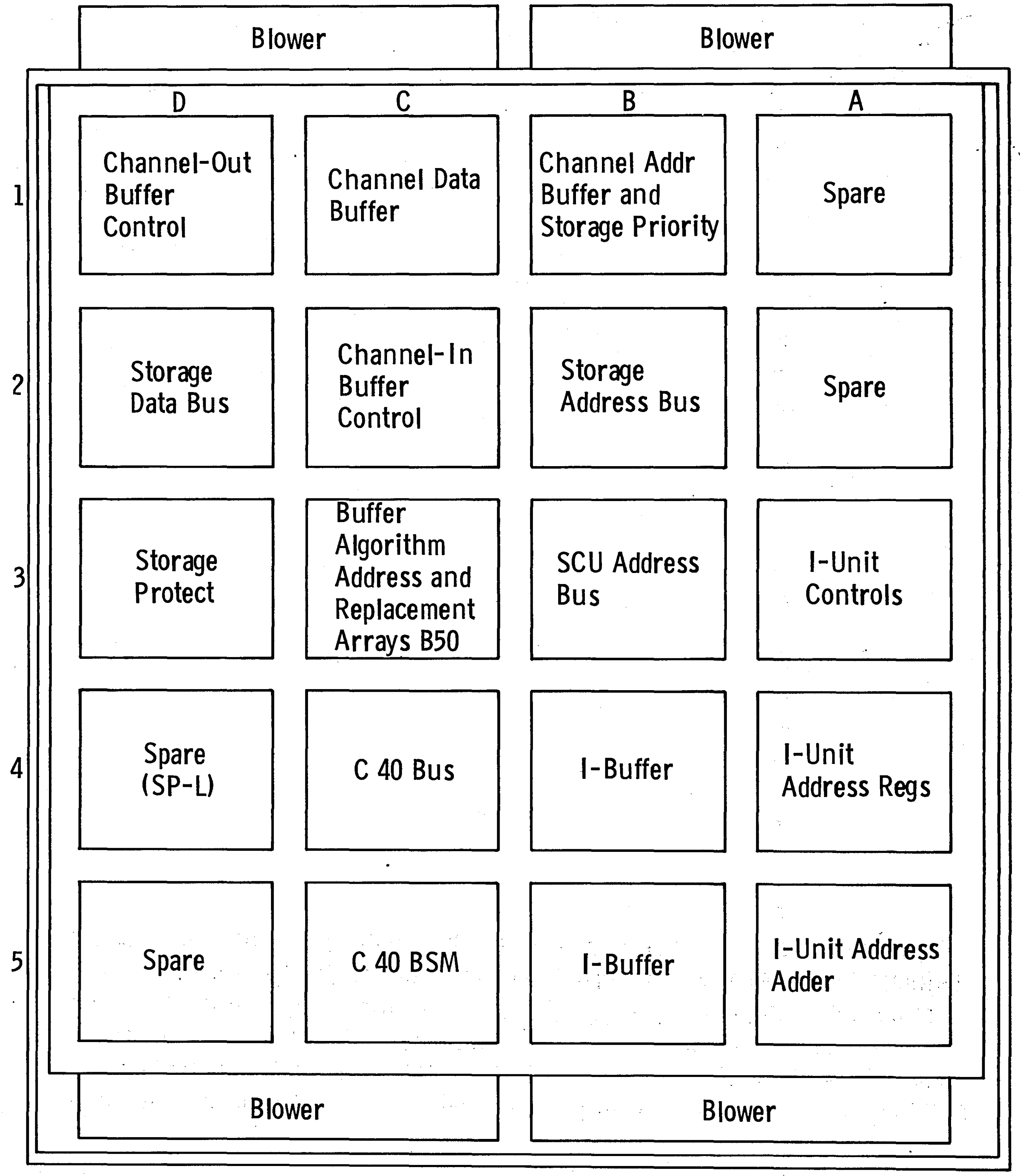


Figure 11-15. CPU Frame 03

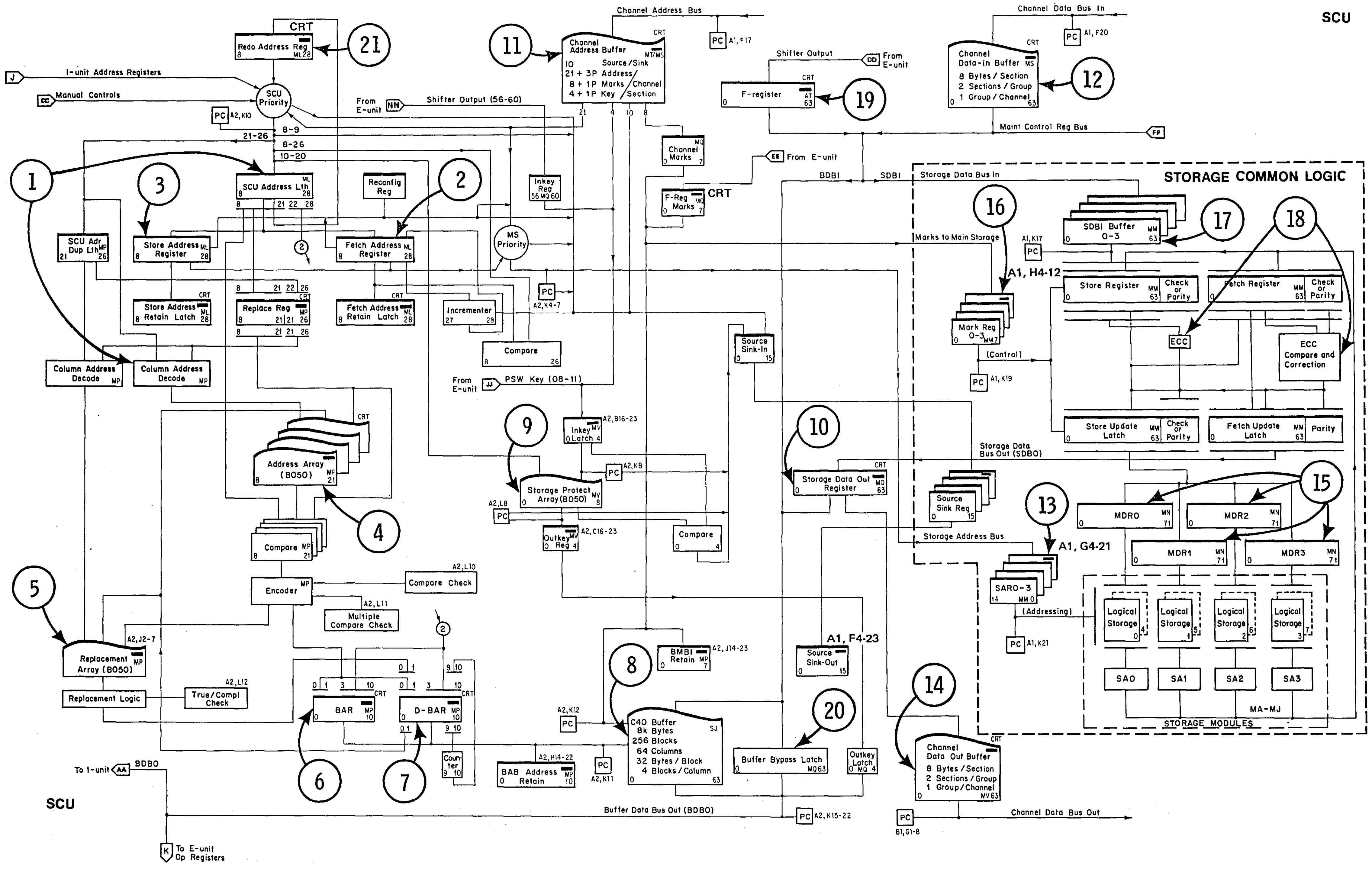


Figure 12-1. SCU Data Flow

CPU FETCH - DATA NOT IN BUFFER

All SCU operations begin when the originator of a request to either store or fetch data presents an address and sends a request signal. The address arrives at SCU at the input to the SCU address latches. A simplified data flow is shown on SCM 12-4. High speed buffer controls are not shown because the example chose does not find the data in the buffer. Note that the address input path to SCU enters by way of the priority circle. The circle in this figure represents the input control to SCU. All requests must go through the priority circuits to start an SCU operation.

Each part of the system that makes requests to SCU has its own request latch located in the SCU. When a request latch is turned on, SCU operation starts.

For an example, the SCU starts the fetch processing when the 1-unit turns on the source request latch. (Address in Source Request Latch). This latch makes an entry into the SCU priority circuit. If there are no higher priority requests waiting - destination, redo, nor any channel request - source is given priority. The I-unit can alter the priority sequence by sending a control line that can give instruction address register A or B preference over the source request.

Once the SCU grants priority to a request, the source request in this example, several things must take place. First, a test (MS CPU Adv) to see if some previous request is being filled on this cycle, because if data is returning from main storage the buffer would be in use. The "source address gate trigger" is turned on to gate the address from the source address register to the SCU address latches. The "accept" signal is sent to the I-unit to tell that the source register contents have been taken and that the source register can be made not busy.

Also, the granting of priority generates the identification bits that are passed through SCU to serve as control bits and at the end of the request handling to direct the data to the proper place. (Source Sink).

Figure 12-2-1.

Another important item started by source priority is the reference to the high speed buffer. Although for the chosen example the requested data is not in the buffer, all CPU requests unconditionally check the buffer. The address from the source register is compared against the addresses in the address array and a decision is made in time to stop the request to main storage if the addressed data is in the buffer.

After a test to see if the fetch address register (FAR) is busy, the address is moved from the SCU address latches to FAR. If FAR is found busy at this point, the address and the identification bits are moved to the redo register and the request starts over as a redo request into SCU priority.

When the address is moved into FAR the bits are decoded to see which logical storage unit (LSU) is needed. The busy latches for the LSU's are used to see if the addressed LSU is available and if so a request for main storage priority is made.

If the FAR Busy trigger is on, to signify that the addressed data was not in the buffer, and if no higher priority request is in process, FAR is granted main storage priority. If a higher priority, or a busy LSU blocks FAR main storage priority, the FAR request can wait until the blocking condition ends. The address would stay in FAR and the ID bits would stay in the FAR ID triggers with the FAR busy trigger on to block any following request to use FAR.

FAR main storage priority sends the select signal to the selected LSU and gates the address in FAR onto the storage address bus. Also timing gate "FAR Op B" is turned on to gate the ID bits into the source sink latches.

The timing controls to the selected main storage BSM's have various time delay circuits that are needed to synchronize the BSM to the SCU. The "gated select and ready LSU" line gates the SAB into the selected SAR and also starts the two "delay line clocks" in the selected pair of BSM's.

Figure 12-2-2.

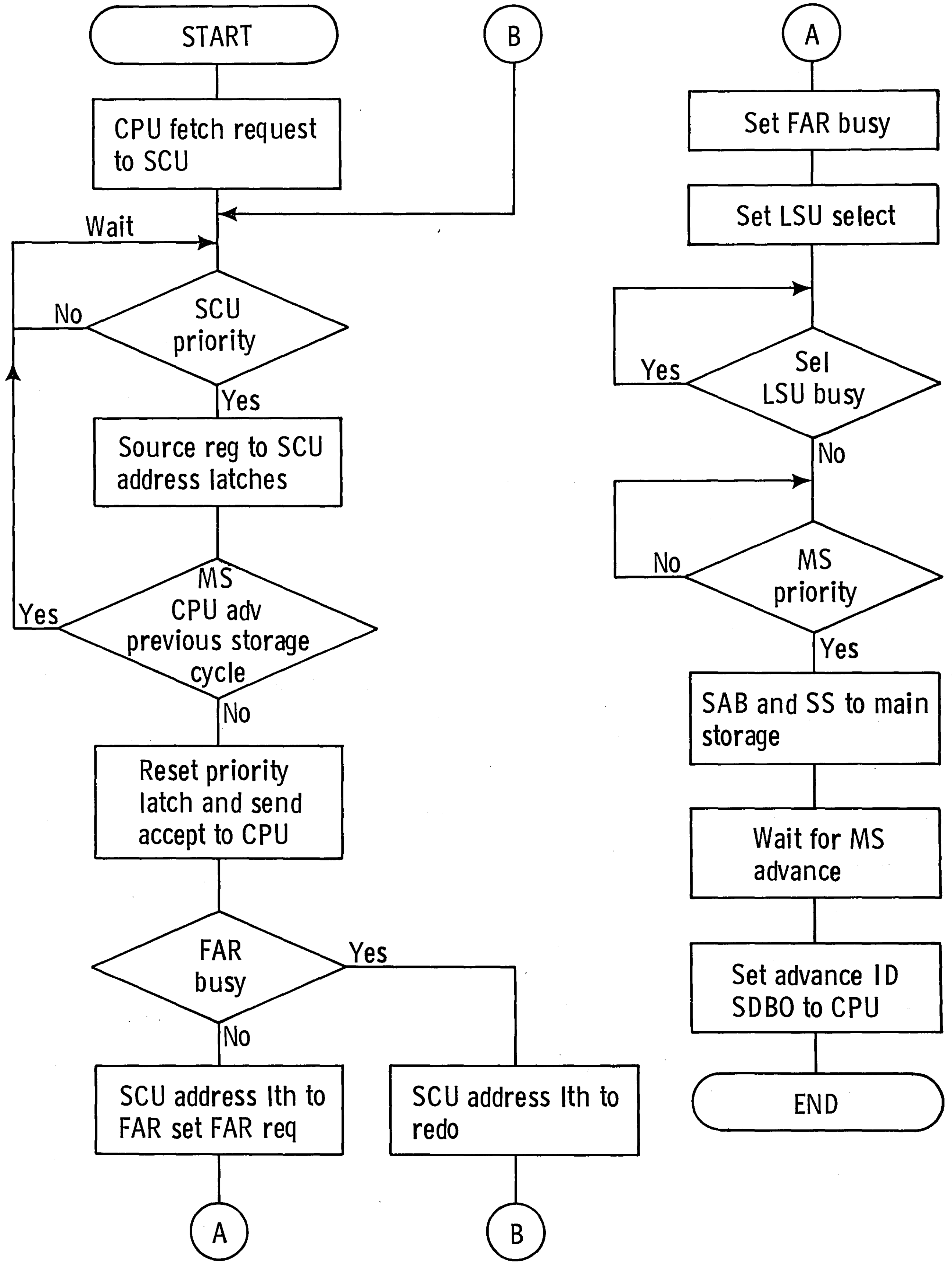
About one microsecond after the start of the BSM's "delay line clock", a busy signal from each BSM occurs. The upper BSM "busy" starts a delay line in the SCU. This delay line controls the flow of the data returning from the BSM pair through the ECC circuits and finally into the storage data out register. And also gates out the source sink ID bits to the CPU. The lower BSM "busy" is used to control restarting the BSM delay line for the write half of the BSM cycle.

The data will pass out of the SDBO register, through the Buffer bypass latch to the CPU. Where it is directied to the area it is needed by the CPU's decode of the ID bits it received.

At this time the SCU will load the buffer with the block of data associated with the previous fetch. But we will not get into that right now.

Timings for the important parts of the fetch operation through the SCU can be seen on SCM 12-5.

Figure 12-2-3.



START
CPU fetch request to SCU
B
Wait
No
SCU priority
Yes
Source reg to SCU address latches
MS CPU adv previous storage cycle
Yes
No
Reset priority latch and send accept to CPU
FAR busy
Yes
No
SCU address lth to FAR set FAR req
SCU address lth to redo
A
B
A
Set FAR busy
Set LSU select
Yes
Sel LSU busy
No
No
MS priority
Yes
SAB and SS to main storage
Wait for MS advance
Set advance ID SDBO to CPU
END

Figure 12-3. CPU Main Storage Fetch (Ignore Buffer Circuits)

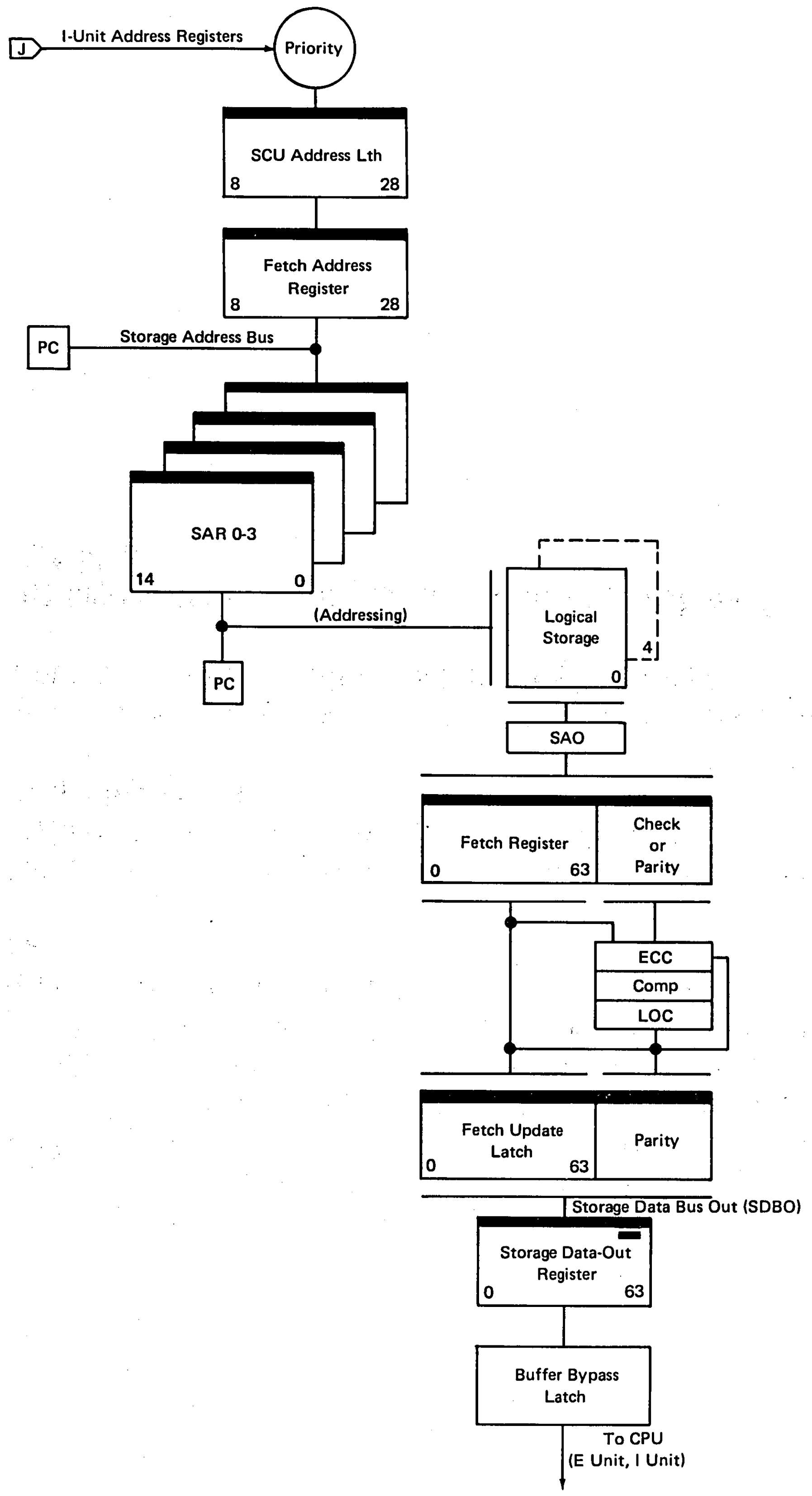


Figure 12-4. Simplified Data Flow - CPU Fetch

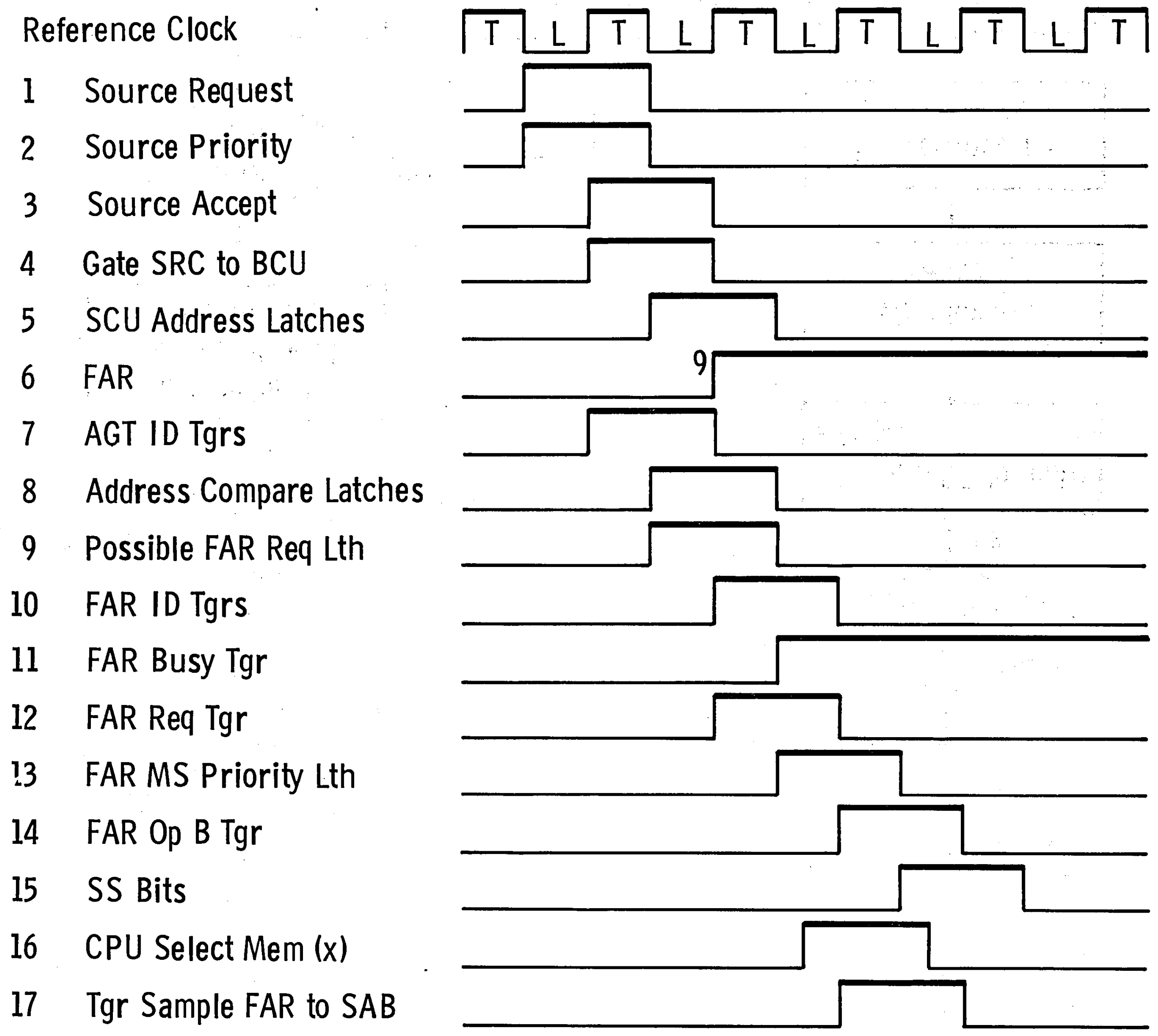


Figure 12-5. Source Fetch Timings

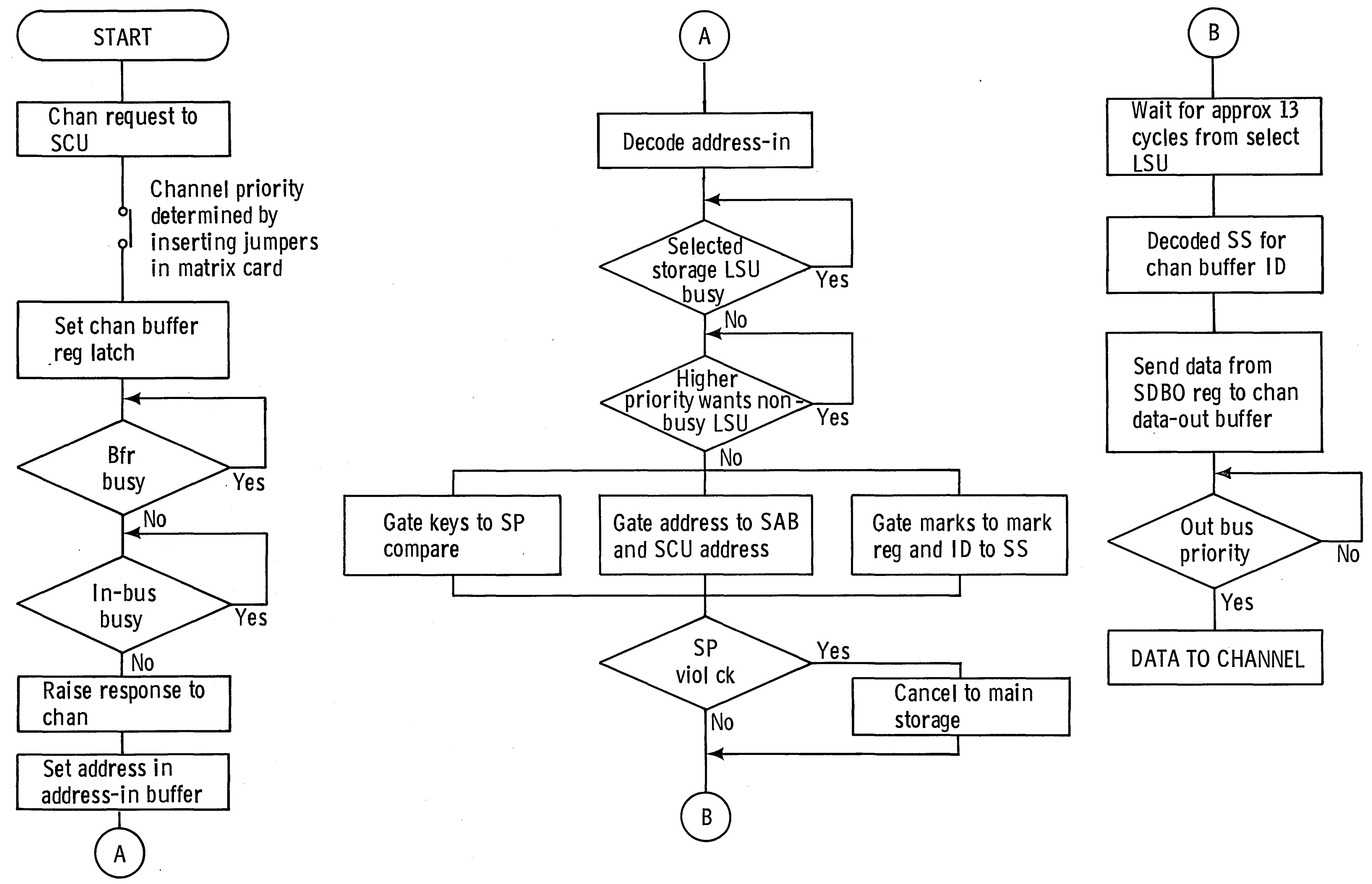

START
Chan request to SCU
Channel priority determined by inserting jumpers in matrix card
Set chan buffer reg latch
Bfr busy
Yes
No
In-bus busy
Yes
No
Raise response to chan
Set address in address-in buffer
A
A
Decode address-in
Selected storage LSU busy
Yes
No
Higher priority wants non-busy LSU
Yes
No
Gate keys to SP compare
Gate address to SAB and SCU address
Gate marks to mark reg and ID to SS
SP viol ck
Yes
Cancel to main storage
No
B
B
Wait for approx 13 cycles from select LSU
Decoded SS for chan buffer ID
Send data from SDBO reg to chan data-out buffer
Out bus priority
No
Yes
DATA TO CHANNEL

Figure 12-6. Channel Fetch

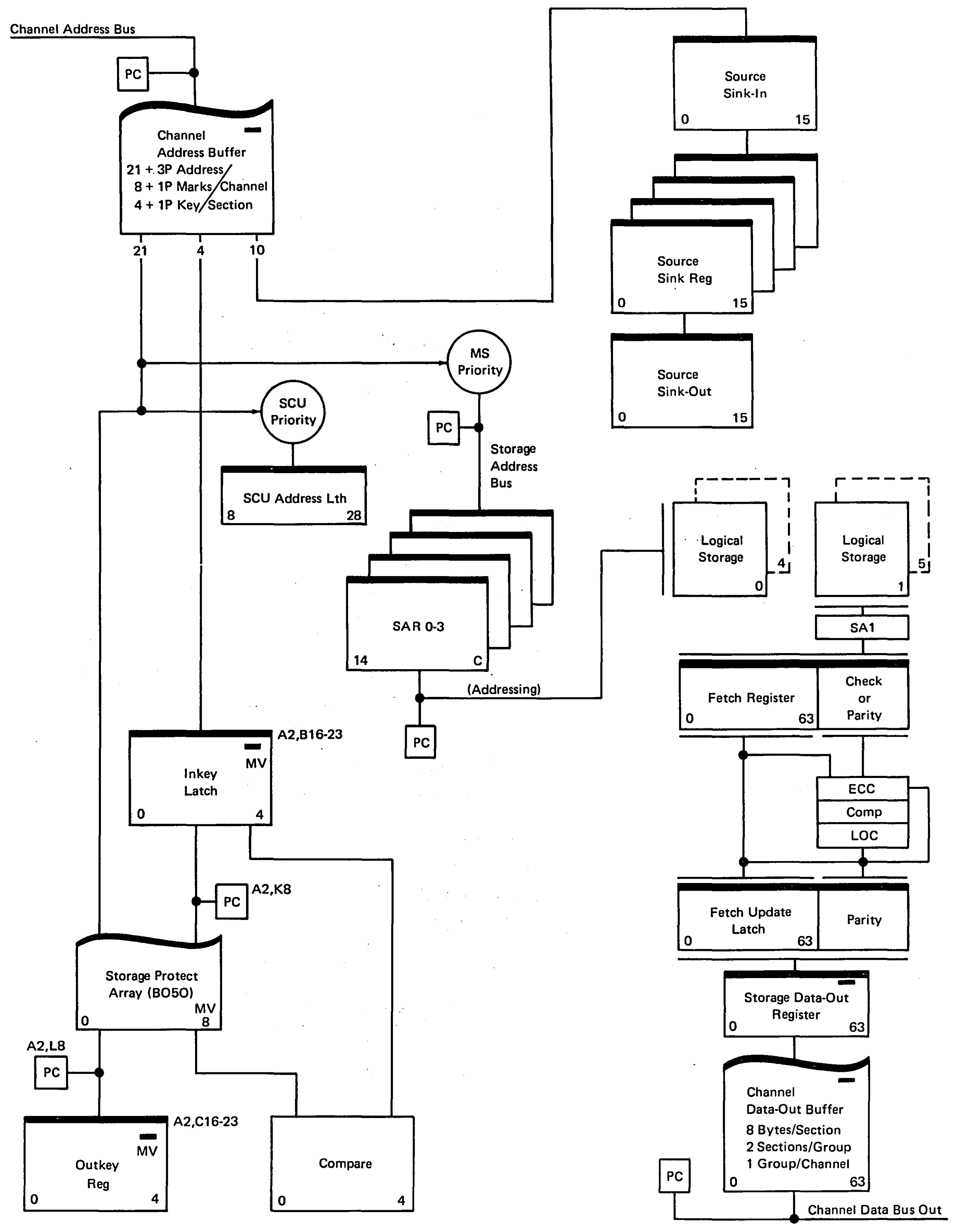


Figure 12-7. Simplified Data Flow - Channel Fetch

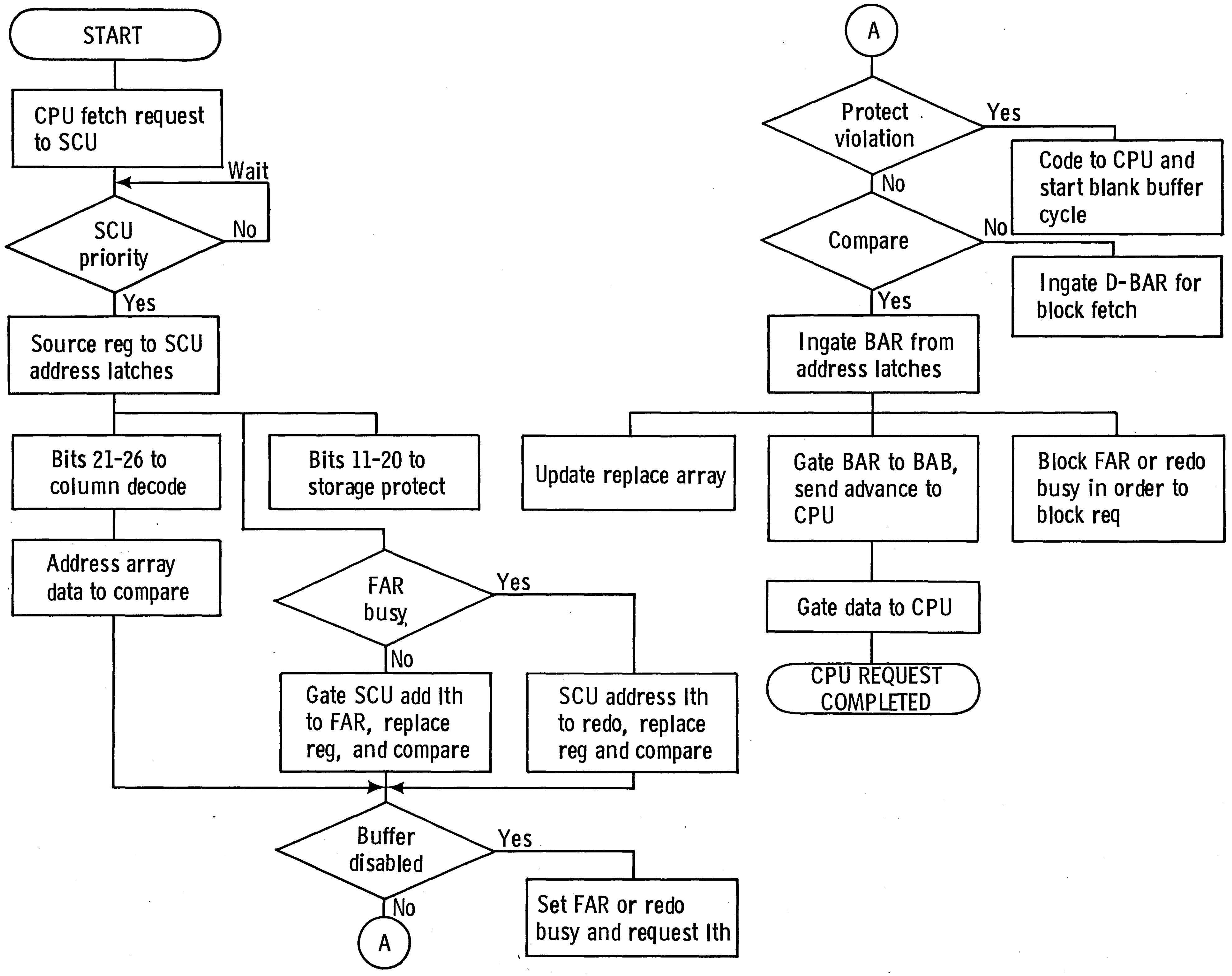
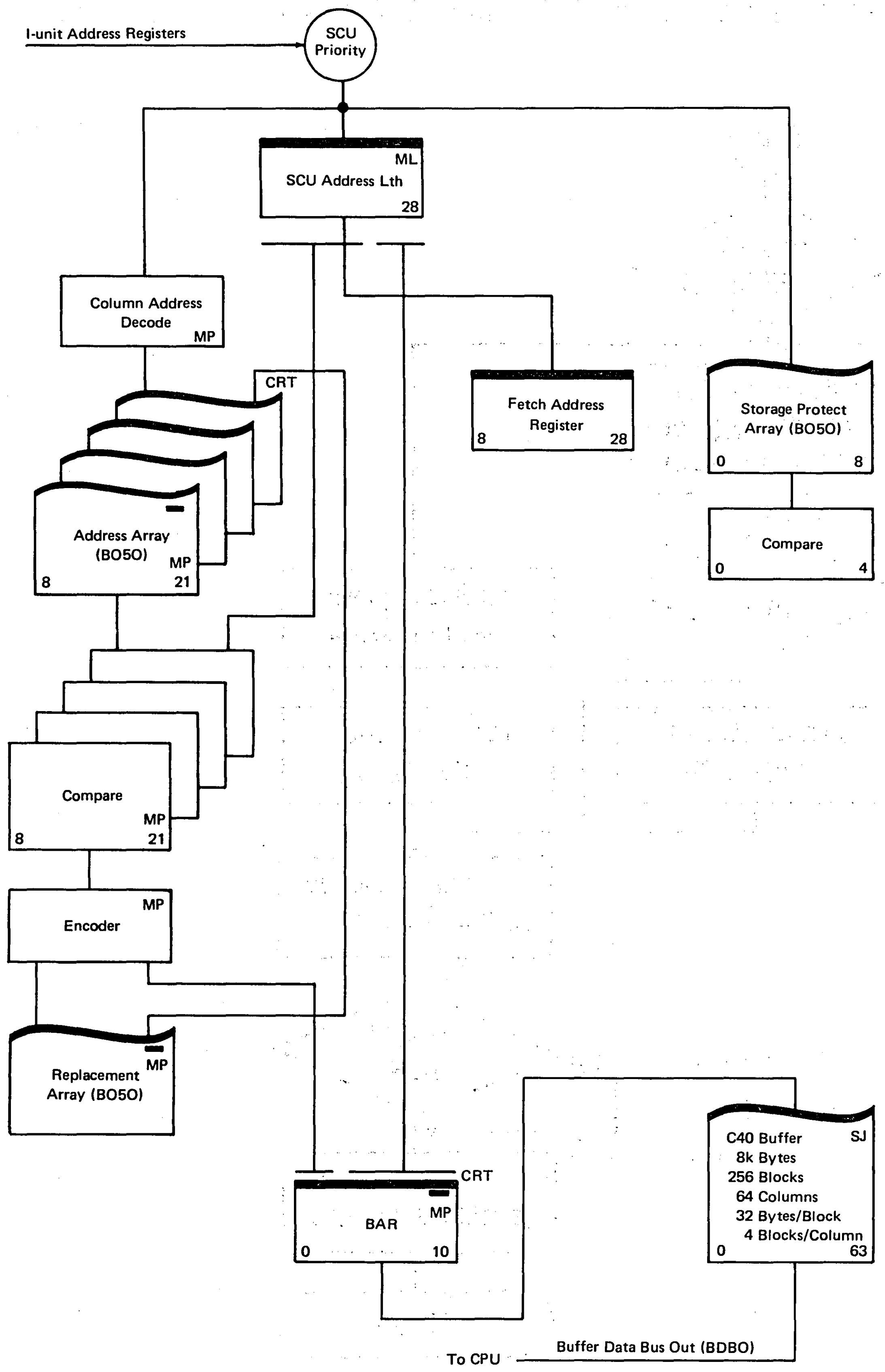

START
CPU fetch request to SCU
Wait
SCU priority
No
Yes
Source reg to SCU address latches
Bits 21-26 to column decode
Bits 11-20 to storage protect
Address array data to compare
FAR busy
Yes
No
Gate SCU add Ith to FAR, replace reg, and compare
SCU address Ith to redo, replace reg and compare
Buffer disabled
Yes
No
A
Set FAR or redo busy and request Ith
A
Protect violation
Yes
No
Code to CPU and start blank buffer cycle
Compare
No
Yes
Ingate D-BAR for block fetch
Ingate BAR from address latches
Update replace array
Gate BAR to BAB, send advance to CPU
Block FAR or redo busy in order to block req
Gate data to CPU
CPU REQUEST COMPLETED

Figure 12-8. CPU Fetch, Data In High-Speed Buffer



I-unit Address Registers
SCU Priority
SCU Address Lth ML 28
Column Address Decode MP
CRT
Fetch Address Register 8 28
Storage Protect Array (BO5O) 0 8
Compare 0 4
Address Array (BO5O) MP 8 21
Compare MP 8 21
Encoder MP
Replacement Array (BO5O) MP
BAR MP 0 10
CRT
C40 Buffer SJ 8k Bytes 256 Blocks 64 Columns 32 Bytes/Block 4 Blocks/Column 0 63
To CPU
Buffer Data Bus Out (BDBO)

Figure 12-9. Simplfied Data Flow - CPU Fetch, Data Found In HS Buffer

CPU STORE

All CPU stores are made from the destination register and except for channel or maintenance controls are under the control of the E-unit. The E-unit does a store by placing the data in the F-register and then sending the request signal to SCU. In most cases the I-unit will have already put the address in the destination register for the E-unit's use.

The address and request are sent to SCU. The priority circuits generate the ID bits for the source sink. Use the simplified flow chart, SCM 12-11 and simplified Data Flow SCM 12-12 to follow the store operation through the SCU.

As soon as the store gets SCU priority, the buffer is checked in the same way as the buffer was checked for a fetch.

With the address in the SCU address latches, accept is sent to CPU which allows the destination register to be made not busy. However, the data in the F-register must be held until the store vies for and gets main storage priority.

The address is next moved to the store address register and to the Buffer Address Register (BAR), (STAR) if STAR is not busy. STAR busy at this point would cause the request to start over as a redo request.

The address in STAR next causes a check to see which LSU is to be used and if the desired LSU is available (not busy). STAR then has to vie for main storage priority.

At this point the store could be blocked only by a channel request to some non busy LSU. When the store receives main storage priority, "STAR op time B" gates through the source sink-in bits into the desired LSU's source sink register and then turns on "STAR op time C" to gate the data to be stored from the F-register to the selected SDBI register. At the same time, BAR will address the high-speed buffer and F-register data will also be gated to the high-speed buffer.

Also the granting of main storage priority gates the address from STAR to the storage address bus and the select signal to the selected LSU. This gated select signal also gates the mark bus into the selected mark register.

Figure 12-10-1.

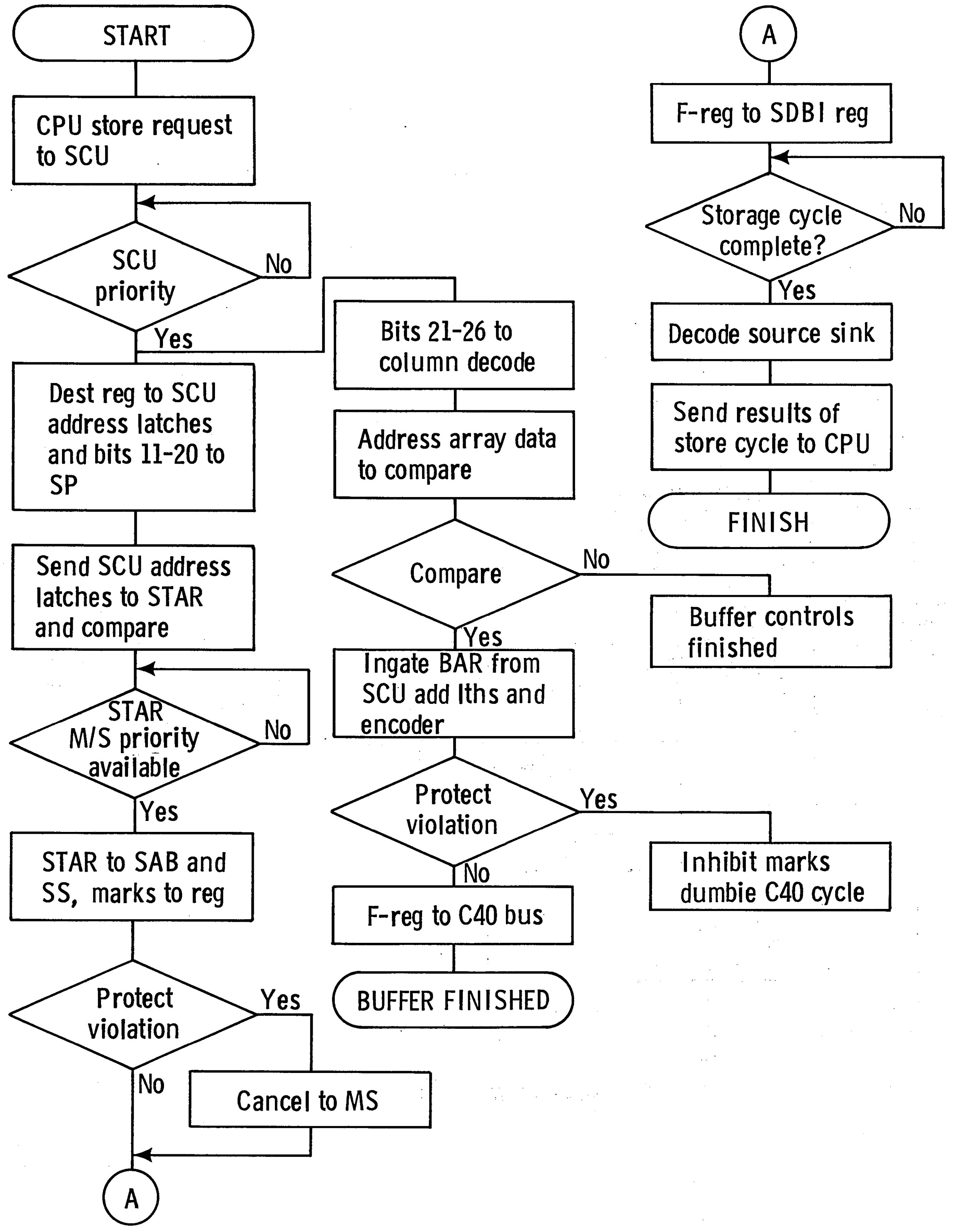

START
CPU store request to SCU
SCU priority
No
Yes
Dest reg to SCU address latches and bits 11-20 to SP
Send SCU address latches to STAR and compare
STAR M/S priority available
No
Yes
STAR to SAB and SS, marks to reg
Protect violation
Yes
No
Cancel to MS
A
Bits 21-26 to column decode
Address array data to compare
Compare
No
Yes
Buffer controls finished
Ingate BAR from SCU add lths and encoder
Protect violation
Yes
No
Inhibit marks dumbie C40 cycle
F-reg to C40 bus
BUFFER FINISHED
A
F-reg to SDBI reg
Storage cycle complete?
No
Yes
Decode source sink
Send results of store cycle to CPU
FINISH

Figure 12-11. CPU Store (Address In High-Speed Buffer)

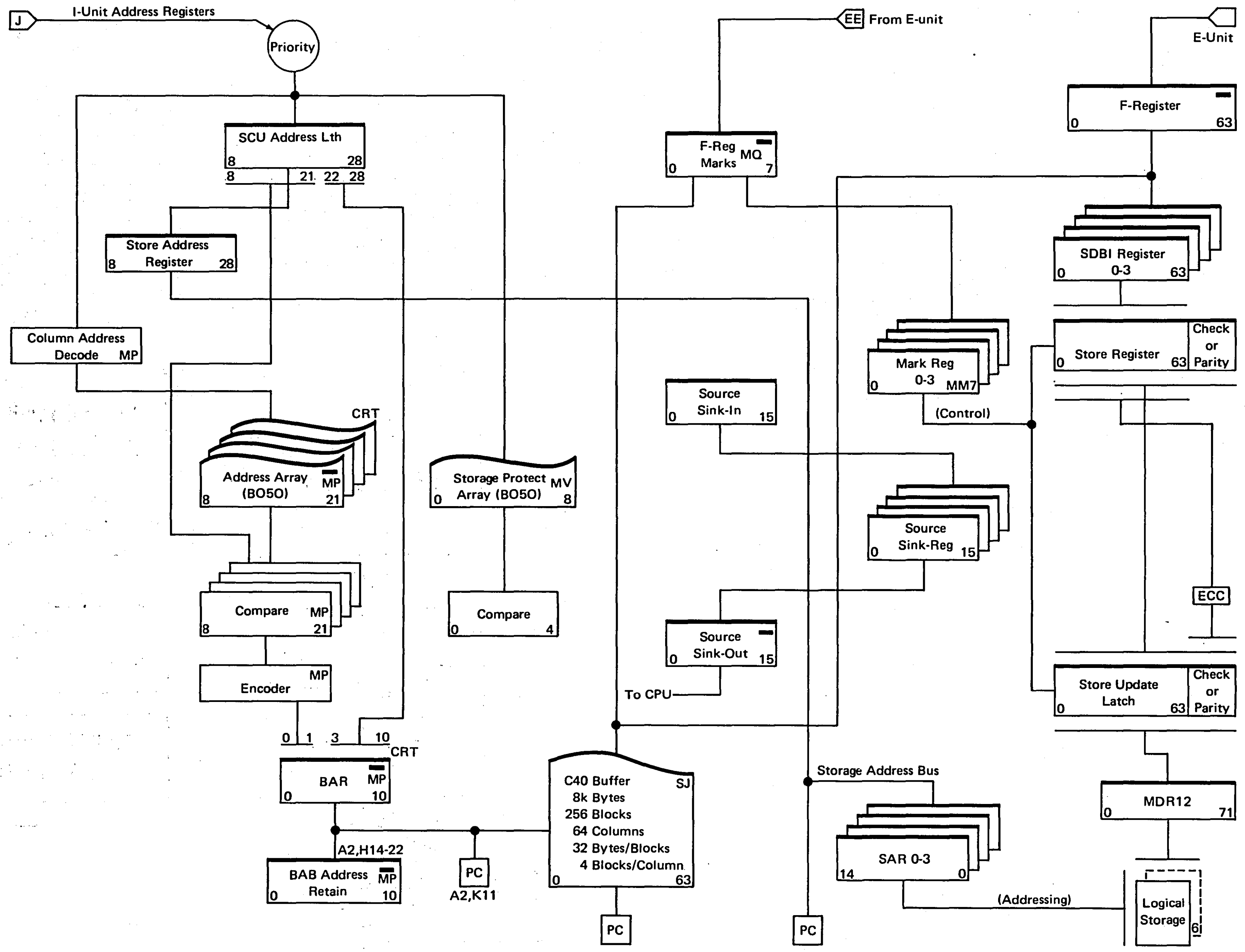


Figure 12-12. Simplified Data Flow - CPU Store

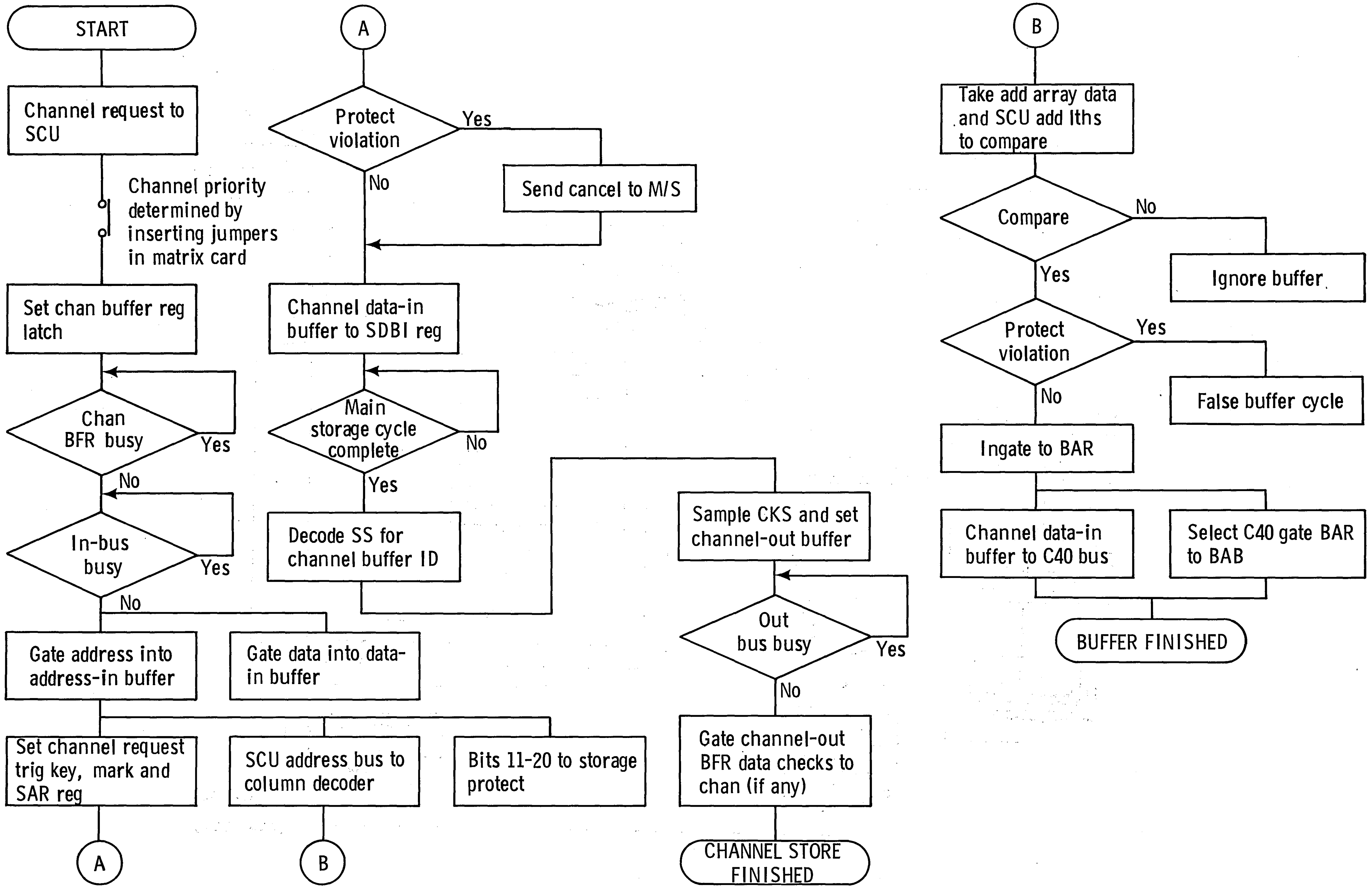

START
Channel request to SCU
Channel priority determined by inserting jumpers in matrix card
Set chan buffer reg latch
Chan BFR busy
Yes
No
In-bus busy
Yes
No
Gate address into address-in buffer
Gate data into data-in buffer
Set channel request trig key, mark and SAR reg
SCU address bus to column decoder
Bits 11-20 to storage protect
A
B
A
Protect violation
Yes
No
Send cancel to M/S
Channel data-in buffer to SDBI reg
Main storage cycle complete
No
Yes
Decode SS for channel buffer ID
Sample CKS and set channel-out buffer
Out bus busy
Yes
No
Gate channel-out BFR data checks to chan (if any)
CHANNEL STORE FINISHED
B
Take add array data and SCU add lths to compare
Compare
No
Yes
Ignore buffer
Protect violation
Yes
No
False buffer cycle
Ingate to BAR
Channel data-in buffer to C40 bus
Select C40 gate BAR to BAB
BUFFER FINISHED

Figure 12-14. Channel Store (Address In High-Speed Buffer)

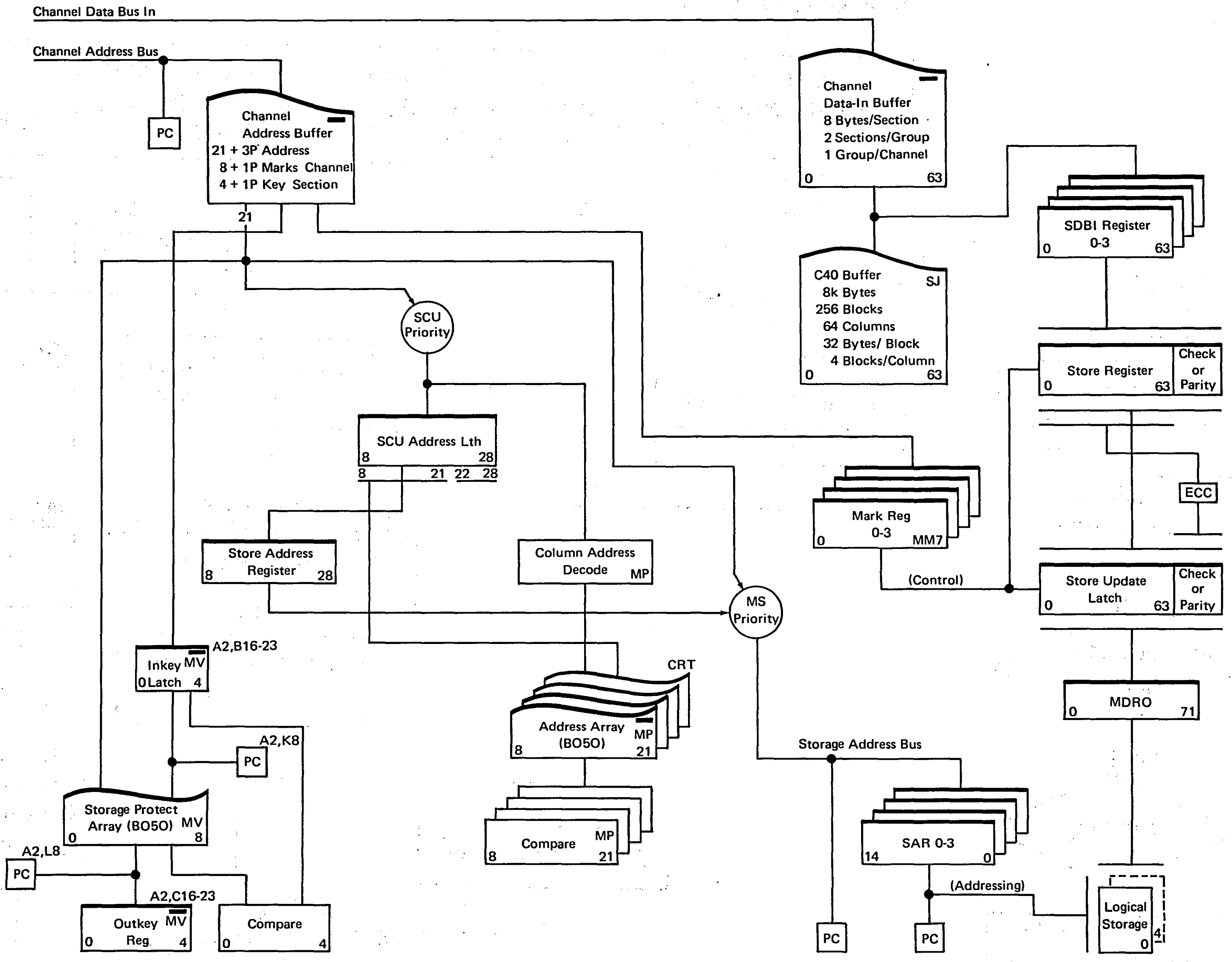


Figure 12-15. Simplified Data Flow - Channel Store

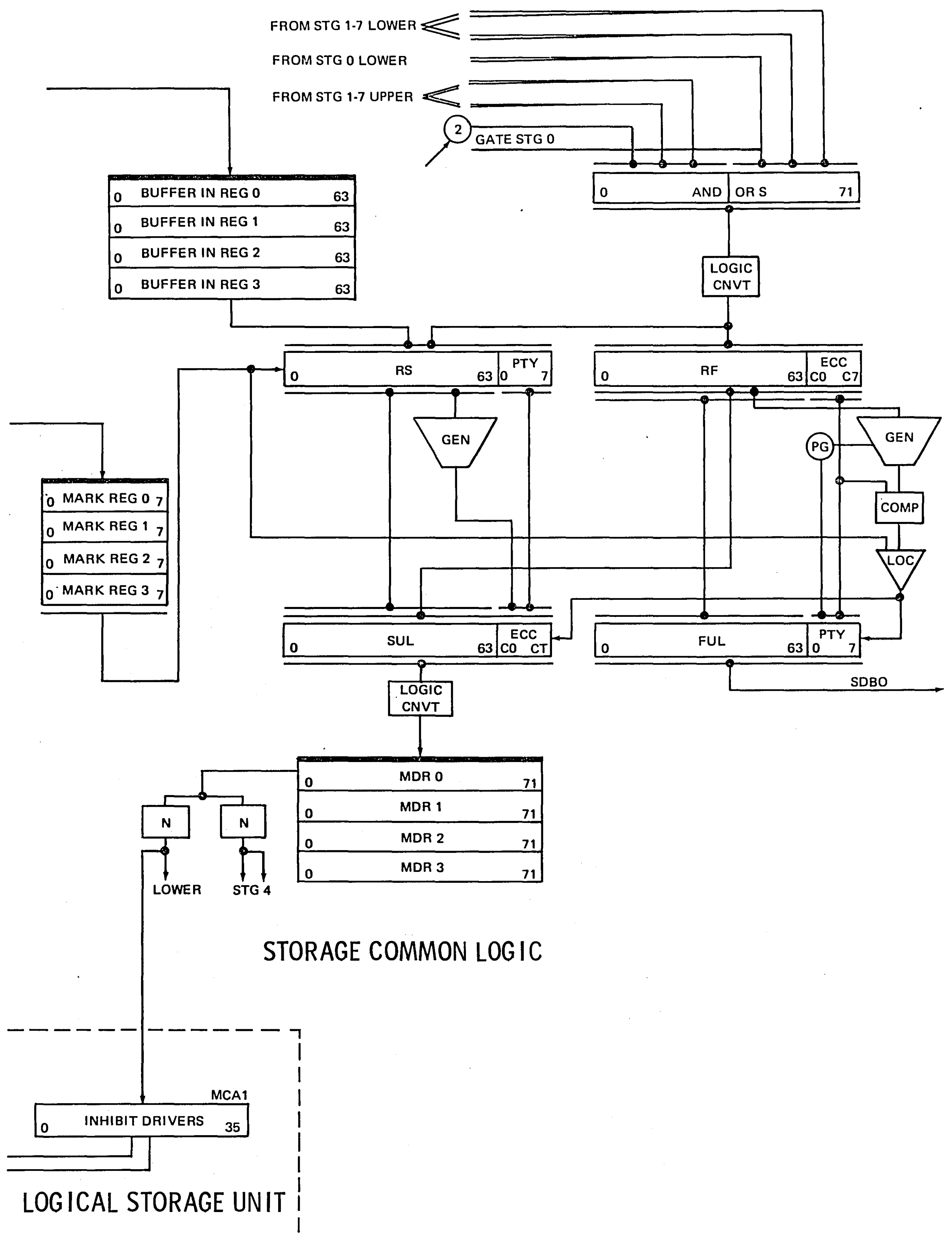


Figure 12-16. ECC Logic

CHANNEL BUFFERING

Each channel has a fixed assignment of one buffer group and each group consists of two sections. A buffer section stores all the information for one main storage access. Each buffer section stores data in, data out, marks, key, address, and various check and identification information.

Channel requests for input or output information must first vie for priority on the channel-in bus. The order of in-bus priority is established by inserting jumpers on a matrix card located in Frame 03. The wiring of the jumpers for a particular system results in a fixed-buffer assignment to each channel which is displayed on the microfiche viewer.

When in-bus priority is resolved, the appropriate channel is signalled. The channel, in turn, responds with address, marks, key and, if a store operation, a doubleword of data and a store signal. Upon the receipt of an address valid signal, this information is loaded into one of the channel's assigned buffer sections. If both buffer sections are empty (X 0 and X 1 where X is the buffer group number), the channel transfer is loaded into section X0. If a buffer section is not available for use (empty), that channel is blocked from receiving in-bus priority.

Once a buffer section is loaded, it is set to a busy state and entered as a contender into main storage priority. After main storage priority has been received, the storage unit is selected. Identification of the channel buffer section is placed on the source-sink bus-in, and the buffer section contents are gated to main storage. Main storage control receives the address, marks, and key. In the case of a store, the buffer section contents are also gated to the high-speed buffer control section to update the buffer if the address block is in the buffer.

When the channel request processing is completed by main storage, a storage signal followed by data, source-sink identification, and error information is sent to the main storage control which assembles and sends the information to the initiating channel buffer section.

When the information from storage is loaded into a channel buffer section, use of the channel out-bus is vied for by entering channel-out bus priority. When priority is resolved in favor of a buffer section the contents of that buffer section are gated onto the channel-out bus. At this point that buffer section is made not busy. All address and data type checks, associated with this channel request, are sent to the channel at the same time as the channel buffer data.

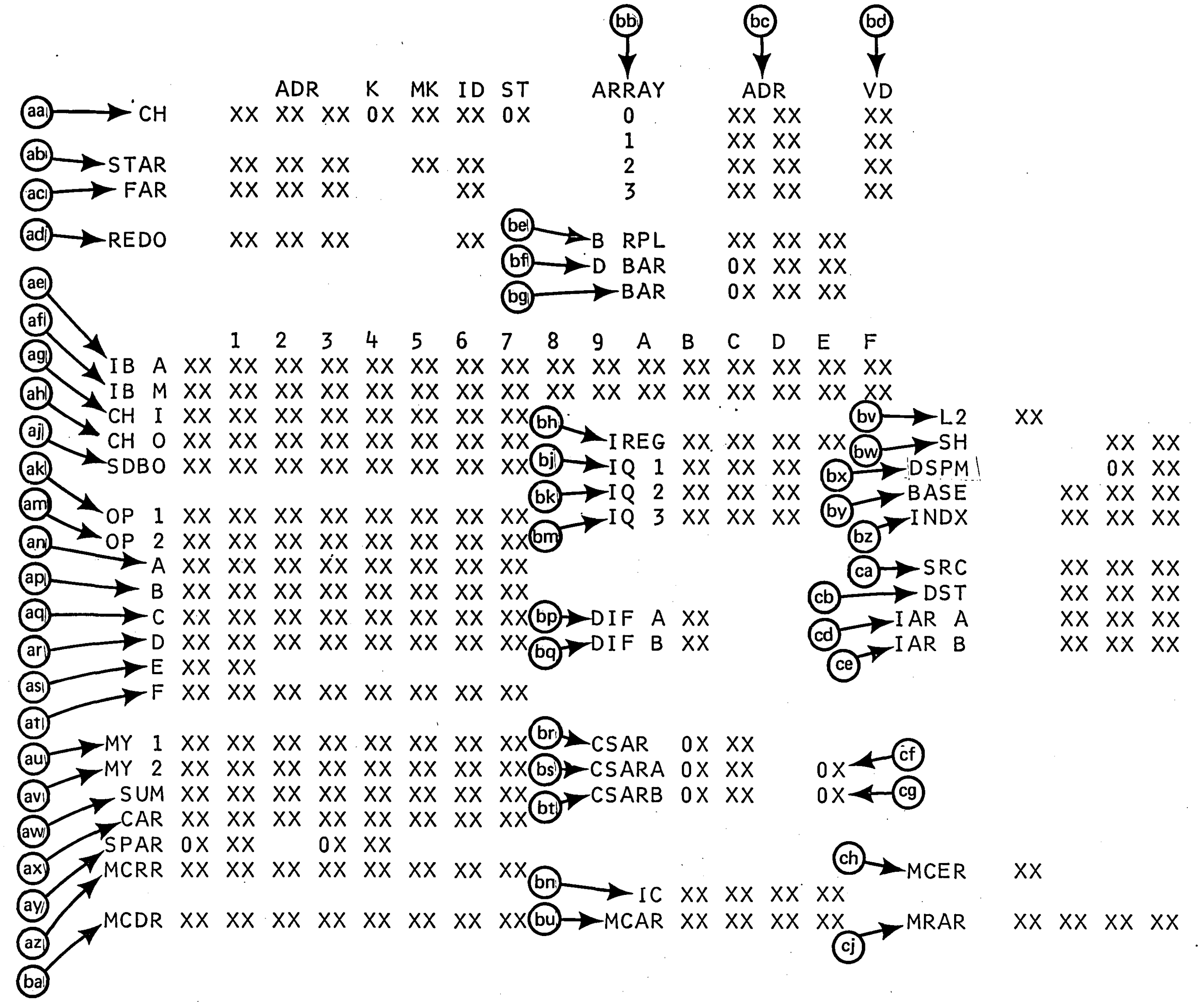
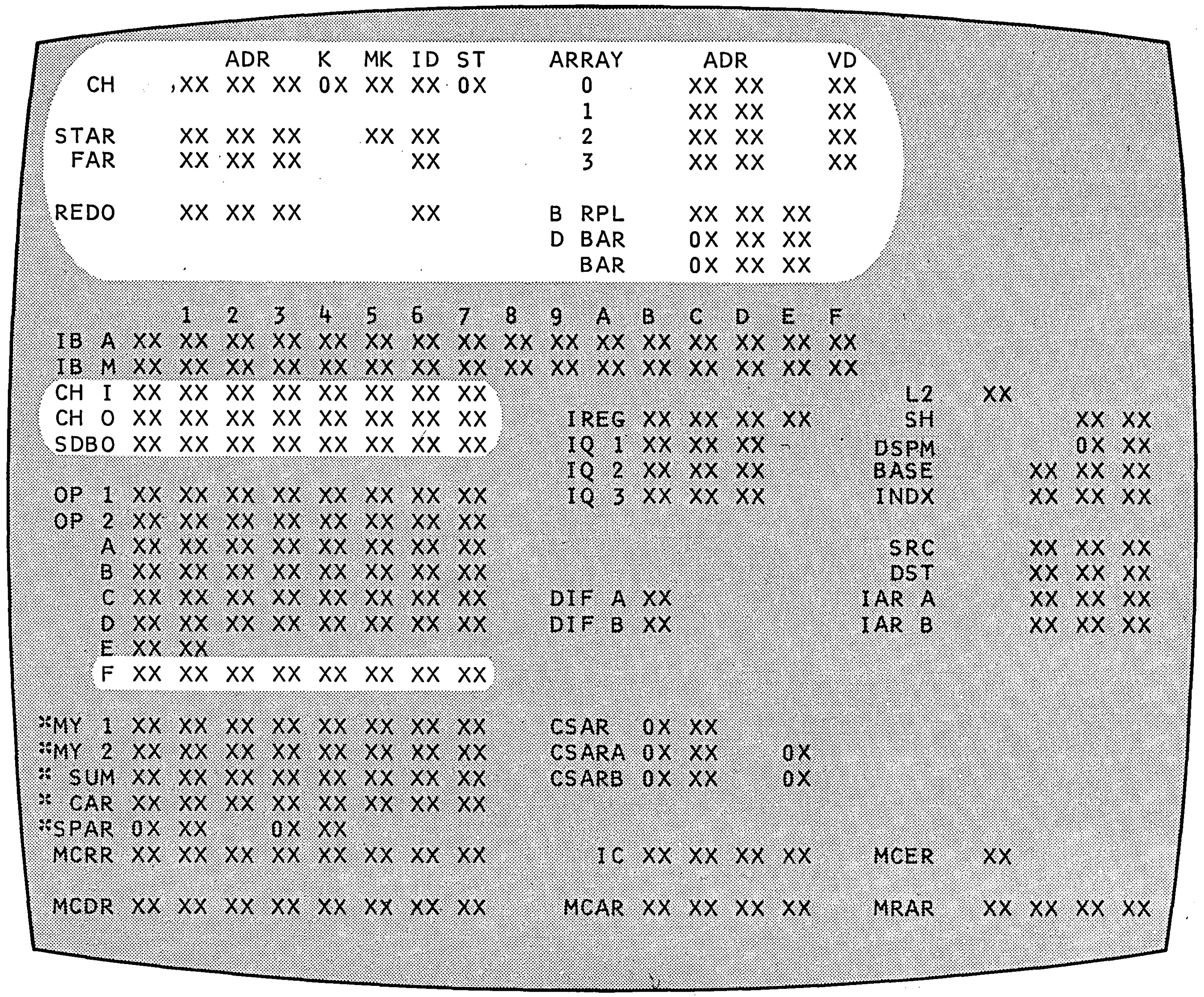


Figure 13-1.



```
           ADR  K  MK  ID  ST      ARRAY       ADR      VD
 CH    ,XX XX XX 0X XX XX 0X         0        XX XX     XX
                                     1        XX XX     XX
STAR    XX XX XX    XX XX            2        XX XX     XX
 FAR    XX XX XX       XX            3        XX XX     XX

REDO    XX XX XX       XX          B RPL      XX XX XX
                                   D BAR      0X XX XX
                                     BAR      0X XX XX

        1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
IB M XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
CH I XX XX XX XX XX XX XX XX                          L2   XX
CH O XX XX XX XX XX XX XX XX     IREG XX XX XX XX     SH           XX XX
SDBO XX XX XX XX XX XX XX XX     IQ 1 XX XX XX        DSPM         0X XX
                                 IQ 2 XX XX XX        BASE      XX XX XX
OP 1 XX XX XX XX XX XX XX XX     IQ 3 XX XX XX        INDX      XX XX XX
OP 2 XX XX XX XX XX XX XX XX
   A XX XX XX XX XX XX XX XX                           SRC      XX XX XX
   B XX XX XX XX XX XX XX XX                           DST      XX XX XX
   C XX XX XX XX XX XX XX XX     DIF A XX             IAR A     XX XX XX
   D XX XX XX XX XX XX XX XX     DIF B XX             IAR B     XX XX XX
   E XX XX
   F XX XX XX XX XX XX XX XX

:MY 1 XX XX XX XX XX XX XX XX    CSAR  0X XX
:MY 2 XX XX XX XX XX XX XX XX    CSARA 0X XX     0X
: SUM XX XX XX XX XX XX XX XX    CSARB 0X XX     0X
: CAR XX XX XX XX XX XX XX XX
:SPAR 0X XX    0X XX
MCRR  XX XX XX XX XX XX XX XX      IC XX XX XX XX     MCER  XX

MCDR  XX XX XX XX XX XX XX XX    MCAR XX XX XX XX     MRAR    XX XX XX XX
```

Figure 13-2. CRT With SCU Units Hi-Lighted

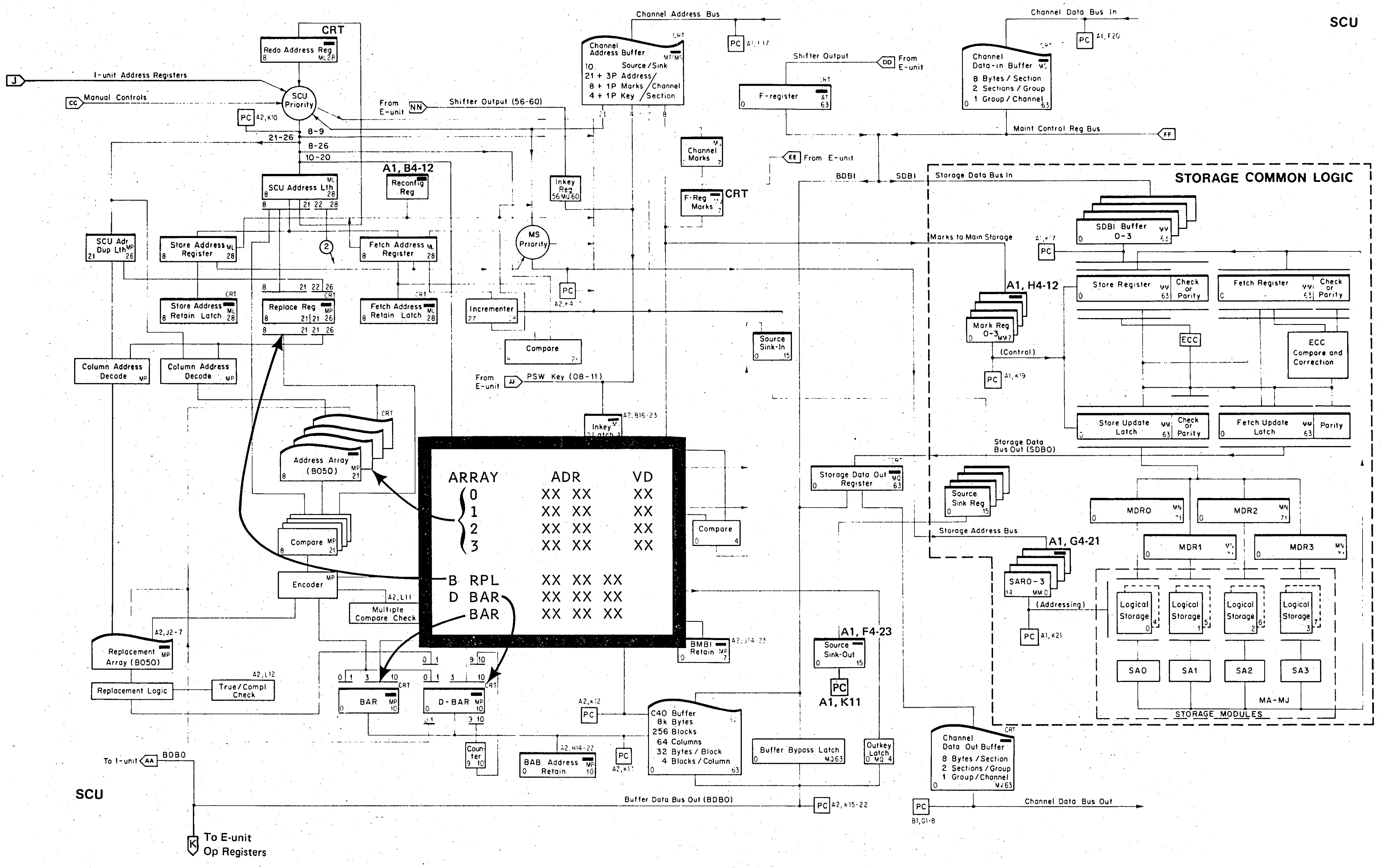


Figure 13-3. SCU Data Flow

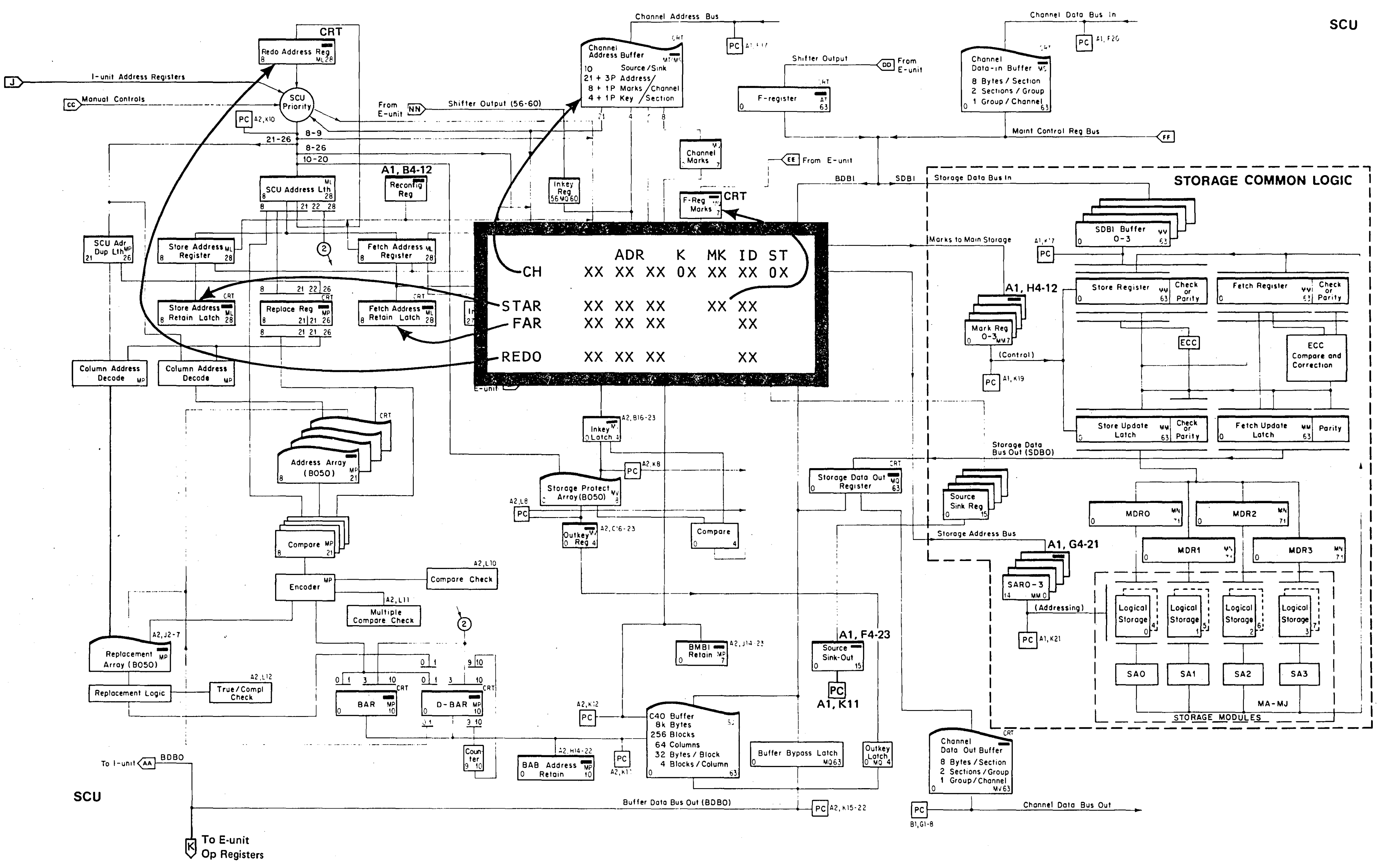


Figure 13-4. SCU Data Flow

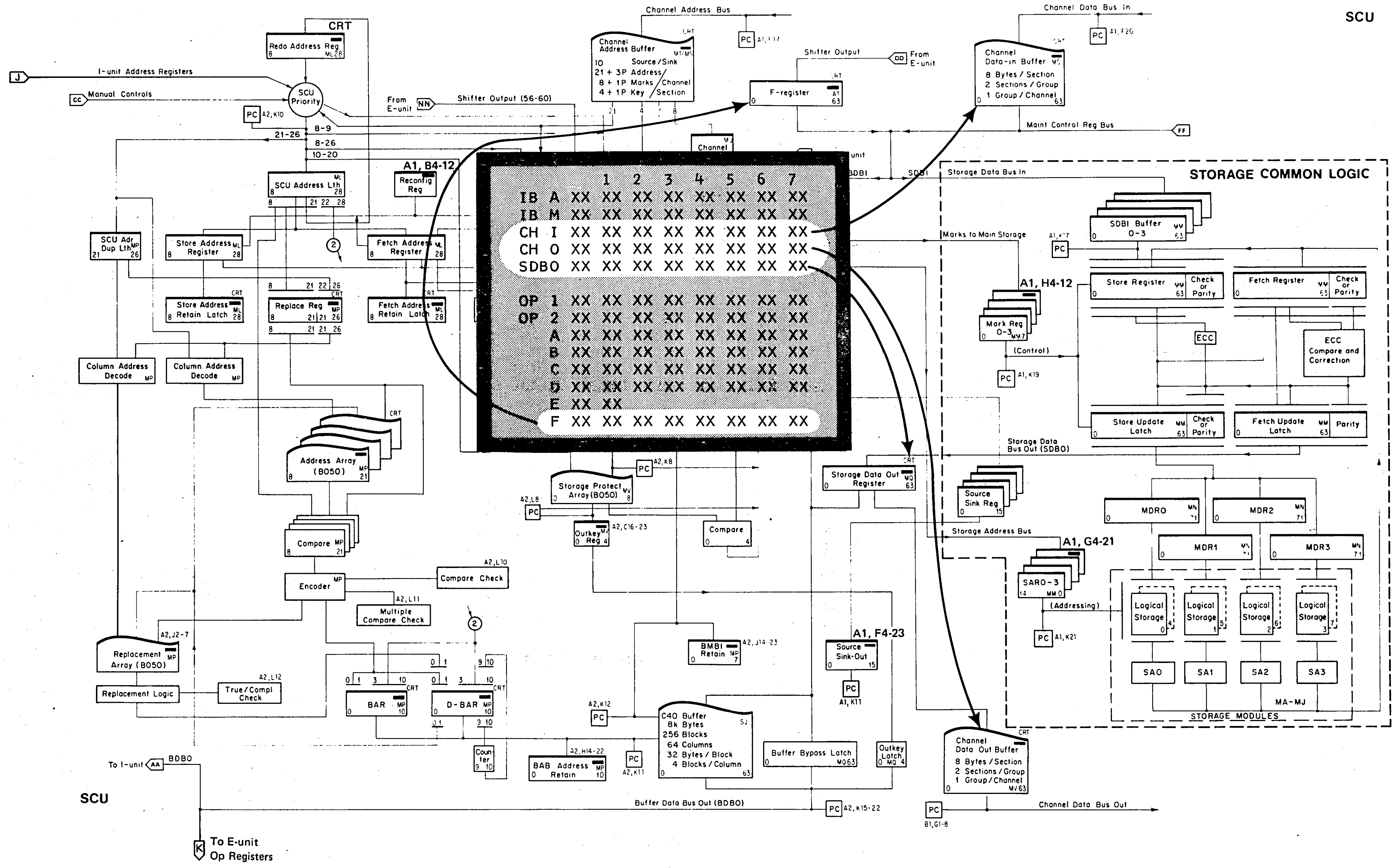


Figure 13-5. SCU Data Flow

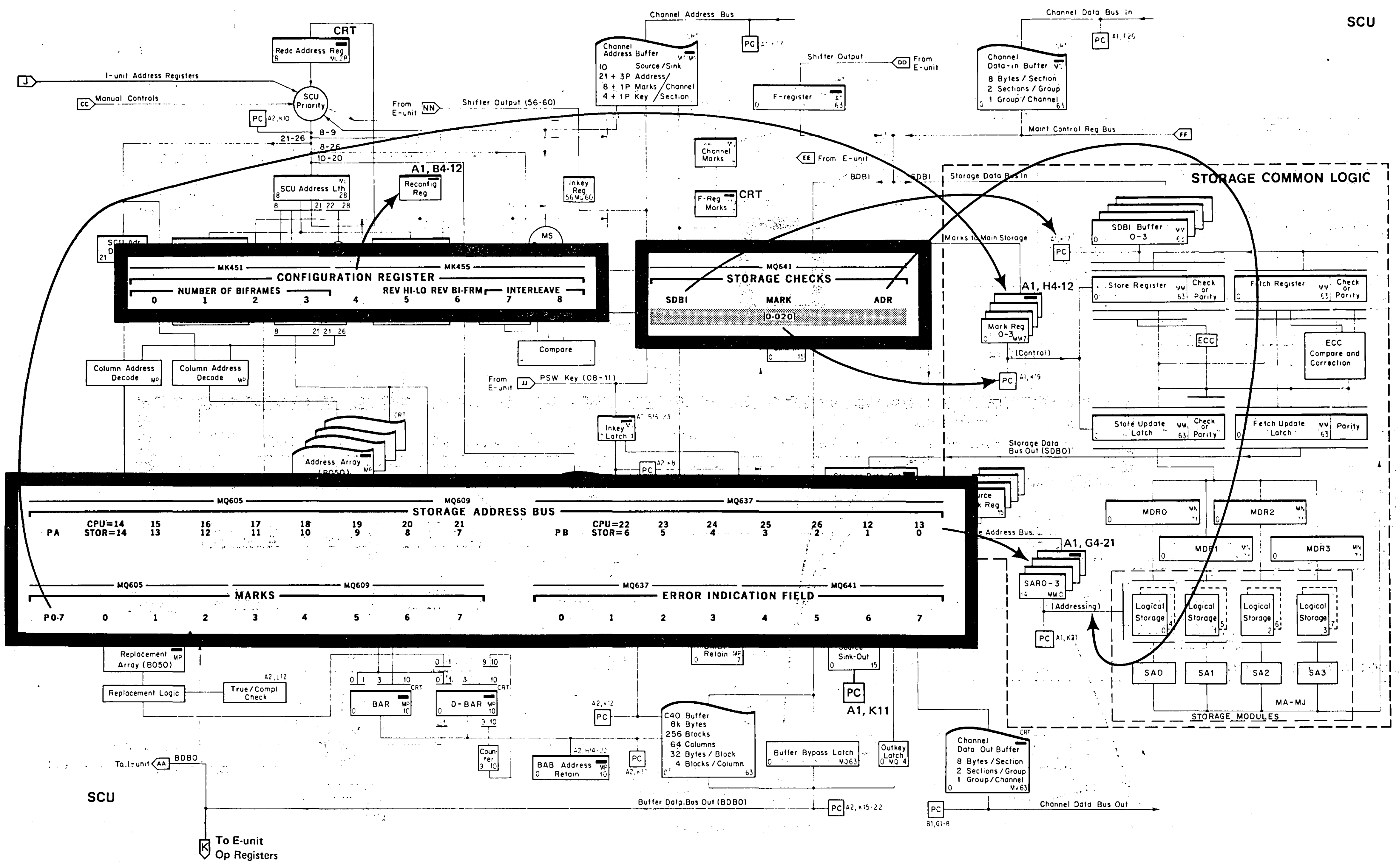


Figure 13-6. SCU Data Flow

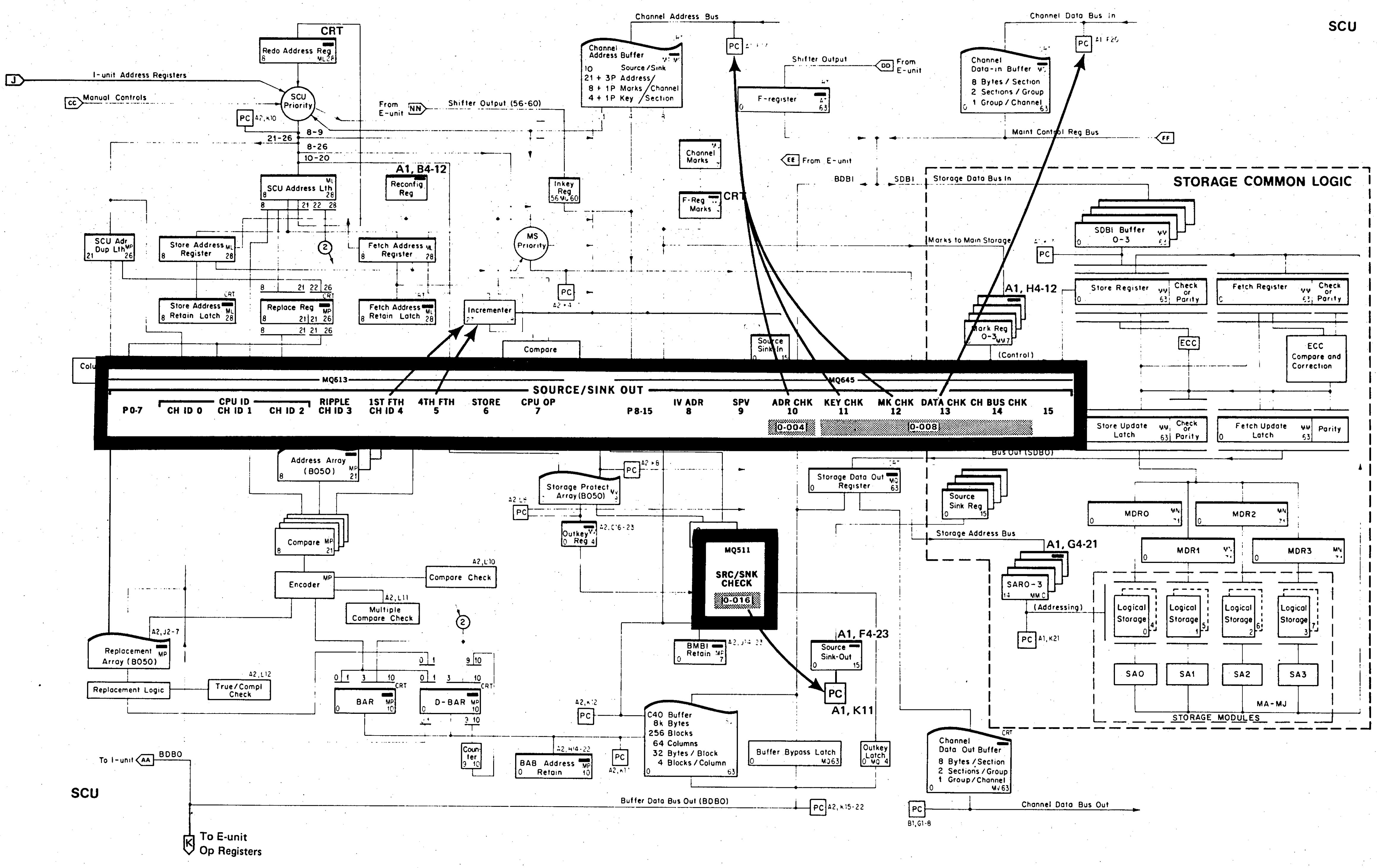


Figure 13-7. SCU

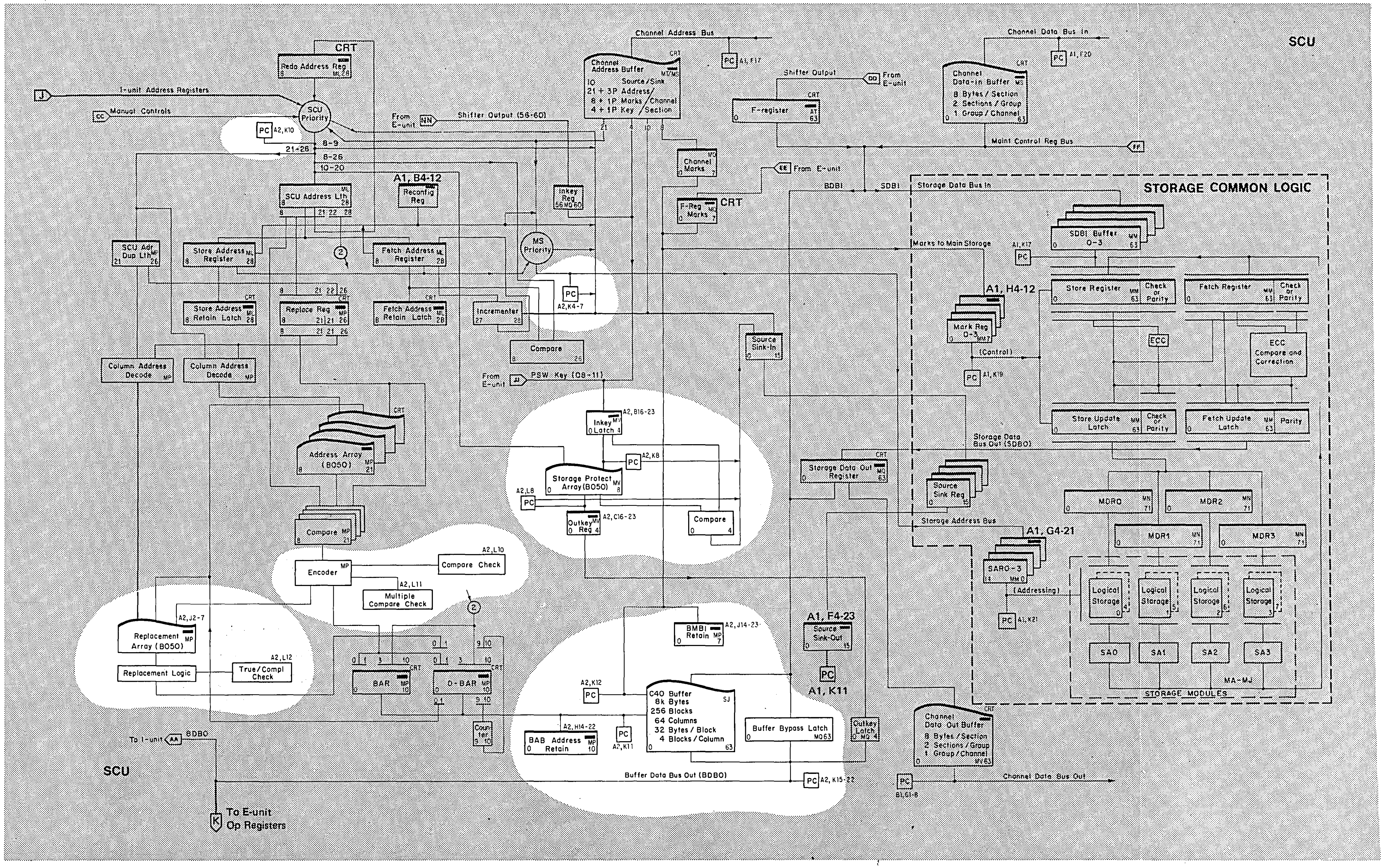


Figure 13-8. SCU

CPU Advance Lines
(From SCU to I-Unit, E-Unit, and Mnt. Ctrls.)

Advance 0 and 1
(Encoded as follows)

00	Null
01	Normal Advance
10	Invalid Address
11	Protected

Rescind Lines
(Within SCU)

00	No Request in FAR
01	Request in FAR
10	Not Used
11	I-Unit Rescinded

SCU Address Gate
(Within SCU)

Priority
ID Bits

0 1 2	
0 0 0	Not Used
0 0 1	Maint. Request
0 1 0	Unused
0 1 1	Maint. Ripple
1 0 0	Destination
1 0 1	Source
1 1 0	IARA
1 1 1	IARB

Figure 13-9. CPU Advance Lines

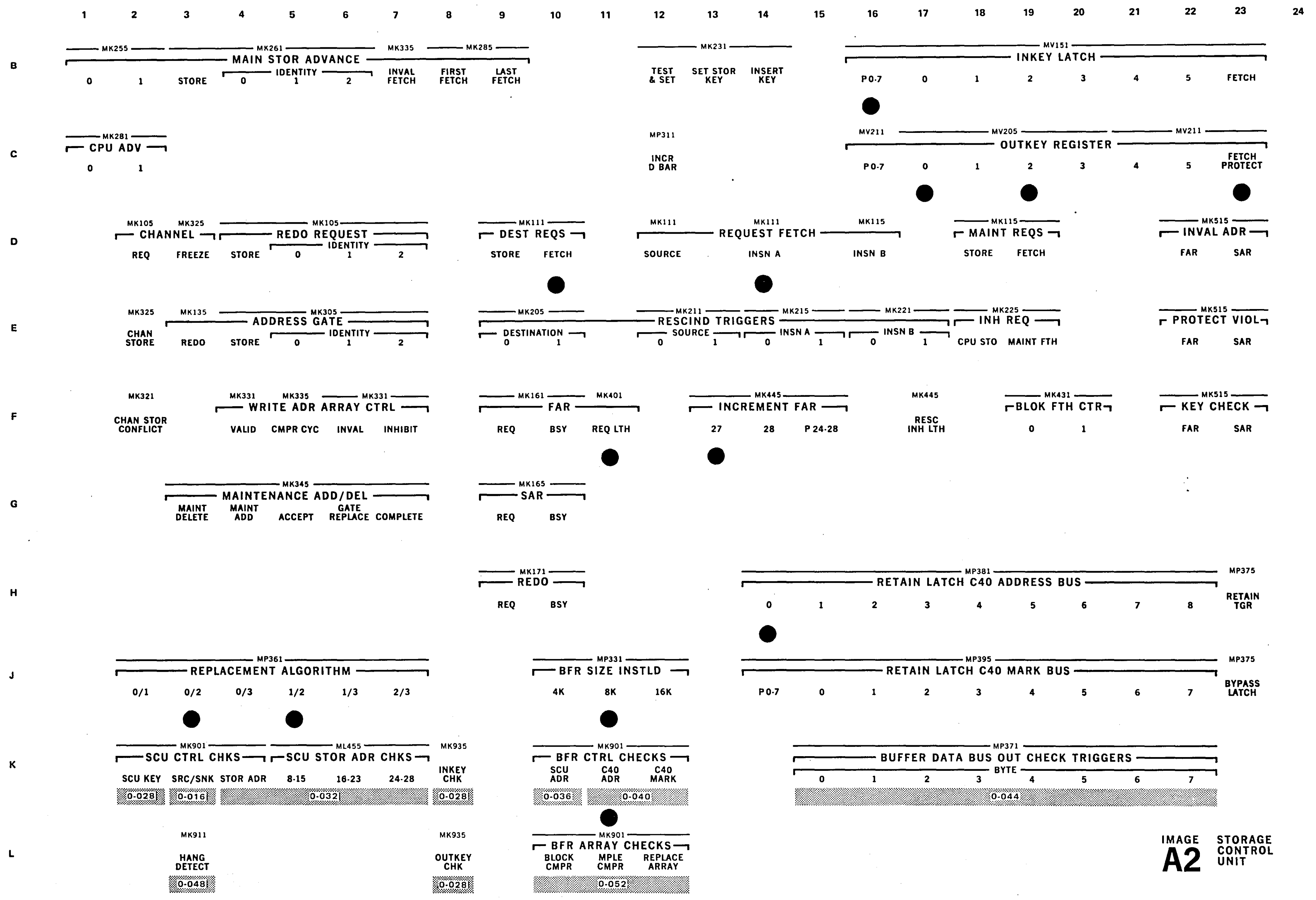

MK255 MK261 MAIN STOR ADVANCE MK335 MK285
0 1 STORE IDENTITY 0 1 2 INVAL FETCH FIRST FETCH LAST FETCH
MK231 TEST & SET SET STOR KEY INSERT KEY
MV151 INKEY LATCH P0-7 0 1 2 3 4 5 FETCH
MK281 CPU ADV 0 1
MP311 INCR D BAR
MV211 MV205 OUTKEY REGISTER MV211 P0-7 0 1 2 3 4 5 FETCH PROTECT
MK105 CHANNEL REQ MK325 FREEZE
MK105 REDO REQUEST STORE IDENTITY 0 1 2
MK111 DEST REQS STORE FETCH
MK111 REQUEST FETCH SOURCE INSN A MK115 INSN B
MK115 MAINT REQS STORE FETCH
MK515 INVAL ADR FAR SAR
MK325 CHAN STORE MK135 REDO
MK305 ADDRESS GATE STORE IDENTITY 0 1 2
MK205 DESTINATION 0 1
MK211 RESCIND TRIGGERS SOURCE 0 1 MK215 INSN A 0 1 MK221 INSN B 0 1
MK225 INH REQ CPU STO MAINT FTH
MK515 PROTECT VIOL FAR SAR
MK321 CHAN STOR CONFLICT
MK331 MK335 MK331 WRITE ADR ARRAY CTRL VALID CMPR CYC INVAL INHIBIT
MK161 FAR REQ BSY MK401 REQ LTH
MK445 INCREMENT FAR 27 28 P 24-28
MK445 RESC INH LTH
MK431 BLOK FTH CTR 0 1
MK515 KEY CHECK FAR SAR
MK345 MAINTENANCE ADD/DEL MAINT DELETE MAINT ADD ACCEPT GATE REPLACE COMPLETE
MK165 SAR REQ BSY
MK171 REDO REQ BSY
MP381 RETAIN LATCH C40 ADDRESS BUS 0 1 2 3 4 5 6 7 8
MP375 RETAIN TGR
MP361 REPLACEMENT ALGORITHM 0/1 0/2 0/3 1/2 1/3 2/3
MP331 BFR SIZE INSTLD 4K 8K 16K
MP395 RETAIN LATCH C40 MARK BUS P0-7 0 1 2 3 4 5 6 7
MP375 BYPASS LATCH
MK901 SCU CTRL CHKS SCU KEY SRC/SNK 0-028 0-016
ML455 SCU STOR ADR CHKS STOR ADR 8-15 16-23 24-28 0-032
MK935 INKEY CHK 0-028
MK901 BFR CTRL CHECKS SCU ADR C40 ADR C40 MARK 0-036 0-040
MP371 BUFFER DATA BUS OUT CHECK TRIGGERS BYTE 0 1 2 3 4 5 6 7 0-044
MK911 HANG DETECT 0-048
MK935 OUTKEY CHK 0-028
MK901 BFR ARRAY CHECKS BLOCK CMPR MPLE CMPR REPLACE ARRAY 0-052
IMAGE A2 STORAGE CONTROL UNIT

Figure 14-1. Microfiche Image A2

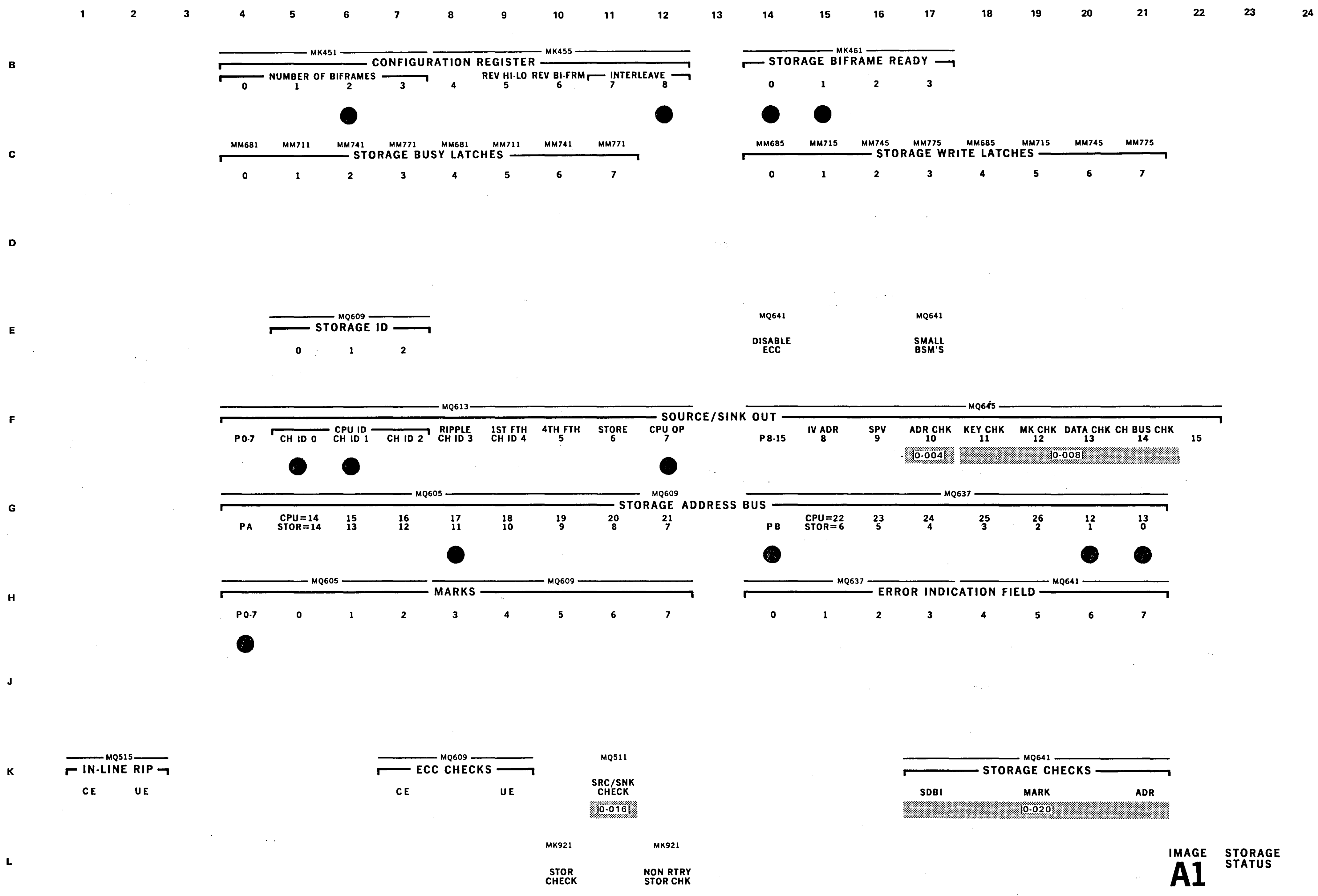


Figure 14-2. Microfiche Image A1

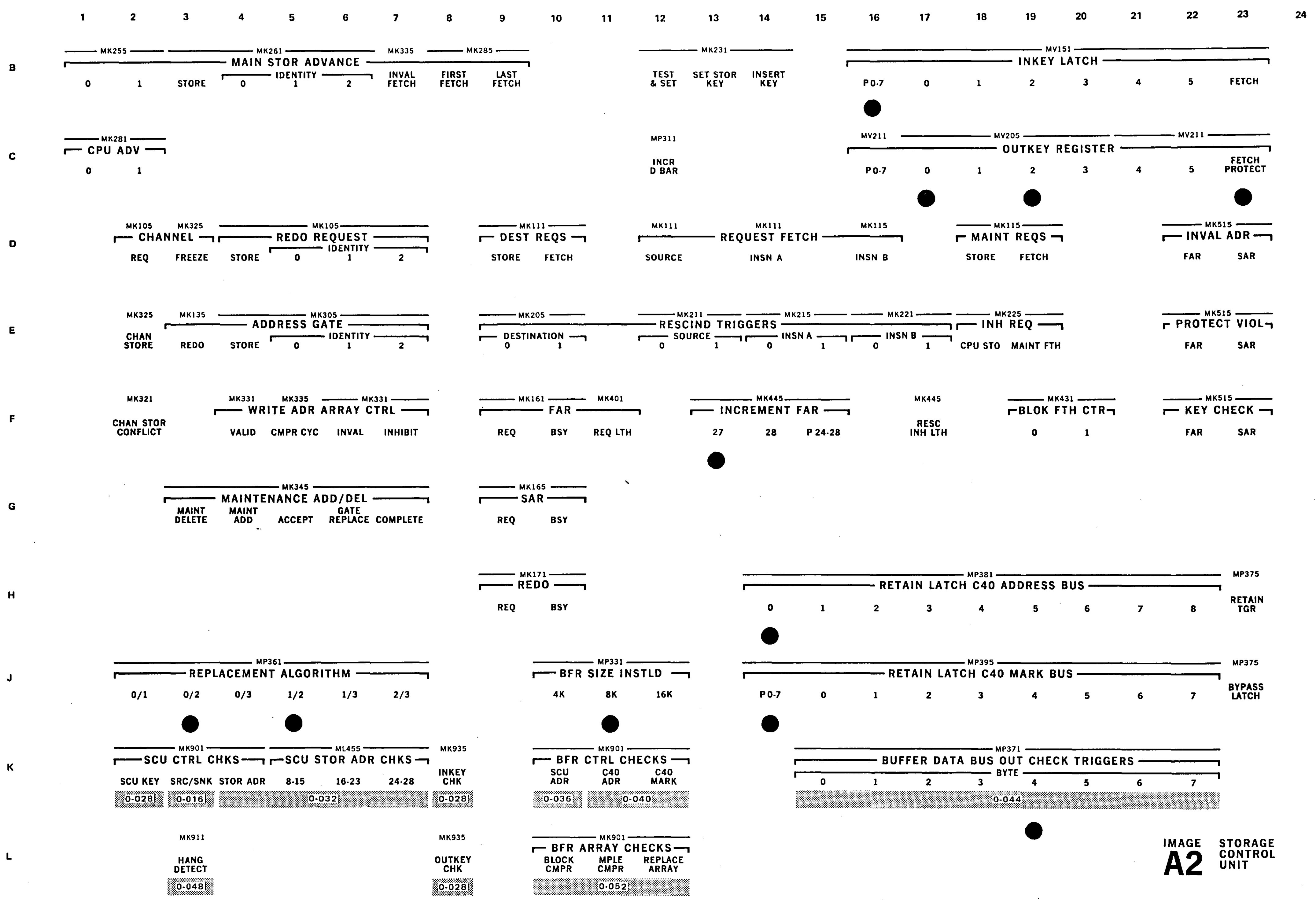
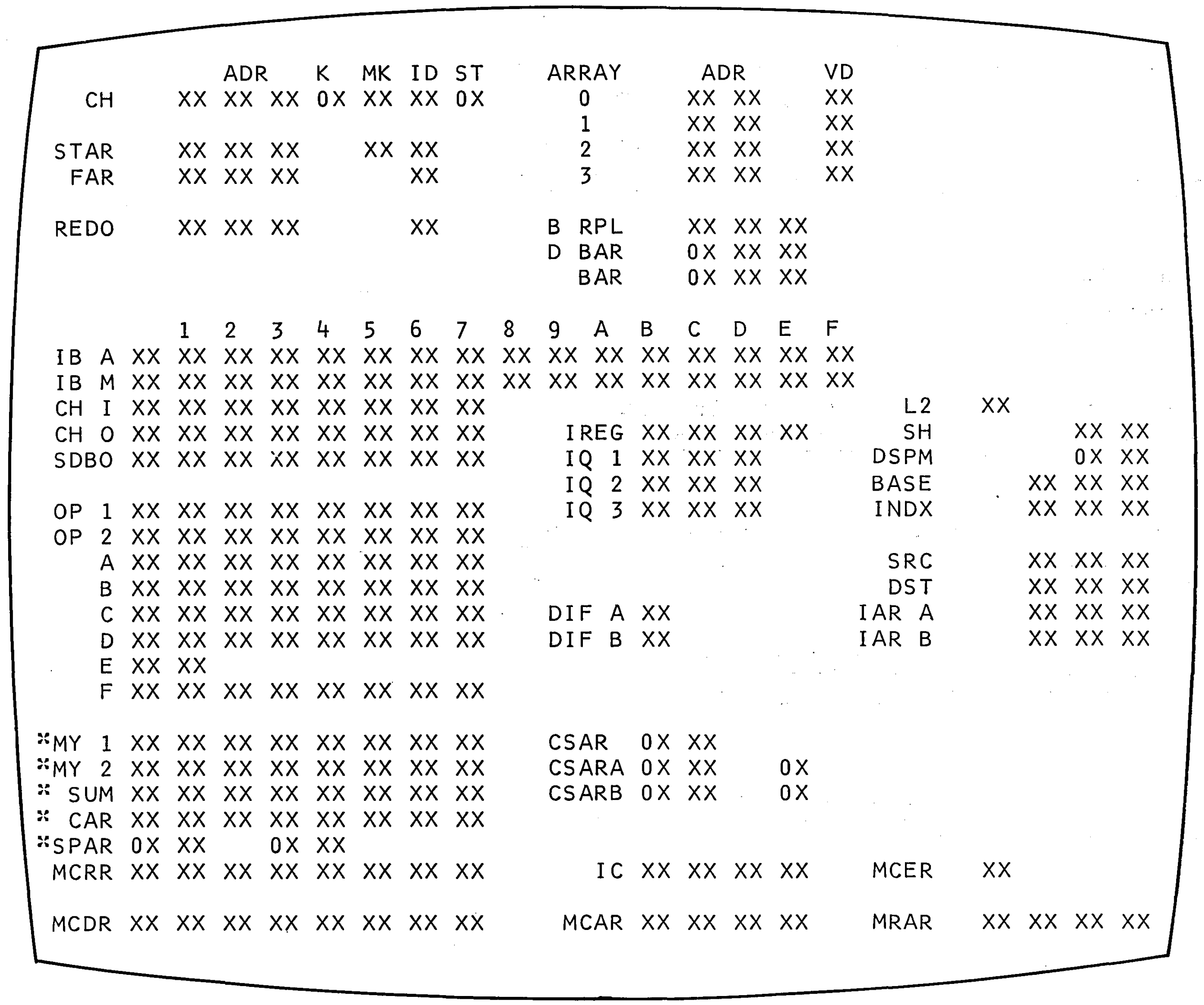


Figure 14-3. Microfiche Image A2



```
            ADR   K  MK ID ST    ARRAY     ADR       VD
   CH    XX XX XX 0X XX XX 0X      0      XX XX    XX
                                   1      XX XX    XX
 STAR    XX XX XX    XX XX         2      XX XX    XX
  FAR    XX XX XX       XX         3      XX XX    XX

 REDO    XX XX XX       XX       B RPL    XX XX XX
                                 D BAR    0X XX XX
                                   BAR    0X XX XX

         1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
 IB A XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
 IB M XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
 CH I XX XX XX XX XX XX XX XX                           L2   XX
 CH O XX XX XX XX XX XX XX XX     IREG XX XX XX XX     SH          XX XX
 SDBO XX XX XX XX XX XX XX XX     IQ 1 XX XX XX       DSPM         0X XX
                                  IQ 2 XX XX XX       BASE      XX XX XX
 OP 1 XX XX XX XX XX XX XX XX     IQ 3 XX XX XX       INDX      XX XX XX
 OP 2 XX XX XX XX XX XX XX XX
    A XX XX XX XX XX XX XX XX                          SRC      XX XX XX
    B XX XX XX XX XX XX XX XX                          DST      XX XX XX
    C XX XX XX XX XX XX XX XX    DIF A XX            IAR A      XX XX XX
    D XX XX XX XX XX XX XX XX    DIF B XX            IAR B      XX XX XX
    E XX XX
    F XX XX XX XX XX XX XX XX

*MY 1 XX XX XX XX XX XX XX XX    CSAR  0X XX
*MY 2 XX XX XX XX XX XX XX XX    CSARA 0X XX    0X
* SUM XX XX XX XX XX XX XX XX    CSARB 0X XX    0X
* CAR XX XX XX XX XX XX XX XX
*SPAR 0X XX    0X XX
 MCRR XX XX XX XX XX XX XX XX       IC XX XX XX XX    MCER   XX

 MCDR XX XX XX XX XX XX XX XX     MCAR XX XX XX XX    MRAR   XX XX XX XX
```

Figure 14-4.

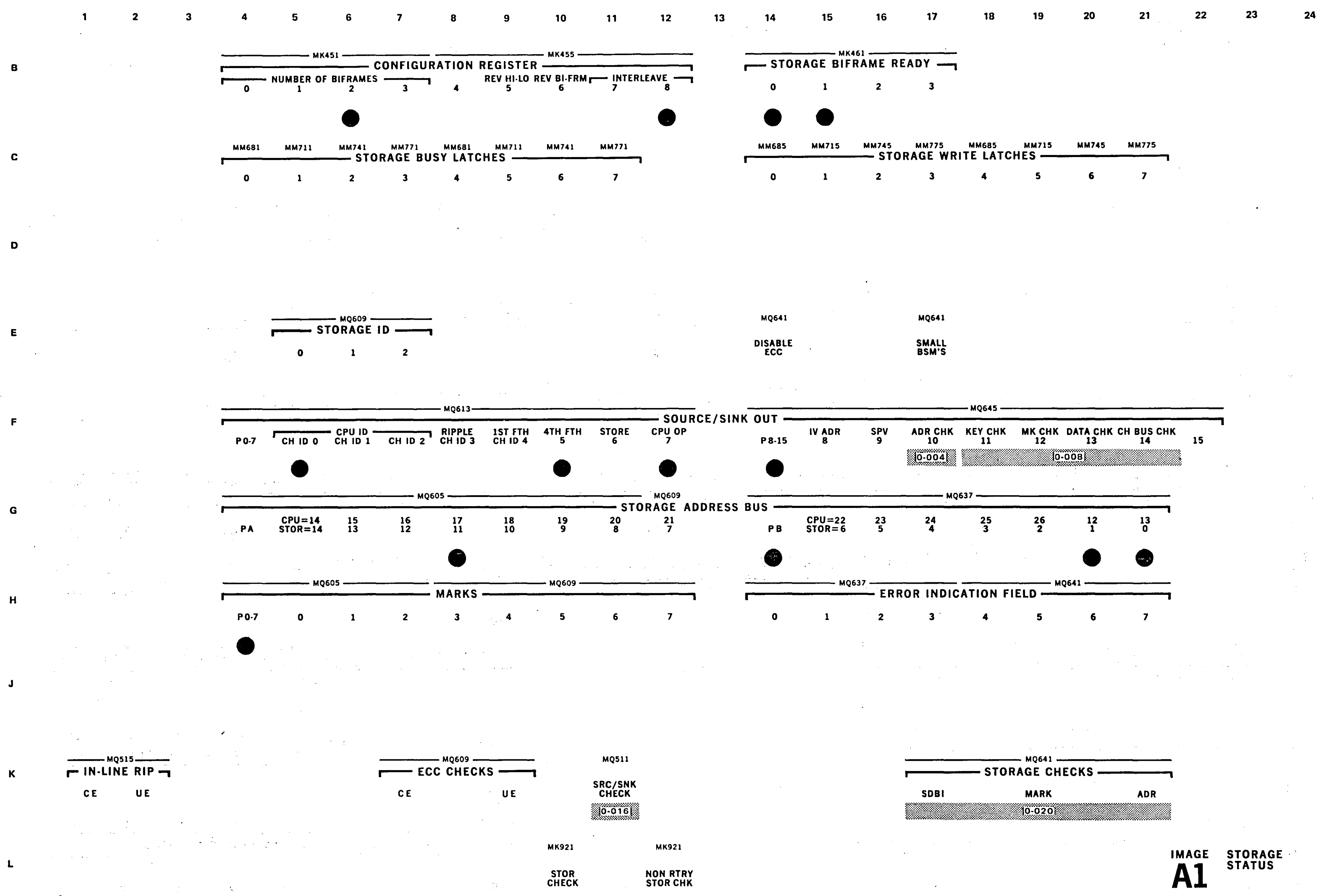


Figure 14-5. Microfiche Image A1

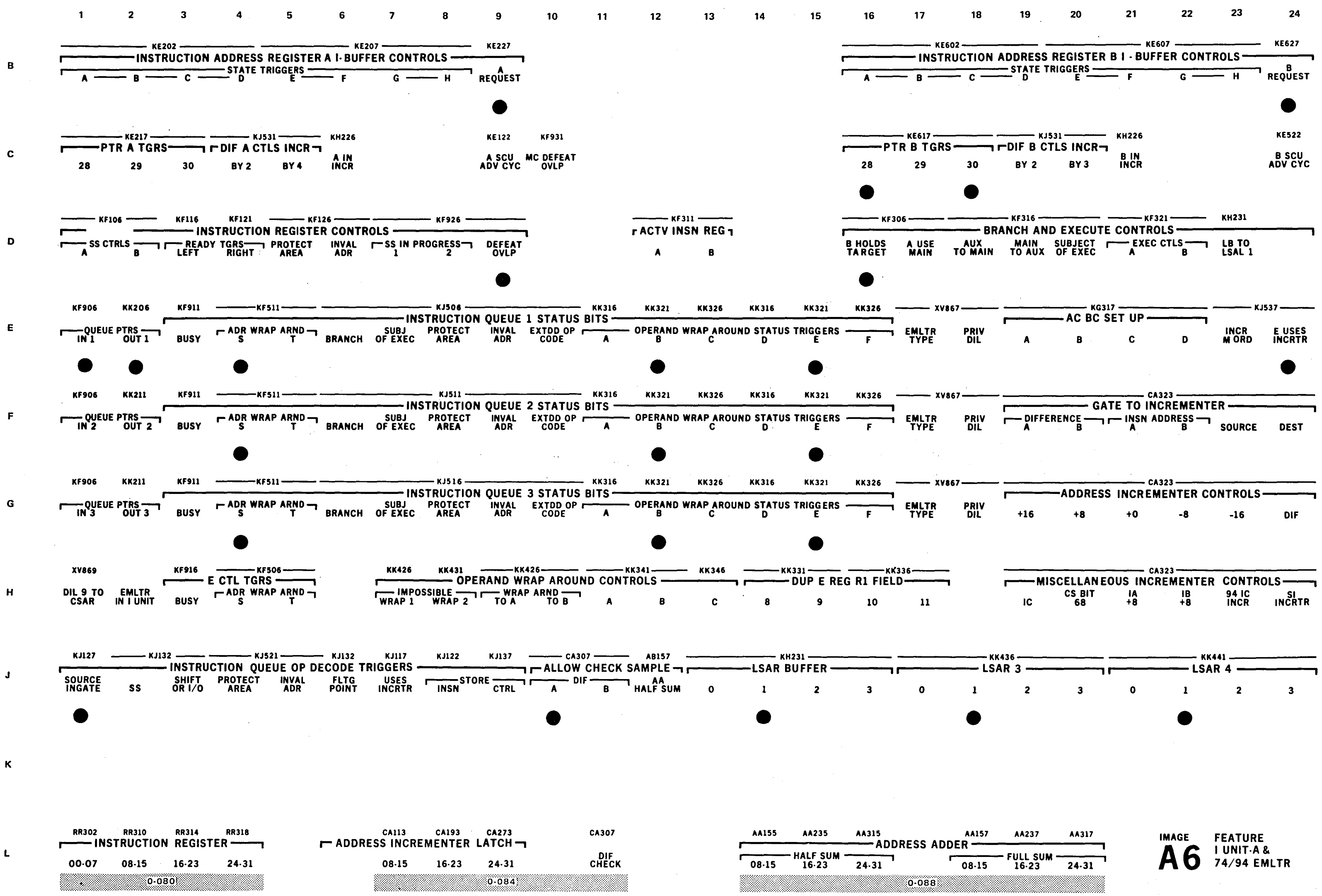


Figure 14-6. Microfiche Image A6

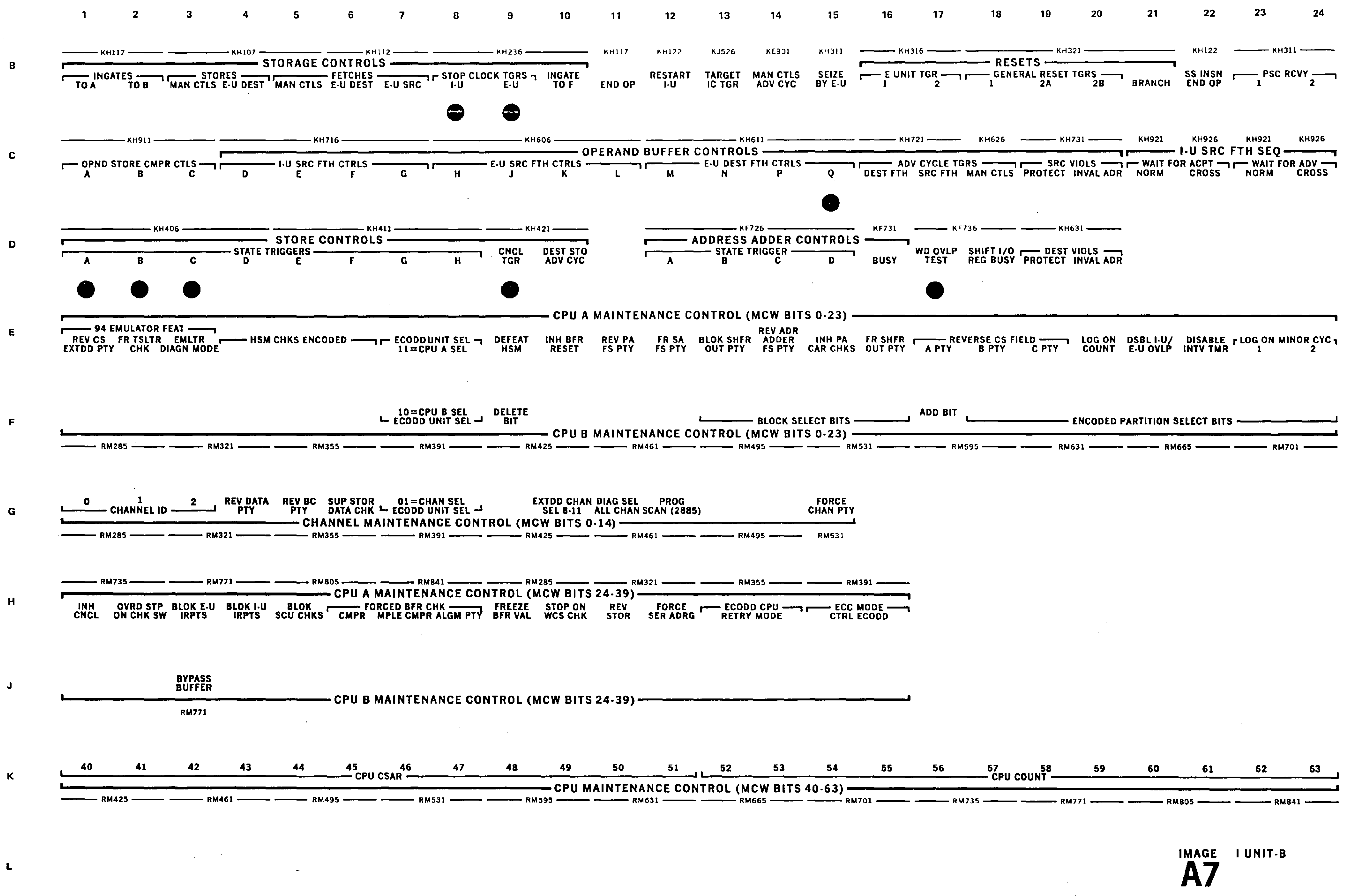

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
B
KH117 KH107 KH112 KH236 KH117 KH122 KJ526 KE901 KH311 KH316 KH321 KH122 KH311
STORAGE CONTROLS
INGATES TO A TO B
STORES MAN CTLS E-U DEST
FETCHES MAN CTLS E-U DEST E-U SRC
STOP CLOCK TGRS I-U E-U
INGATE TO F
END OP
RESTART I-U
TARGET IC TGR
MAN CTLS ADV CYC
SEIZE BY E-U
RESETS
E UNIT TGR 1 2
GENERAL RESET TGRS 1 2A 2B
BRANCH
SS INSN END OP
PSC RCVY 1 2
C
KH911 KH716 KH606 KH611 KH721 KH626 KH731 KH921 KH926 KH921 KH926
OPERAND BUFFER CONTROLS
OPND STORE CMPR CTLS A B C
I-U SRC FTH CTRLS D E F G
E-U SRC FTH CTRLS H J K L
E-U DEST FTH CTRLS M N P Q
ADV CYCLE TGRS DEST FTH SRC FTH MAN CTLS
SRC VIOLS PROTECT INVAL ADR
I-U SRC FTH SEQ
WAIT FOR ACPT NORM CROSS
WAIT FOR ADV NORM CROSS
D
KH406 KH411 KH421 KF726 KF731 KF736 KH631
STORE CONTROLS
STATE TRIGGERS A B C D E F G H
CNCL TGR
DEST STO ADV CYC
ADDRESS ADDER CONTROLS
STATE TRIGGER A B C D
BUSY
WD OVLP TEST
SHIFT I/O REG BUSY
DEST VIOLS PROTECT INVAL ADR
E
CPU A MAINTENANCE CONTROL (MCW BITS 0-23)
94 EMULATOR FEAT
REV CS EXTDD PTY
FR TSLTR CHK
EMLTR DIAGN MODE
HSM CHKS ENCODED
ECODDUNIT SEL 11=CPU A SEL
DEFEAT HSM
INH BFR RESET
REV PA FS PTY
FR SA FS PTY
BLOK SHFR OUT PTY
REV ADR ADDER FS PTY
INH PA CAR CHKS
FR SHFR OUT PTY
REVERSE CS FIELD A PTY B PTY C PTY
LOG ON COUNT
DSBL I-U/ E-U OVLP
DISABLE INTV TMR
LOG ON MINOR CYC 1 2
F
10=CPU B SEL ECODD UNIT SEL
DELETE BIT
BLOCK SELECT BITS
ADD BIT
ENCODED PARTITION SELECT BITS
CPU B MAINTENANCE CONTROL (MCW BITS 0-23)
RM285 RM321 RM355 RM391 RM425 RM461 RM495 RM531 RM595 RM631 RM665 RM701
G
0 1 2 CHANNEL ID
REV DATA PTY
REV BC PTY
SUP STOR DATA CHK
01=CHAN SEL ECODD UNIT SEL
EXTDD CHAN SEL 8-11
DIAG SEL ALL CHAN
PROG SCAN (2885)
FORCE CHAN PTY
CHANNEL MAINTENANCE CONTROL (MCW BITS 0-14)
RM285 RM321 RM355 RM391 RM425 RM461 RM495 RM531
H
RM735 RM771 RM805 RM841 RM285 RM321 RM355 RM391
CPU A MAINTENANCE CONTROL (MCW BITS 24-39)
INH CNCL
OVRD STP ON CHK SW
BLOK E-U IRPTS
BLOK I-U IRPTS
BLOK SCU CHKS
FORCED BFR CHK CMPR MPLE CMPR ALGM PTY
FREEZE BFR VAL
STOP ON WCS CHK
REV STOR
FORCE SER ADRG
ECODD CPU RETRY MODE
ECC MODE CTRL ECODD
J
BYPASS BUFFER
CPU B MAINTENANCE CONTROL (MCW BITS 24-39)
RM771
K
40 41 42 43 44 45 46 47 48 49 50 51 CPU CSAR
52 53 54 55 56 57 58 59 60 61 62 63 CPU COUNT
CPU MAINTENANCE CONTROL (MCW BITS 40-63)
RM425 RM461 RM495 RM531 RM595 RM631 RM665 RM701 RM735 RM771 RM805 RM841
L
IMAGE A7
I UNIT-B

Figure 14-7. Microfiche Image A7

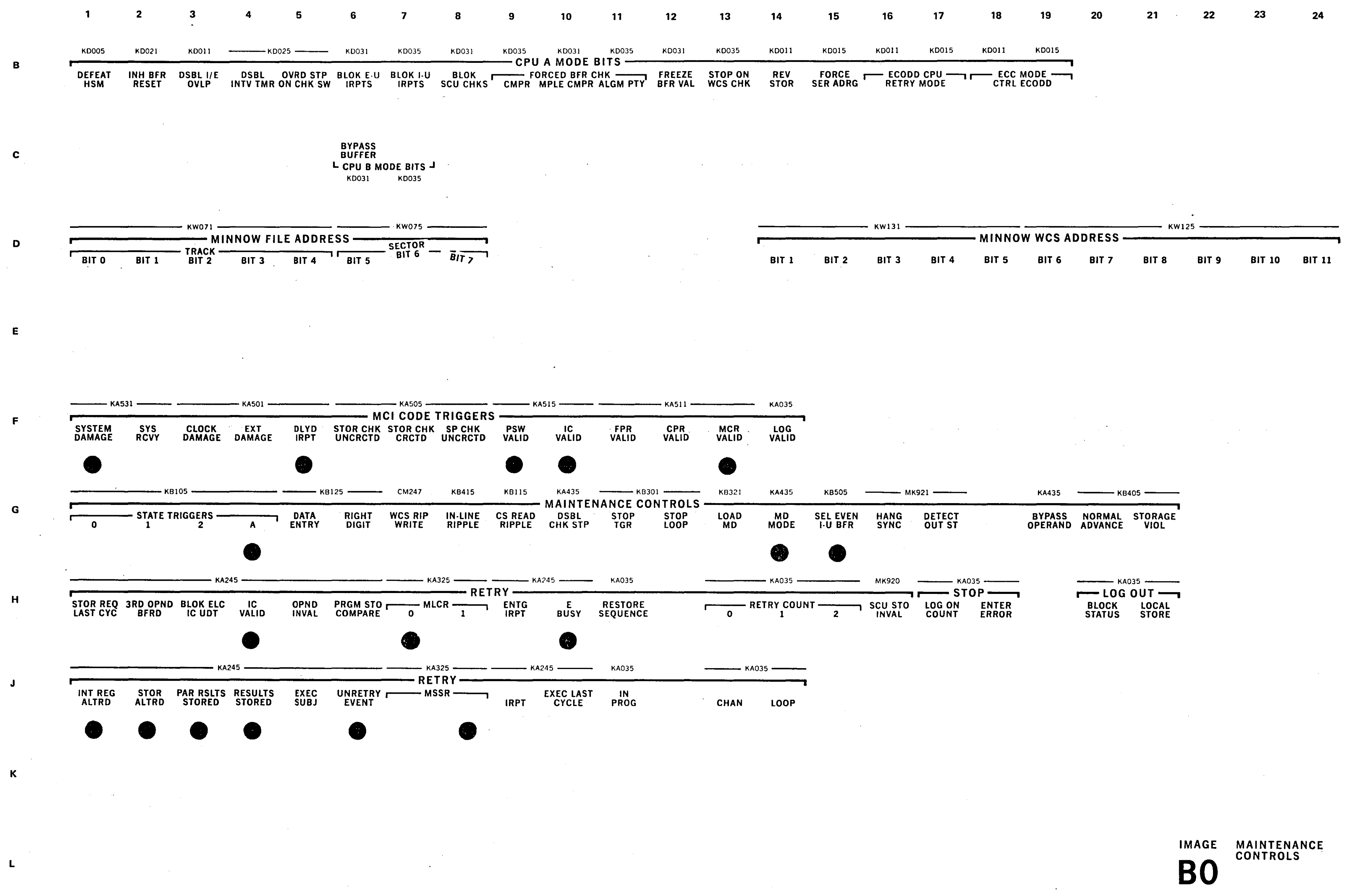
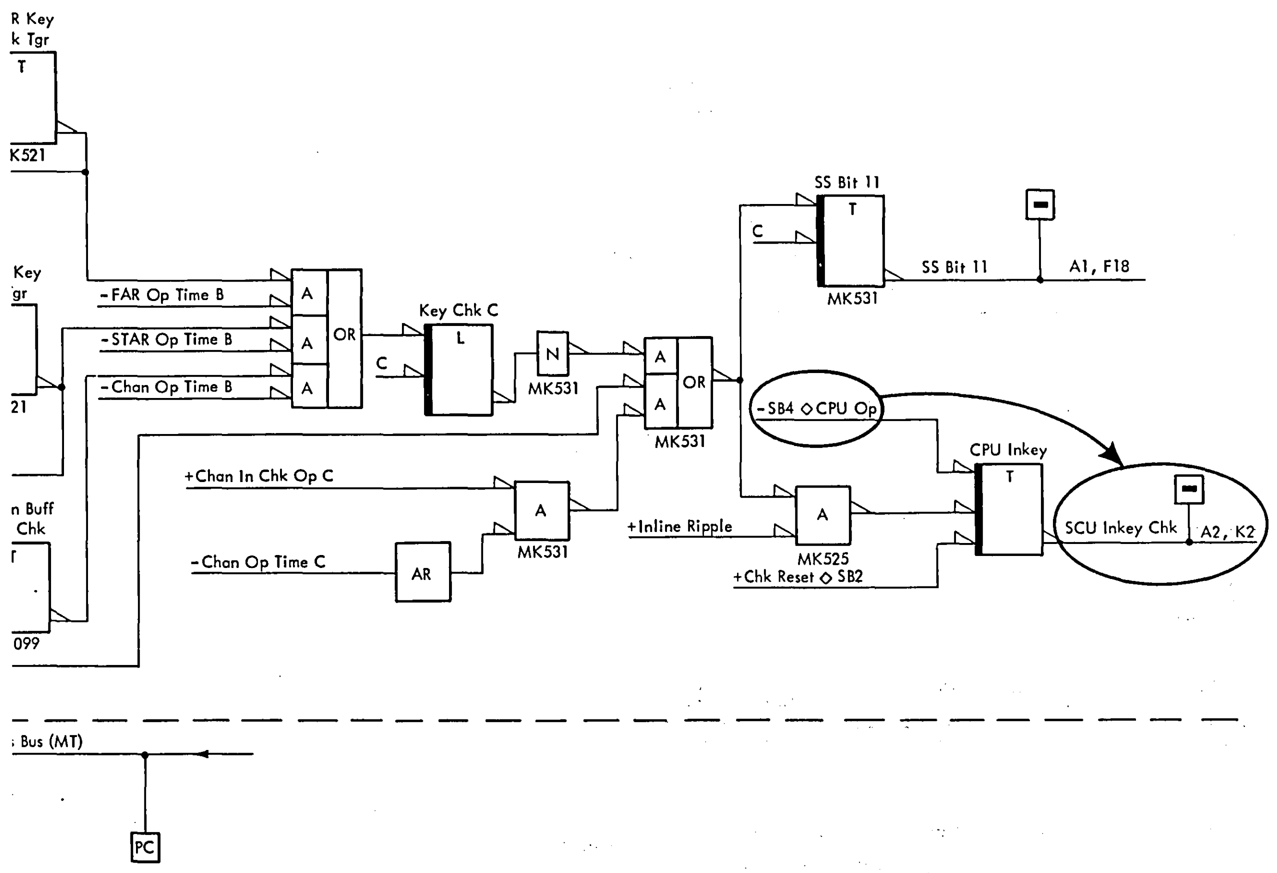


Figure 14-8. Microfiche Image B0



SCU Key Control Check Indicator (A2, K2): This light turns on if an SP inkey or outkey check occurs during a CPU operation.

Inkey Check Indicator (A2, K8) and Outkey Check Indicator (A2, L8): These lights turn on when errors are detected at their respective sides of the storage protect unit.

Suggested Diagnostics:
1017 Micro Error Check Section
1391 SCU Buffer and Addressing
13CB Storage Protect
13E0 Random
E401, E402, E403 for 2860/2870
E431, E436 for 2870

Related Diagrams:
SCU Channel Controls
Advance, Key, and Data Indicators; Microfiche Frame A2
SCU HS Buffer Controls, Change Key Sequence
SCU HS Buffer Controls, Timing for Block Load and Change Key Sequence
Functional Significance of Storage Address Bits

Figure 14-9. SCU Key Control Check

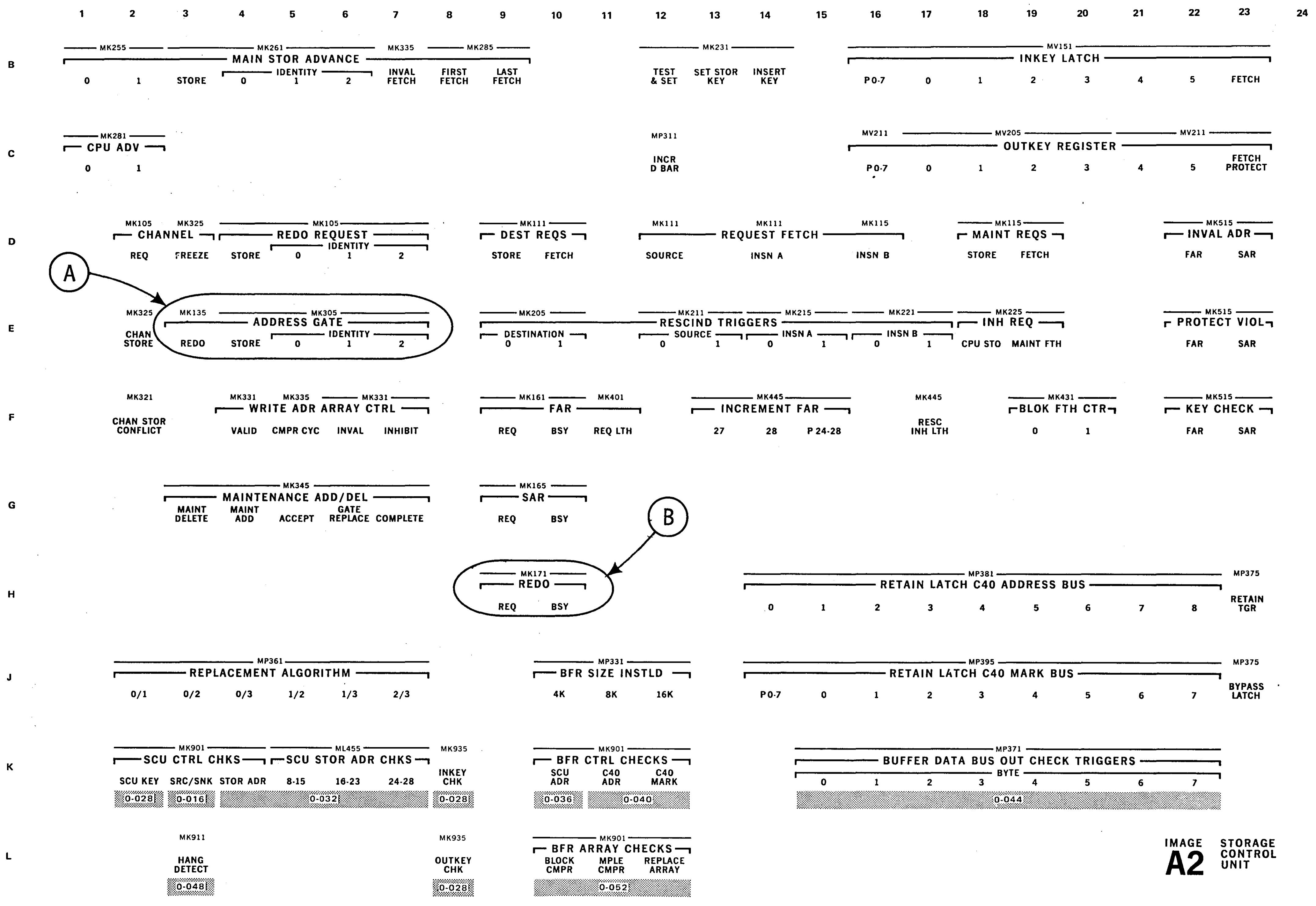
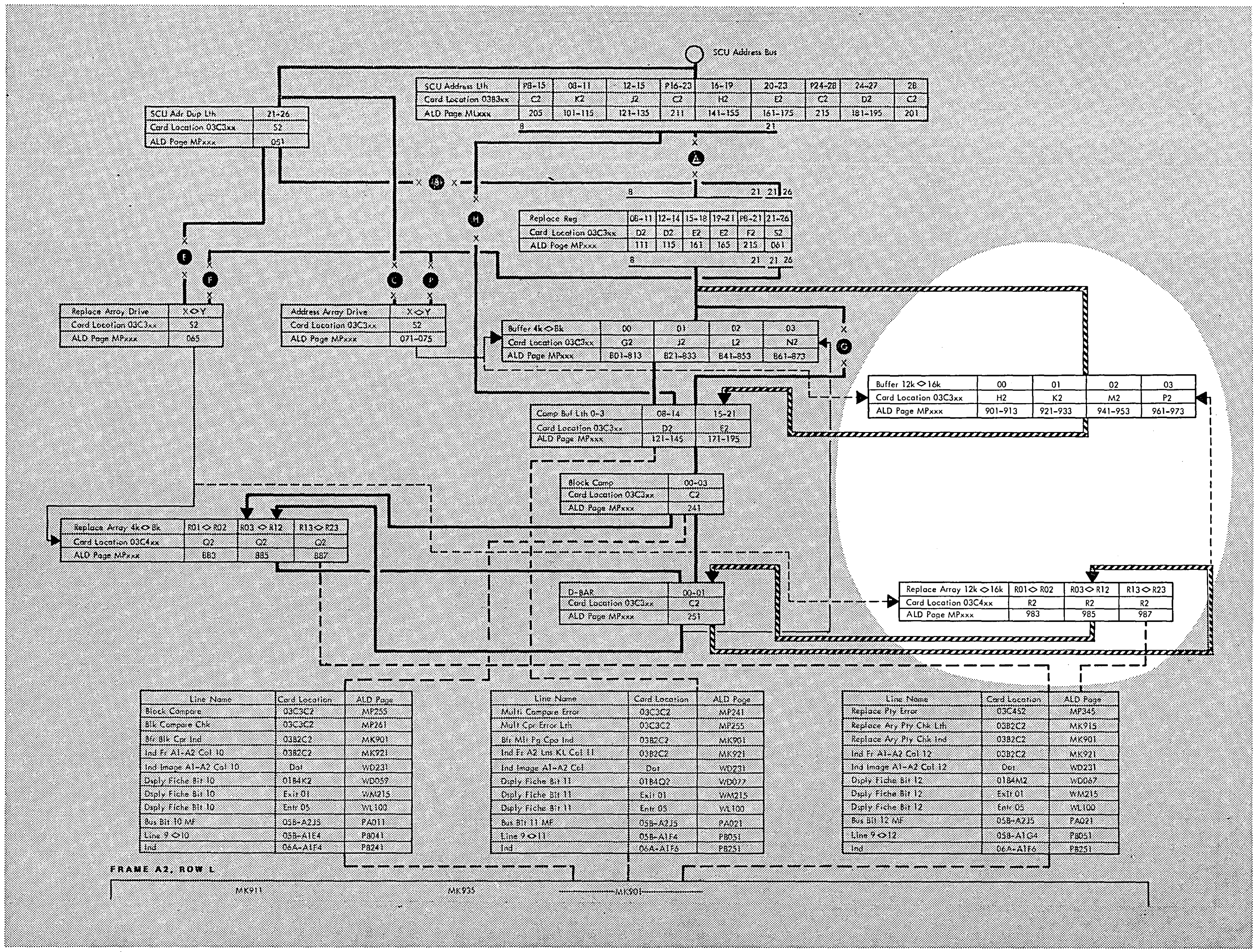


Figure 14-10. Microfiche Image A2



SCU Address Lth	P8-15	08-11	12-15	P16-23	16-19	20-23	P24-28	24-27	28
Card Location 03B3xx	C2	K2	J2	C2	H2	E2	C2	D2	C2
ALD Page MLxxx	205	101-115	121-135	211	141-155	161-175	215	181-195	201

SCU Adr Dup Lth	21-26
Card Location 03C3xx	S2
ALD Page MPxxx	051

Replace Reg	08-11	12-14	15-18	19-21	P8-21	21-26
Card Location 03C3xx	D2	D2	E2	E2	F2	S2
ALD Page MPxxx	111	115	161	165	215	061

Replace Array Drive	X◇Y
Card Location 03C3xx	S2
ALD Page MPxxx	065

Address Array Drive	X◇Y
Card Location 03C3xx	S2
ALD Page MPxxx	071-075

Buffer 4k◇8k	00	01	02	03
Card Location 03C3xx	G2	J2	L2	N2
ALD Page MPxxx	801-813	821-833	841-853	861-873

Buffer 12k◇16k	00	01	02	03
Card Location 03C3xx	H2	K2	M2	P2
ALD Page MPxxx	901-913	921-933	941-953	961-973

Comp Buf Lth 0-3	08-14	15-21
Card Location 03C3xx	D2	E2
ALD Page MPxxx	121-145	171-195

Block Comp	00-03
Card Location 03C3xx	C2
ALD Page MPxxx	241

Replace Array 4k◇8k	R01◇R02	R03◇R12	R13◇R23
Card Location 03C4xx	Q2	Q2	Q2
ALD Page MPxxx	883	885	887

D-BAR	00-01
Card Location 03C3xx	C2
ALD Page MPxxx	251

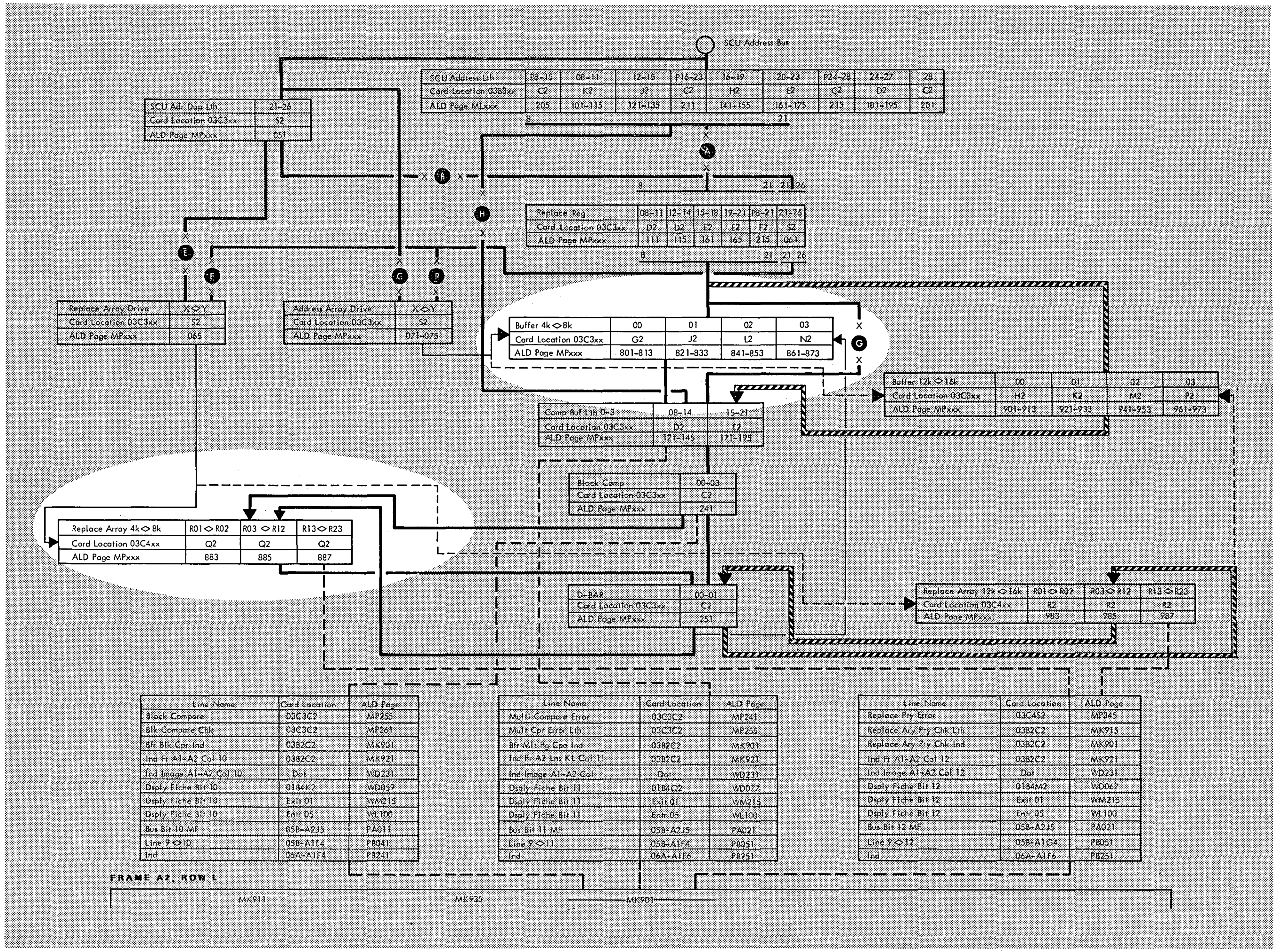
Replace Array 12k◇16k	R01◇R02	R03◇R12	R13◇R23
Card Location 03C4xx	R2	R2	R2
ALD Page MPxxx	983	985	987

Line Name	Card Location	ALD Page
Block Compare	03C3C2	MP255
Blk Compare Chk	03C3C2	MP261
Bfr Blk Cpr Ind	03B2C2	MK901
Ind Fr A1-A2 Col 10	03B2C2	MK921
Ind Image A1-A2 Col 10	Dot	WD231
Dsply Fiche Bit 10	01B4K2	WD059
Dsply Fiche Bit 10	Exit 01	WM215
Dsply Fiche Bit 10	Entr 05	WL100
Bus Bit 10 MF	05B-A2J5	PA011
Line 9◇10	05B-A1E4	PB041
Ind	06A-A1F4	PB241

Line Name	Card Location	ALD Page
Multi Compare Error	03C3C2	MP241
Mult Cpr Error Lth	03C3C2	MP255
Bfr Mlt Pg Cpa Ind	03B2C2	MK901
Ind Fr A2 Lns KL Col 11	03B2C2	MK921
Ind Image A1-A2 Col	Dot	WD231
Dsply Fiche Bit 11	01B4Q2	WD077
Dsply Fiche Bit 11	Exit 03	WM215
Dsply Fiche Bit 11	Entr 05	WL100
Bus Bit 11 MF	05B-A2J5	PA021
Line 9◇11	05B-A1F4	PB051
Ind	06A-A1F6	PB251

Line Name	Card Location	ALD Page
Replace Pty Error	03C4S2	MP345
Replace Ary Pty Chk Lth	03B2C2	MK915
Replace Ary Pty Chk Ind	03B2C2	MK901
Ind Fr A1-A2 Col 12	03B2C2	MK921
Ind Image A1-A2 Col 12	Dot	WD231
Dsply Fiche Bit 12	01B4M2	WD067
Dsply Fiche Bit 12	Exit 01	WM215
Dsply Fiche Bit 12	Entr 05	WL100
Bus Bit 12 MF	05B-A2J5	PA021
Line 9◇12	05B-A1G4	PB051
Ind	06A-A1F6	PB251

Figure 14-11. Buffer Array Checks



SCU Address Lth	P8-15	08-11	12-15	P16-23	16-19	20-23	P24-28	24-27	28
Card Location 03B3xx	C2	K2	J2	C2	H2	E2	C2	D2	C2
ALD Page MLxxx	205	101-115	121-135	211	141-155	161-175	215	181-195	201

SCU Adr Dup Lth	21-26
Card Location 03C3xx	S2
ALD Page MPxxx	051

Replace Reg	08-11	12-14	15-18	19-21	P8-21	21-26
Card Location 03C3xx	D2	D2	E2	E2	F2	S2
ALD Page MPxxx	111	115	161	165	215	061

Replace Array Drive	X<>Y
Card Location 03C3xx	S2
ALD Page MPxxx	065

Address Array Drive	X<>Y
Card Location 03C3xx	S2
ALD Page MPxxx	071-075

Buffer 4k<>8k	00	01	02	03
Card Location 03C3xx	G2	J2	L2	N2
ALD Page MPxxx	801-813	821-833	841-853	861-873

Buffer 12k<>16k	00	01	02	03
Card Location 03C3xx	H2	K2	M2	P2
ALD Page MPxxx	901-913	921-933	941-953	961-973

Comp Buf Lth 0-3	08-14	15-21
Card Location 03C3xx	D2	E2
ALD Page MPxxx	121-145	171-195

Block Comp	00-03
Card Location 03C3xx	C2
ALD Page MPxxx	241

Replace Array 4k<>8k	R01<>R02	R03<>R12	R13<>R23
Card Location 03C4xx	Q2	Q2	Q2
ALD Page MPxxx	883	885	887

D-BAR	00-01
Card Location 03C3xx	C2
ALD Page MPxxx	251

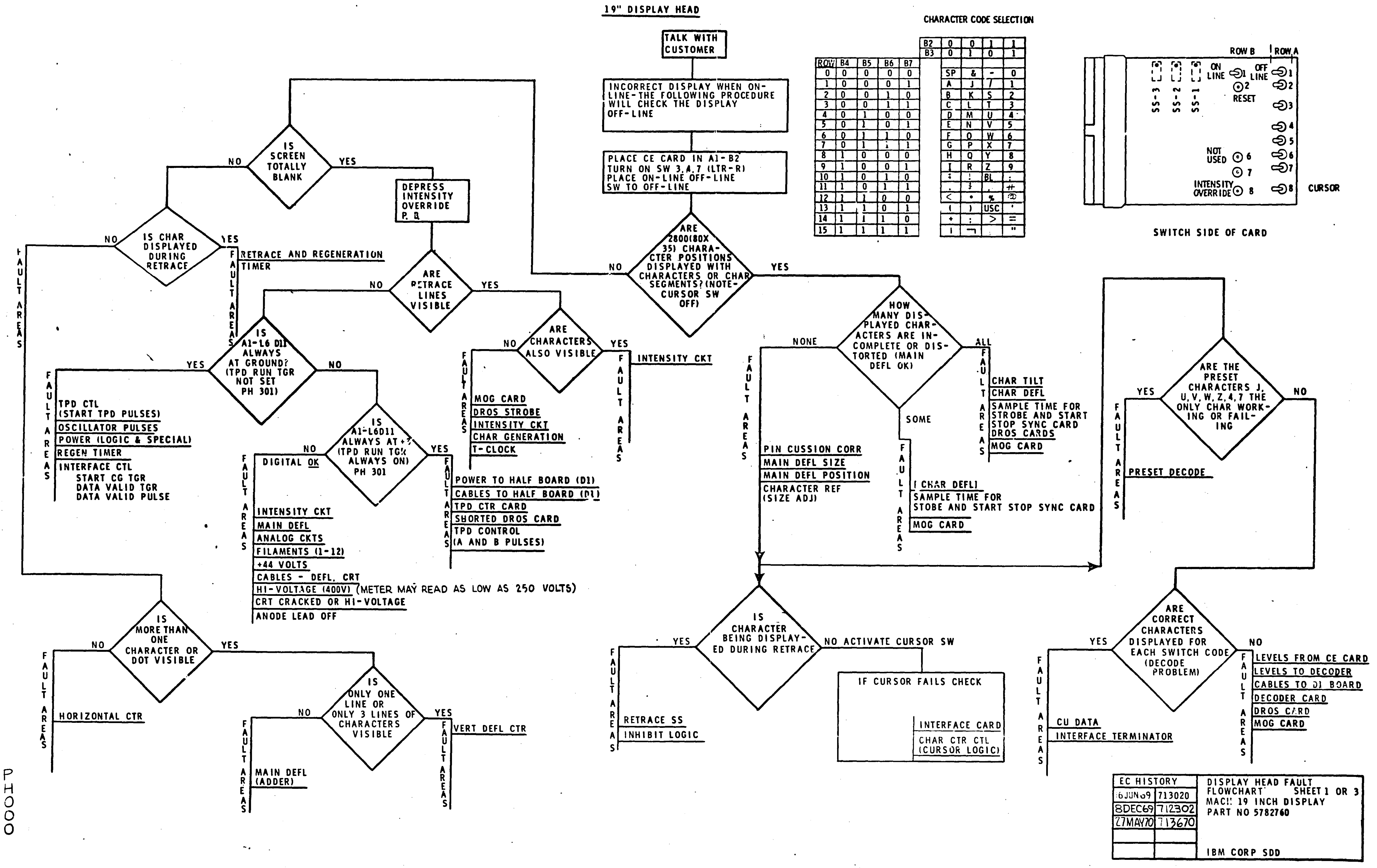
Replace Array 12k<>16k	R01<>R02	R03<>R12	R13<>R23
Card Location 03C4xx	R2	R2	R2
ALD Page MPxxx	983	985	987

Line Name	Card Location	ALD Page
Block Compare	03C3C2	MP255
Blk Compare Chk	03C3C2	MP261
Bfr Blk Cpr Ind	03B2C2	MK901
Ind Fr A1-A2 Col 10	03B2C2	MK921
Ind Image A1-A2 Col 10	Dot	WD231
Dsply Fiche Bit 10	01B4K2	WD059
Dsply Fiche Bit 10	Exit 01	WM215
Dsply Fiche Bit 10	Entr 05	WL100
Bus Bit 10 MF	05B-A2J5	PA011
Line 9<>10	05B-A1E4	PB041
Ind	06A-A1F4	PB241

Line Name	Card Location	ALD Page
Multi Compare Error	03C3C2	MP241
Mult Cpr Error Lth	03C3C2	MP255
Bfr Mlt Pg Cpo Ind	03B2C2	MK901
Ind Fr A2 Lns KL Col 11	03B2C2	MK921
Ind Image A1-A2 Col	Dot	WD231
Dsply Fiche Bit 11	01B4Q2	WD077
Dsply Fiche Bit 11	Exit 01	WM215
Dsply Fiche Bit 11	Entr 05	WL100
Bus Bit 11 MF	05B-A2J5	PA021
Line 9<>11	05B-A1F4	PB051
Ind	06A-A1F6	PB251

Line Name	Card Location	ALD Page
Replace Pty Error	03C4S2	MP345
Replace Ary Pty Chk Lth	03B2C2	MK915
Replace Ary Pty Chk Ind	03B2C2	MK901
Ind Fr A1-A2 Col 12	03B2C2	MK921
Ind Image A1-A2 Col 12	Dot	WD231
Dsply Fiche Bit 12	01B4M2	WD067
Dsply Fiche Bit 12	Exit 01	WM215
Dsply Fiche Bit 12	Entr 05	WL100
Bus Bit 12 MF	05B-A2J5	PA021
Line 9<>12	05B-A1G4	PB051
Ind	06A-A1F6	PB251

Figure 14-12. Buffer Array Checks



19" DISPLAY HEAD
TALK WITH CUSTOMER
INCORRECT DISPLAY WHEN ON-LINE-THE FOLLOWING PROCEDURE WILL CHECK THE DISPLAY OFF-LINE
PLACE CE CARD IN A1-B2 TURN ON SW 3,4,7 (LTR-R) PLACE ON-LINE OFF-LINE SW TO OFF-LINE
ARE 2800(80X 35) CHARACTER POSITIONS DISPLAYED WITH CHARACTERS OR CHAR SEGMENTS? (NOTE-CURSOR SW OFF)
NO
YES
IS SCREEN TOTALLY BLANK
NO
YES
DEPRESS INTENSITY OVERRIDE P. B.
IS CHAR DISPLAYED DURING RETRACE
NO
YES
FAULT AREAS
RETRACE AND REGENERATION TIMER
ARE RETRACE LINES VISIBLE
NO
YES
IS A1-L6 D11 ALWAYS AT GROUND? (TPD RUN TGR NOT SET PH 301)
YES
NO
TPD CTL (START TPD PULSES)
OSCILLATOR PULSES
POWER (LOGIC & SPECIAL)
REGEN TIMER
INTERFACE CTL
START CG TGR
DATA VALID TGR
DATA VALID PULSE
IS A1-L6D11 ALWAYS AT +? (TPD RUN TGR ALWAYS ON) PH 301
NO
YES
DIGITAL OK
INTENSITY CKT
MAIN DEFL
ANALOG CKTS
FILAMENTS (1-12)
+44 VOLTS
CABLES - DEFL. CRT
HI-VOLTAGE (400V) (METER MAY READ AS LOW AS 250 VOLTS)
CRT CRACKED OR HI-VOLTAGE
ANODE LEAD OFF
POWER TO HALF BOARD (D1)
CABLES TO HALF BOARD (D1)
TPD CTR CARD
SHORTED DROS CARD
TPD CONTROL (A AND B PULSES)
ARE CHARACTERS ALSO VISIBLE
NO
YES
MOG CARD
DROS STROBE
INTENSITY CKT
CHAR GENERATION
T-CLOCK
INTENSITY CKT
IS MORE THAN ONE CHARACTER OR DOT VISIBLE
NO
YES
HORIZONTAL CTR
IS ONLY ONE LINE OR ONLY 3 LINES OF CHARACTERS VISIBLE
NO
YES
MAIN DEFL (ADDER)
VERT DEFL CTR
HOW MANY DISPLAYED CHARACTERS ARE INCOMPLETE OR DISTORTED (MAIN DEFL OK)
NONE
SOME
ALL
PIN CUSSION CORR
MAIN DEFL SIZE
MAIN DEFL POSITION
CHARACTER REF (SIZE ADJ)
I CHAR DEFL1
SAMPLE TIME FOR STOBE AND START STOP SYNC CARD
MOG CARD
CHAR TILT
CHAR DEFL
SAMPLE TIME FOR STROBE AND START STOP SYNC CARD
DROS CARDS
MOG CARD
IS CHARACTER BEING DISPLAYED DURING RETRACE
YES
NO ACTIVATE CURSOR SW
RETRACE SS
INHIBIT LOGIC
IF CURSOR FAILS CHECK
INTERFACE CARD
CHAR CTR CTL (CURSOR LOGIC)
ARE THE PRESET CHARACTERS J, U, V, W, Z, 4, 7 THE ONLY CHAR WORKING OR FAILING
YES
NO
PRESET DECODE
ARE CORRECT CHARACTERS DISPLAYED FOR EACH SWITCH CODE (DECODE PROBLEM)
YES
NO
CU DATA
INTERFACE TERMINATOR
LEVELS FROM CE CARD
LEVELS TO DECODER
CABLES TO D1 BOARD
DECODER CARD
DROS CARD
MOG CARD
SS-3
SS-2
SS-1
ROW B
ROW A
ON LINE
OFF LINE
RESET
NOT USED
INTENSITY OVERRIDE
CURSOR
1
2
3
4
5
6
7
8
SWITCH SIDE OF CARD

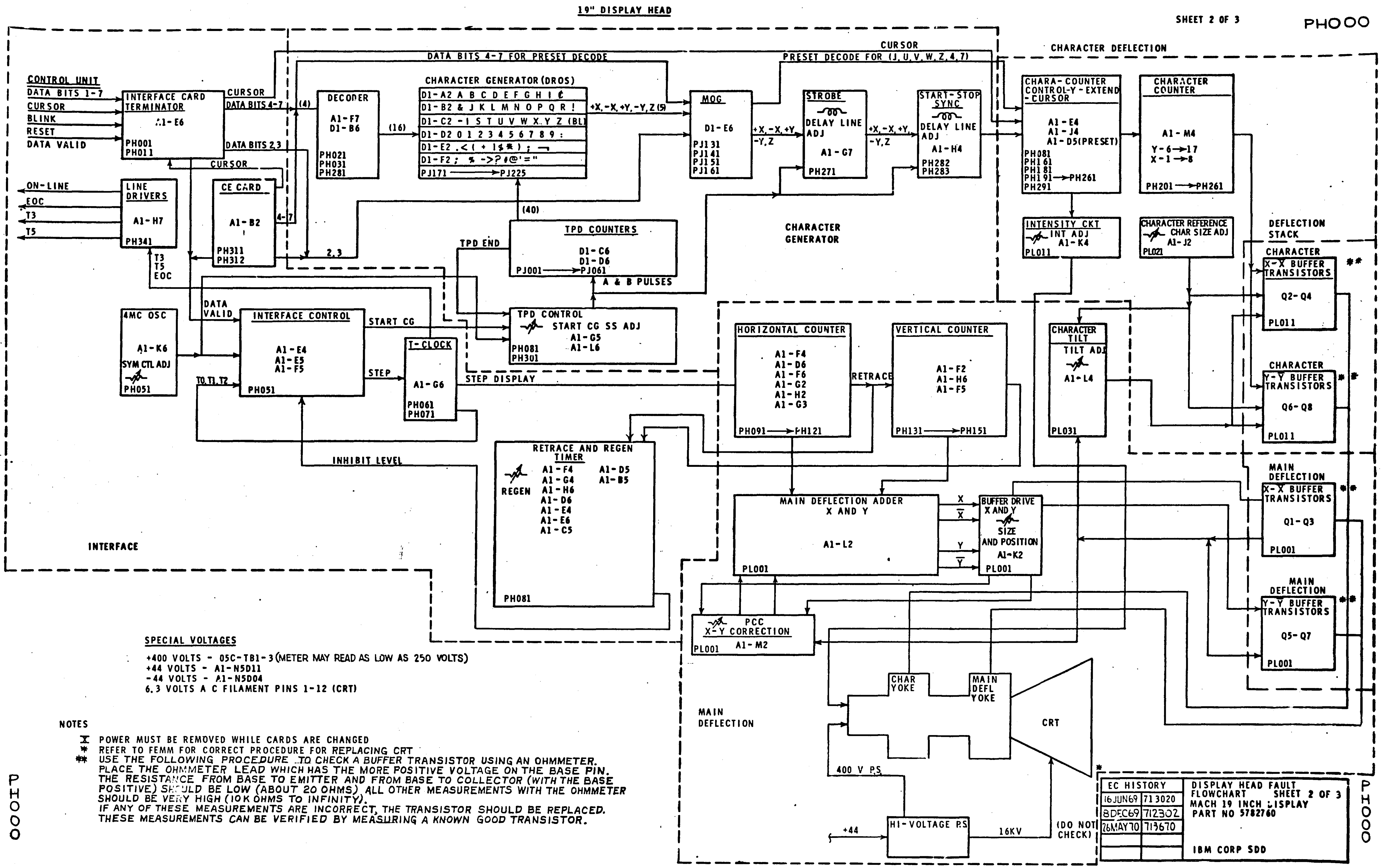
CHARACTER CODE SELECTION

ROW	B4	B5	B6	B7	B2 / B3: 0 / 0	0 / 1	1 / 0	1 / 1
0	0	0	0	0	SP	&	-	0
1	0	0	0	1	A	J	/	1
2	0	0	1	0	B	K	S	2
3	0	0	1	1	C	L	T	3
4	0	1	0	0	D	M	U	4
5	0	1	0	1	E	N	V	5
6	0	1	1	0	F	O	W	6
7	0	1	1	1	G	P	X	7
8	1	0	0	0	H	Q	Y	8
9	1	0	0	1	I	R	Z	9
10	1	0	1	0	¢	!	BL	:
11	1	0	1	1	.	$	,	#
12	1	1	0	0	<	*	%	@
13	1	1	0	1	(	)	USC	'
14	1	1	1	0	+	;	>	=
15	1	1	1	1	|	¬		"

EC HISTORY		DISPLAY HEAD FAULT FLOWCHART SHEET 1 OR 3 MACH 19 INCH DISPLAY PART NO 5782760
6JUN69	713020	
8DEC69	712302	
27MAY70	713670	
		IBM CORP SDD

PH000

Figure 18-1-1. 19" Display Head



SPECIAL VOLTAGES

+400 VOLTS - 05C-TB1-3 (METER MAY READ AS LOW AS 250 VOLTS)
+44 VOLTS - A1-N5D11
-44 VOLTS - A1-N5D04
6.3 VOLTS A C FILAMENT PINS 1-12 (CRT)

NOTES

- I POWER MUST BE REMOVED WHILE CARDS ARE CHANGED
- * REFER TO FEMM FOR CORRECT PROCEDURE FOR REPLACING CRT
- ** USE THE FOLLOWING PROCEDURE TO CHECK A BUFFER TRANSISTOR USING AN OHMMETER. PLACE THE OHMMETER LEAD WHICH HAS THE MORE POSITIVE VOLTAGE ON THE BASE PIN. THE RESISTANCE FROM BASE TO EMITTER AND FROM BASE TO COLLECTOR (WITH THE BASE POSITIVE) SHOULD BE LOW (ABOUT 20 OHMS) ALL OTHER MEASUREMENTS WITH THE OHMMETER SHOULD BE VERY HIGH (10K OHMS TO INFINITY).
  IF ANY OF THESE MEASUREMENTS ARE INCORRECT, THE TRANSISTOR SHOULD BE REPLACED. THESE MEASUREMENTS CAN BE VERIFIED BY MEASURING A KNOWN GOOD TRANSISTOR.

EC HISTORY	
16JUN69	713020
8DEC69	712302
26MAY70	713670

DISPLAY HEAD FAULT FLOWCHART SHEET 2 OF 3
MACH 19 INCH DISPLAY
PART NO 5782760

IBM CORP SDD

Figure 18-1-2. 19" Display Head

1.0 **SINGLE SHOT ADJUSTMENT:**

SET EACH SINGLE SHOT OUTPUT WITH AN OSCILLOSCOPE TO THE NOMINAL SETTING AS INDICATED IN THE FOLLOWING PROCEDURE:

1.1 CE CARD ADJUSTMENTS (SEE SHEET 1 OF 3)

1.1.1 SELECT DOUBLE CARD (CE TEST CARD) AND PLACE IN HOME LOCATION 05C-A1B2. SYNC SCOPE INTERNAL. SWITCH TO OFF LINE MODE (SWITCH 1 ROW B). THIS WILL GIVE A NEGATIVE LEVEL AT 05C-A1B2J11
NOTE: FOR THE FOLLOWING TWO SINGLE SHOT ADJUSTMENTS, IT IS NECESSARY TO USE THE ON/OFF LINE SWITCH (SWITCH 1 ROW B)

1.1.2 (CE RESET SS) WITH A POSITIVE GOING INPUT SIGNAL AT B2J10, ADJUST SS1 (ROW C) FOR A 100 ±10 USEC WIDTH NEGATIVE PULSE AT B2G04

1.1.3 (CE DATA VALID SS) WITH A POSITIVE GOING INPUT SIGNAL AT B2G04 ADJUST SS3 (ROW C) FOR A MINIMUM WIDTH NEGATIVE PULSE AT B2G02. 1.5 ±0.25 USEC

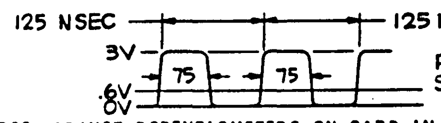
1.2 SINGLE SHOT-HOME LOCATION 05C-A1G5

1.2.1 DATA VALID SS
INSERT CARD IN SPARE LOCATION 05C-A1C7. CONNECT JUMPER BETWEEN B2J11 AND C7B03. FLIP ON LINE-OFF LINE SWITCH ON CE CARD AND ADJUST MIDDLE POTENTIOMETER ON CARD IN LOCATION C7 FOR .3 MICROSECONDS ±5% NEGATIVE OUTPUT AT C7D13. LOGIC REFERENCE PAGE PH061

1.2.2 START TPD
CONNECT JUMPER BETWEEN B2J11 AND C7D04. FLIP ON LINE-OFF LINE SWITCH ON CE CARD AND ADJUST BOTTOM POTENTIOMETER ON CARD IN LOCATION C7 FOR .8 MICROSECONDS NEGATIVE OUTPUT AT C7B13. LOGIC REFERENCE PAGE PH081

1.3 INHIBIT START CLOCK
SINGLE SHOT-HOME LOCATION 05C-A1G4
INSERT CARD IN SPARE LOCATION 05C-A1C7. CONNECT JUMPER BETWEEN B2J11 AND C7B12. CONNECT JUMPER BETWEEN C7B04 AND C7B03. FLIP ON LINE-OFF LINE SWITCH ON CE CARD AND ADJUST BOTTOM POTENTIOMETER ON CARD IN LOCATION C7 FOR 105 MICROSECONDS ±5% NEGATIVE OUTPUT AT C7D04

1.4 REGEN SS
SINGLE SHOT CARD - HOME LOCATION 05C-A1C5
INSERT CARD IN SPARE LOCATION 05C-A1C7. CONNECT JUMPER BETWEEN B2J11 AND C7B03, C7D02 AND C7D05. FLIP ON LINE-OFF LINE SWITCH ON CE CARD AND ADJUST POTENTIOMETER ON CARD IN LOCATION C7 FOR 8 MILLISECONDS ±5% NEGATIVE OUTPUT AT C7B10



1.5 REGEN SS
SINGLE SHOT-HOME LOCATION 05C-A1B5

1.5.1 USE SAME PROCEDURE AS IN SECTION 1.4

1.5.2 A AND B PULSE WIDTH AND SYMMETRY
SCOPE "A OR B" DELAY PULSE AT 05C-A1G7D02. ADJUST POTENTIOMETERS ON CARD IN LOCATION 05C-A1L6 FOR 75 NSEC POSITIVE PULSES. FOR OSCILLATOR SYMMETRY ADJUST POTENTIOMETER ON CARD IN LOCATION 05C-A1K6 FOR BALANCED WAVE SHAPE. NOTE: INITIAL DELAY SETTING MUST BE MADE ON CARD AT 05C-A1G7 AS PER PARAGRAPH 1.7.2

1.6 DELAY LINE ADJUSTMENT
JUMPER AND DELAY ARRANGEMENT FOR DROS GATING AND START STOP SYNC CARDS:

PIN	1	2-3	4-5	6-7	8-9	10-11	12
FUNCTION	INPUT	50	40	20	10	5	OUTPUT

EXAMPLE: 90 NANOSECONDS: INSERT JUMPERS 1-2, 3-4, 5-12

1.7 DROS GATING CARD - HOME LOCATION 05C-A1G7

1.7.1 THE PURPOSE OF THIS ADJUSTMENT IS TO POSITION THE "A OR B DELAYED PULSES" SUCH THAT THE OUTPUT PULSE OBTAINED FROM THE GATE HAS A CONSTANT WIDTH WITH MINIMUM SKEW

1.7.2 INITIAL DELAY SETTING--ADJUST THE DELAY LINE FOR 65 NANOSECONDS AS SHOWN IN PARAGRAPH 1.6

1.7.3 INSERT CARD IN HOME LOCATION. SELECT CHARACTER "A" (BIT 7) AND COMPARE THE +X, ¬X, +Y, -Y AND INTENSITY PULSES COMING FROM THE DROS (LOGIC PAGE PH271) WITH THE SAMPLE PULSE AT 05C-A1G7D02 (LOGIC PAGE PH271)
IF THE SAMPLE PULSE IS NARROWER THAN THE DROS PULSE, THE SAMPLE PULSE SHOULD FALL WITHIN THE DROS PULSES. IF THE SAMPLE PULSE IS WIDER THAN THE DROS PULSE, THE SAMPLE PULSE SHOULD COVER THE DROS PULSES. ADJUST THE DELAY SO THAT THIS CONDITION IS OPTIMIZED.

1.8 START - STOP SYNC CARD - HOME LOCATION 05C-A1H4

1.8.1 THE PURPOSE OF THIS ADJUSTMENT IS TO ESTABLISH THE CORRECT SAMPLE TIME OF THE X AND Y STORAGE TRIGGERS, NOMINAL 30 NANOSECONDS.

1.8.2 DELAY LINE SHALL BE SET ACCORDING TO THE PROCEDURE DESCRIBED BELOW. USE THE NOMINAL TIMING FOR INITIAL SETTING. THE DELAY TIME CAN BE VARIED WITHIN THE MINIMUM AND MAXIMUM RANGE FOR FINE ADJUSTMENT DURING THE ALIGNMENT PROCEDURE.

1.8.3 SET UP PRCEDURES:

1.8.3.1 SWITCH CE CARD TO "OFF LINE" MODE. SET DATA SWITCHES FOR CHARACTER "X" (CHARACTER CODE 1100111)

1.8.3.2 SYNC THE OSCILLOSCOPE POSITIVE AT PIN 05C-A1H5D13.

1.8.3.3 CONNECT PROBE 1 AT THE FOLLOWING POINT:
+X 05C-A1H4D07 (PH282)

1.8.3.4 CONNECT PROBE 2 TO THE "A OR B DELAY PULSE" AT 05C-A1H4D05. ADJUST THE DELAY LINE SO THAT THE POSITIVE TRANSITION OF THE "DELAYED PULSE" LAGS THE POSITIVE TRANSITION ON PROBE 1 BY 30 ±5 NANOSECONDS AT THE 20% POINT

NOTE: BEFORE REMOVING OR INSERTING CARDS
SET THE DISPLAY HEAD POWER SWITCH
TO OFF (LOCATED ON BACK OF FRAME 05)

2.0 **INTENSITY:**

2.1 THE INTENSITY CARD IS LOCATED AT 05C-A1K4. TURN INTENSITY CONTROL FULLY CCW AND THEN ADJUST "BIAS ADJUST" POTENTIOMETER ON INTENSITY CARD SO THAT THE RETRACE LINES ARE VISIBLE. NOW ADJUST THIS POTENTIOMETER UNTIL THESE LINES (OR A DOT WHICH APPEARS IN THE UPPER LEFT CORNER OF THE CRT DURING A RESET CONDITION) JUST DISAPPEAR. INTENSITY CONTROL NOW CAN BE ADJUSTED TO PRODUCE DESIRED BRIGHTNESS
NOTE: IF THE RETRACE LINES (OR RESET DOT) CANNOT BE MADE VISIBLE WITH THE BIAS ADJUST POTENTIOMETER. TURN THE POTENTIOMETER COUNTER-CLOCKWISE TO STOP. THEN ADJUST THE INTENSITY CONTROL FOR DESIRED BRIGHTNESS

2.2 FOCUS
THE SPOT QUALITY OF THIS DISPLAY IS SUCH THAT NO OPERATOR FOCUS IS REQUIRED

CE FOCUS ADJUSTMENT IS PROVIDED. CRT GRID 4 (SINGLE LEAD #5 WITH RING TONGUE TERM) MAY BE RETURNED TO EITHER +400 ±15% VOLTS (05C-TB1-3), 267 VOLTS ±15% (05C-TB1-4), 133 VOLTS ±15% (05C-TB1-5) OR GROUND (05C-TB1-6). SELECT WHICHEVER VOLTAGE GIVES BETTER FOCUS OVER THE ENTIRE DISPLAYED IMAGE

3.0 **DISPLAY ADJUSTMENT AND MEASUREMENT:**

3.1 DEFLECTION YOKE ADJUSTMENT

3.1.1 POSITION MASK AGAINST FRONT OF DISPLAY FRAME AND SECURE WITH TAPE. REFER TO FEMM FOR BEZEL AND FRAME REMOVAL

3.1.2 LOOSEN THE YOKE CLAMP

3.1.3 PUSH THE YOKE FORWARD SO THAT IT SEATS FIRMLY AGAINST THE CRT BELL

3.1.4 ROTATE THE YOKE SO THAT THE DISPLAY FRAME IS PARALLEL WITH THE GUIDELINES OF MASK, P/N 5787159

3.1.5 TIGHTEN THE YOKE CLAMP SO AS TO HOLD THE YOKE FIRMLY IN POSITION

3.2 ADJUST THE IMAGE TO THE VALUES GIVEN BELOW, WITH THE FOUR POTS. LOCATED ON 05C-A1K2. THE TOP TWO CONTROLS ADJUST HEIGHT AND VERTICAL CENTERING WHILE THE BOTTOM TWO ADJUST WIDTH AND HORIZONTAL CENTERING. CENTER THE DISPLAY ON THE TUBE WITH MASK A AS A GUIDE. USE A FULL PATTERN OF "H" 'S FOR WIDTH ADJUSTMENT, AND A FULL PATTERN FOR "E" 'S FOR HEIGHT ADJUSTMENT

3.2.1 X-Y PIN CUSHION AND KEYSTONE CORRECTION
IMAGE PIN CUSHIONING AND KEYSTONING ARE CORRECTED BY ADJ THE FOUR POTS ON THE CARD IN LOC 05C-A1M2 THE TOP AND BOTTOM POTS CONTROL THE TOP/BOTTOM AND LEFT/RIGHT PIN CUSHIONING RESPECTIVELY WHILE THE SECOND AND THIRD POTS CONTROL THE LEFT/RIGHT AND TOP/BOTTOM WIDTH RATIOS RESPECTIVELY

3.2.2 THE ADJ SHALL BE MADE SUCH THAT THE BOUNDARIES OF THE DISPLAY IMAGE SHALL FALL WITHIN THE GUIDELINES OF THE MASK

3.3 BOTH IMAGE SIZE AND IMAGE CENTERING SHALL BE CAPABLE OF MOVING ±.500 (12,7) FROM THE NOMINAL SETTING AFTER BEING ADJUSTED TO ITS PROPER DISCERNIBLE SETTING

3.4 IMAGE DRIFT

3.4.1 SHORT TERM
THERE WILL BE NO DISCERNIBLE MOVEMENT OF ANY POINT OF A CHARACTER FROM A REFERENCE CROSSHAIR OVER A PERIOD OF 70 SECONDS WHEN VIEWED FROM A MINIMUM DISTANCE OF 18.000 INCHES

4.0 CHARACTER ADJUSTMENTS

4.1 OVERDRIVE

4.1.1 WITH A FULL DISPLAY OF "B" 'S ADJUST THE OVERDRIVE POT LOCATED IN 05C-A1L2 UNTIL THE BOTTOM LINES OF THE CHARACTER JUST CLOSE. CHARACTERS IN THE FIRST COLUMN WILL NOT BE AFFECTED BY THE OVERDRIVE POTENTIOMETER

4.2 CHARACTER SIZE
THE CHARACTER SIZE IS ADJUSTABLE. THE POTENTIOMETER IN LOCATION 01A-A1J2 WILL CHANGE THE SIZE OF THE HEIGHT AND WIDTH SIMULTANEOUSLY (THE RATIO OF THE HEIGHT/WIDTH IS CONSTANT) THE SETTING OF THIS POTENTIOMETER CAN BEST BE DONE BY ENTERING VARIOUS CHARACTERS ON THE SCREEN AND THEN ADJUSTING FOR THE BEST COMPROMISE BETWEEN SEPERATION OF CHARACTERS AND CONTINUITY BETWEEN ADJACENT CHARACTERS

4.3 CHARACTER TILT

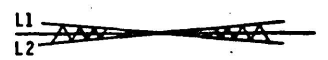
4.3.1 CHARACTER TILT CORRECTION CARD IS LOCATED AT 05C-A1L4
CAUTION: THIS CARD MUST NOT BE INSERTED OR REMOVED WITH POWER ON.

4.3.2 ADJUSTMENT OF CHARACTER CORRECTION CARD IS AS FOLLOWS:

4.3.2.1 SYNC SCOPE ONE PER FRAME (05C-J2D13)

4.3.2.2 ADJUST THE POT. UNTIL THE SIGNAL AT L4J09 HAS AN AVERAGE VALUE OF 6 VOLTS ±1 VOLT. AMPLITUDE OF THIS SIGNAL IS APPROXIMATELY 4 VOLTS PEAK TO PEAK. THIS IS A COARSE ADJUSTMENT

4.3.2.3 ADJUST THE POT. UNTIL THE SIGNAL AT L4G13 IS SYMMETRICAL. SEE BELOW



4.3.2.4 FOR FINAL ADJUSTMENT, OBSERVE A DISPLAY OF CHARACTERS SUCH AS "P" AND ADJUST THE POT. FOR MINIMUM TILT IN ALL FOUR CORNERS

EC HISTORY		DISPLAY HEAD FAULT FLOWCHART SHEET 3 OF 3
16JUN69	713020	
8DEC69	712302	MACH 19 INCH DISPLAY
26MAY70	713670	
		PART NO 5782760
		IBM CORP SDD

PHOOO

PHOOO

Figure 18-1-3. 19" Display Head Adjustments

SR23-3475-1

System/370 Model 165 Phase 1 Supplementary Course Material Printed in U.S.A. SR23-3475-1

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
[U.S.A. only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]