AMDASM FILES

Am29116 & Am29203

& misc.

PREFACE

The new Am2900 family members which require microprogramming, the 16-bit bipolar microprocessor, the Am29116, and the new registered-arithmetic unit (RALU), the Am29203, are supported by the AmSYS29/10 development system, TRACE29 and all of the bipolar support software.

The Am29116 has attracted attention due to the 16-bit instruction field and the two-microcycle immediate data instructions, as well as the intended speed of operation. The Am29203 adds BCD operations to a chip (the Am2903A) which already supports binary (2's complement) operations and provides aides for floating point arithmetic.

In both cases, the richness of the instruction set provides a feature and a problem. Both parts require some standard set of mnemonics to facilitate the exchange of microroutines between various users and between AMD customer support and the customer. Both parts have enough instructions to require the creation of some "master" AMDASM definition file, .DEF, to reduce the effort required to create microprograms.

The Am29116 instruction set, along with those of several other parts (that may or may not be present in a given controller design), is given such a master file in "CONTROLR.DEF". A complete controller design, created with this file as a reference, is provided in "DISKCTLR.DEF". Source-code files (.SRC) are provided for both of the definition files. The mnemonics for the Am29116 were created directly from the data sheet.

The Am29203 instruction set, along with instruction sets for the other RALUs, the Am2903 and the Am2901, and several other parts (that may or may not be present in a given CPU design), is given a master file in "AM29CPU.DEF". A sample code source file is also provided. The mnemonics for the Am29203 were selected for readability. Duplicate mnemonics for the Am2901 and Am2903 were "tagged" with ".01" and ".03" respectively.

The purpose and intent of these files is to reduce the effort required for the designer either to become familiar with the individual parts or to create microcode for a design based around one or more of them. The user of these files is expected to edit them into a usable form for a particular application.

[illegible]

Donnamaie E. White, Ph.D.
Manager
Advanced Micro Devices, Inc.
Customer Education Center
2/22/82

DISK.DOC

DISK.DOC

This diskette should contain 6 additional files:

CONTROLR.DEF	The "master" .DEF file for the Am29116 + misc other parts in comma-positional notation.
CONTROLR.SRC	A test SRC file (partial microword).
DISKCTLR.DEF	The .DEF file for the Am29116-Am9520 (BEP) based disk controller that is discussed in the application note, "A High Performance Intelligent Disk Controller" by O. Tabler and B. Kitson. This file is in DEF statement-overlay format.
DISKCTLR.SRC	The source code for the disk controller. (This file assembles to a 92K P2L file.)
Am29CPU.DEF	The "MASTER" .DEF file for RALU-based designs (Am2901-2903 or 29203) RALUs + misc other parts in comma-positional notation.
Am29203.SRC	Sample source code for an Am29203 CPU (Am2910 sequencer). (Uses Am29CPU.DEF)

These AMDASM.DEF files are provided as a service to AmSYS29 customers. They are intended to provide a basis from which the customer may select those equates, subs and definitions appropriate to a given design problem. It is intended that the customer edit a copy of one or more .DEF files to produce the desired .DEF file.

The .SRC files are provided as a reference. The microword will probably not match the customer's requirements.

CONTROLR.DEF

Originally created as a master file for study, it has been edited into one application,"DISKCTLR.DEF". It is recommended that the user compare these two .DEF files for similarities and differences.

The file includes a number of parts which may or may not appear in a controller. The first page after the WORD declaration provides an effective index to the file. The various EQU groups of the Am29116 instructions are indexed ([i]) for cross reference by the DEF statements. Each instruction type has its own DEF statement.

The Am2910 instructions, without any of the additional control lines ($\overline{OE}y$, $\overline{RLD}$, $\overline{CCEN}$) are presented with a SUB statement. The Am2910, a 3-bit conditional MUX field, and a 12-bit variable address field are part of each of the following DEF statements. By editing the SUB statement, all of the other DEF statements can be altered.

This means that any of the three fields can be altered or even eliminated by editing one statement.

Control lines

Several of the control lines for the Am29116 are presented in a second SUB statement. Note that not all of the Am29116 lines appear (the T_i lines, for example, were not input).

This manner of definition requires comma positional substitution in the source file. These lines can be edited into DEF statements and the SUB deleted to convert the control line specification to overlays. The required usage and personal preference are the deciding factors in the approach.

Instructions

Pages 8-20 are the Am29116 instructions. Each class is defined by first defining the mnemonics and then the DEF statement. The mnemonics are taken directly from the Am29116 data sheet. For example, SOR, the single operand-RAM instruction format (page 5 of the Am29116 data sheet) has a choice of word-byte mode ([m]), 1 of 4 op codes ([1]), 1 of 10 source-destination pair selects ([2]), and requires a RAM register address ([R]).

Am2914

The instruction set only is defined.

Am2940

The instruction set only is defined. The instruction set is flagged by ".40" to avoid duplicate mnemonics with the Am2942.

Am2942

The instruction set only is defined.

Am2904

The Am2904 instruction set is very powerful and extensive. Some ideas for mnemonics are given for shift control and various other commands and some sample DEF statements are included.

The user must define the Am2904 by defining only what is required for a given design. There are no standard mnemonics.

Am2925

Equates for cycle length codes plus place-holders (incomplete definition) for control lines are provided. Again, DEF statements may be used instead.

Am2950/51

A comment only.

CONTROLR.SRC

This source file is for debug of the preceding .DEF file. It was prepared primarily for study.

Page 3 provides calls to the single operand instructions. SONR requires an immediate data word and it is provided via a free format (FF) statement.

Example:

```
  SOR    ,,, W, MOVE, SORA, R10

  SONR   ,,, W, MOVE, SOI, NRS
  FF     H#E, 15X, H#0123, B#01000, 24X
```

Note the commas. This can be avoided by the use of overlays.

(Note that this is an incomplete microword.)

CONTROLR.DEF + CONTROLR.SRC MICROWORD

2910 INSTR	COND MUX	BR ADDR	29116 CTLs	OPEN		
4	3	12	16	24	=	64 bits

Note on address fields: To avoid problems, use $ as the attribute in address field for AMDASM, i.e. 12V$X, 6V$X, etc.

Note on microword: This incomplete microword was laid out for study of the Am29116 and will not necessarily match an actual microword format for a controller. The 24 don't cares would be replaced by other controls and the microword might be larger than 64 bits.

DISKCTLR.DEF

An AMD application note is being prepared which discusses a high performance disk controller. The .DEF file for the controller was created by editing the "master" file.

The Am29116 instructions were left basically intact; the major change was the deletion of the Am2910 and ZZZZZ SUB statements. The length of the microword was also increased from 64 to 80 bits. A global substitution (under ED, the editor) replaced Am2910 by nothing and ZZZZZ by 64X.

The instruction sets for all other parts were deleted. Beginning on page 19, EQUs and DEFs were added according to the design requirements. The choice here was to use overlay in the source file; therefore, all Am2910 instructions were defined as DEF statements rather than equates.

Immediate Operand

The study file used FF statements in the .SRC file to provide data for immediate data operand instructions. Here, IMME is a DEF statement which can be overlayed with other DEF statements to build a microword.

Control Lines

Control lines for the Am29116 are defined in DEF statements for overlay.

Am2910

The Am2910 command›are defined in DEF statements. Note the use of JS, JP, RTN, IF and IF NOT which have been added to the basic 16 instructions.

Misc Control Signals

All of the remaining control signals appear as DEF statements for overlay in the source file.

The Microword - DISKCTLR.DEF

Bit Position	#	Function
0-15	16	Am29116 Instruction
0-15	16	Immediate Data Operand
16-19	4	CT Multiplexer Control (T_i Lines)
20	1	Status Register Enable
21	1	Ouput Enable - Yi
22	1	Instruction Enable
23	1	Data Latch (D-I-Latch) Enable
24-27	4	Am2910 Instruction
28-37	10	Branch Address
38-42	5	Condition Code Selections
43	1	Parity
44	1	Address Mark Control
45	1	Memory Bus from Drive Control Bus
46	1	Memory Bus from Translate PROM
47	1	Memory Bus from 9403AS
48	1	Memory Bus from Am29116
49	1	Memory Bus from Am9520 - LSB
50	1	Memory Bus from Am9520 - MSB
51	1	(Disk) Bus Direction out from Controller
52	1	Memory Bus to 9403AS
53	1	Memory Bus to Am29116
54	1	Memory Bus to Am9520 - LSB
55	1	Memory Bus to Am9520 - MSB

Bit Position	#	Function
56	1	Memory Bus to Am9520 - Control Information
57	1	Clock Enable Am9520 to Lower-Byte Bus Int.
58	1	Clock Enable Memory Bus to Am9520 Transfer
59	1	Clock Pulse (Actual Waveform) for Am9520
60	1	Command Request
61	1	Input Serial Data to 9403AS
62-63	2	Jump Indirect Am29116 Register
62-63	2	No Indirect Jump
64	1	Memory Access
65	1	Memory Address
66	1	Memory Write
67	1	Output Serial Data from 9403AS
68	1	Parameter Enable
69	1	Set 9520 P Bits from 9520 PM Bits
70	1	Parallel Fetch from 9403AS
71	1	Parallel Load into 9403AS
72	1	Parameter Request
73	1	Read Gate
74	1	Reset FIFO
75	1	Select/Attention Strobe
76	1	Write Gate
77-79	3	Translate Prefix

Example Source Code Lines

```
     SOR  W, MOVE, SORY, R4  & NODLE  & NOIEN
\    & OEY  & NOSRE  & MADR  & NOJMPI  & CONT
;    MOVE R4 TO MAR

     SOR  W, MOVE, SOZR, R0  & NODLE  & IEN
\    & NOOEY  & NOSRE  & NOJMPI  & RTN
;    Load 0 into R0 and return
```

DISKCTLR.SRC

This file, a very large one, was prepared for the AMD application note, "A High Performance Disk Controller". The code is written using overlaid DEF statements with positional substitutions minimized (primarily appears in the Am29116 instructions). The code is commented.

Am29CPU.DEF

The original AmSYS29 software included a library file, AM2900.LIB, which provided the user with a source of predefined mnemonics for various parts. The AM29CPU.DEF file is the latest descendant of that original file.

The AM29CPU.DEF file provides mnemonics for a number of parts considered to be likely to appear in a CPU design, as opposed to a controller design. This includes the Am2901, Am2903 and Am29203 RALUs. Flags are used to differentiate duplicate mnemonics. (The Am29203 is given priority.)

The user must edit a version of this file to complete a .DEF file for a particular application. Most of the equate groups are indexed ([i]) for cross reference from DEF and SUB statements.

Am2901

The oldest RALU, the function and destination fields are flagged by ".01". Example SUB statements are included following the equates. The mnemonics are from the data sheet.

Am2914

Only the instruction set is included.

Am2930

Only the instruction set is included.

Am2932

Only the instruction set is included. The instruction set is flagged by ".32" to differentiate it from that of the Am2930.

Am2940

Only the instruction set is included.

Am2902 - Am29203

The instruction set is flagged by ".03" in cases where it duplicates that of the Am29203. Certain equate groups (source select and destination for example) are the same for the two parts. The differences are in the function set and the special functions. While many of the equates in these actually were the same, there were enough differences to flag that separate definitions were felt to be clearer.

Expanded Memory

The Am2903 and Am29203 have expandable registers. The Am29705/29707 2-port memories allow groups of 16 registers to be added. This changes the microword source fields. Example mnemonics are provided for a particular diagram.

Register [R]

All RALUs use 4-bit addresses for the on-board registers.

Carry

A few not-inclusive definitions are provided. This is not programming for the Am2904.

Am2910 - Am29811

The instruction set for the Am2910 is provided. The Am29811 differs only in the last mnemonic, substituting JP for TWB. An example SUB statement is provided.

Am2925

Cycle length codes and an indication of control lines are provided. The control line definitions are incomplete.

Am2904

As in the CONTROLR.DEF file -- incomplete as far as total Am2904 capabilities. Intended to provide ideas. The customer must create his or her own definitions for equates and DEF statements. There are no standard mnemonics.

DEF Statements

Comma positional within the part and overlay are used to create the microword. The DEF statements are intended to demonstrate typical ALU and sequencer definitions. The balance of the microword is undefined.

Note the use of commented default line plus the index reference [i]. The reference identifies equate mnemonics which may be substituted in a given variable field in place of the default value.

DEF statements for the Am2904 are given as well as statements specific to a particular .SRC file. This was done for demonstration -- one DEF file may be used with more than one SRC file. The .DEF file is then the main or master mnemonic reference for a set of source programs.

Am29203.SRC

This file, developed from one that was originally prepared for the Am2903 (AM2903.SRC) was prepared to demonstrate how to program the Am29203.

The microword is in a conceptual, standard format:

2 9 1 0				2	9	2	0	3				2 9 0 4		
2910 INSTR	COND MUX	BR ADDR	SOURCE A	FUNC L	DEST U	CARRY IN	Ra ADDR	Rd ADDR	Ien	Oey en		Shift INSTR	SE en	
4	3	12	3	4	4	2	4	4	1	1	14x	4	1	3x

47 bits + 17 don't cares = 64 bit word

Sample Code Lines

```
AM2910  RPCT,  ,LB & AM29203  ,SPECL, MULT, NOCARRY, R1, R0

AM2910  & AM29202  RAMAB,  ,RAM,,,R15,,OENDIS

AM2910  & AM29203  ,ADD,RAMQUPA,NOCARRY,R2,R2  & SHIFT SUR2Q2
```

Since the code was prepared for study, it is more detailed than might otherwise be necessary; i.e., defaults are resubstituted occasionally, etc. Improvements in readability could be obtained by the use of specialized DEF statements and increased use of overlays.

CONTROLR .DEF & .SRC

```
 ;
 ;
 ;
 ;
 ; This file was created as a master file from which the user can create
 ; a .DEF file for a particular application.  Documentation on the Am29116
 ; instruction set can be found in the data sheet and in "The Am29116",
 ; a CUSTOMER EDUCATION CENTER publication, revised to include this file
 ; and its companion, CONTROLR.SRC, a test file.
 ;
 ;
 ; The file DISKCTLR.DEF was created by editing this file; DISKCTLR.SRC
 ; provides an example of disk controller microroutines.
 ;
 ; This file was created for use in AMD CUSTOMER EDUCATION CENTER seminars.
 ;
 ; Advanced Micro Devices reserves the right to make changes in its product
 ; without notice in order to improve design or performance characteristics.
 ; The company assumes no responsibility for the use of any circuits or
 ; programs described herein.
 ;
 ; Am29116 Mnemonics Copyright (c) 1982 Advanced Micro Devices, Inc.
```

```
;
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
WORD   64              ; ASSUMED LAYOUT - CHANGE AS NEEDED
;
; This file contains the EQUs and DEFs for a "typical" Am29116 controller
;
; CREATED APRIL 21, 1981
; UPDATED JAN 12, 1982 DEW
;
; INDEX:
;                     BYTE-WORD MODE SELECT [M]
;                     NUMBER [N]
;                     RAM REGISTERS [R]
; Parts included are:  Am2910 (SUB STATEMENT)
;                      Am29116 - ALL VALID COMBINATIONS
;                      Am2914
;                      Am2940
;                      Am2942
;                      Am2904 - PARTIAL ONLY! (2**22 POSSIBLE VARIATIONS)
;                      Am2925 - CYCLE SELECT, OTHER CONTROLS DRAFTED ONLY
;                      Am2950
;
;      INDEX TO Am29116 INSTRUCTIONS - [i] REFERS TO ALLOWED MNEMONICS GROUP
;
;            SINGLE OPERAND [1], [2], [3], [4]
;            TWO OPERAND    [5], [6], [7], [8]
;            SHIFT          [9], [10], [11]
;            ROTATE         [12], [13], [14]
;            BIT-ORIENTED   [15], [16], [17]
;            ROTATE & MERGE [18]
;            ROTATE & COMPR [19]
;            PRIORITIZE     [20], [21], [22], [23], [24], [25]
;            CYCLIC REDUNDANCY CHECKS
;            NOOP
;            STATUS         [26], [27]
;            TEST STATUS    [CT]
;
```

```
 ; ********************************************
 ; GENERAL MNEMONICS
 ; ********************************************
 ;
 ; ***************************
 ; BYTE - WORD MODE SELECT [M]
 ; ***************************
 ;
B:              EQU            1B#0    ; BYTE MODE
W:              EQU            1B#1    ; WORD MODE
 ;
 ; ********************************************
 ; N SELECT [N]          constant for Am29116
 ; ********************************************
 ;
N0:             EQU            H#0     ; 0
N1:             EQU            H#1     ;
N2:             EQU            H#2     ;
N3:             EQU            H#3     ;
N4:             EQU            H#4     ;
N5:             EQU            H#5     ;
N6:             EQU            H#6     ;
N7:             EQU            H#7     ;
N8:             EQU            H#8     ;
N9:             EQU            H#9     ;
NA:             EQU            H#A     ;
NB:             EQU            H#B     ;
NC:             EQU            H#C     ;
ND:             EQU            H#D     ;
NE:             EQU            H#E     ;
NF:             EQU            H#F     ;
```

```
;
; ***********************************************
; 32 RAM REGISTERS [R]
; ***********************************************
;
R0:             EQU             5D#0    ; 00000
R1:             EQU             5D#1    ;
R2:             EQU             5D#2    ;
R3:             EQU             5D#3    ;
R4:             EQU             5D#4    ;
R5:             EQU             5D#5    ;
R6:             EQU             5D#6    ;
R7:             EQU             5D#7    ;
R8:             EQU             5D#8    ;
R9:             EQU             5D#9    ;
R10:            EQU             5D#10   ;
R11:            EQU             5D#11   ;
R12:            EQU             5D#12   ;
R13:            EQU             5D#13   ;
R14:            EQU             5D#14   ;
R15:            EQU             5D#15   ;
R16:            EQU             5D#16   ;
R17:            EQU             5D#17   ;
R18:            EQU             5D#18   ;
R19:            EQU             5D#19   ;
R20:            EQU             5D#20   ;
R21:            EQU             5D#21   ;
R22:            EQU             5D#22   ;
R23:            EQU             5D#23   ;
R24:            EQU             5D#24   ;
R25:            EQU             5D#25   ;
R26:            EQU             5D#26   ;
R27:            EQU             5D#27   ;
R28:            EQU             5D#28   ;
R29:            EQU             5D#29   ;
R30:            EQU             5D#30   ;
R31:            EQU             5D#31   ;
```

```
; ************************
; Am2910 INSTRUCTION SET
; Microprogram Controller
; ************************
;
JZ:     EQU     H#0     ; JUMP ZERO - RESET
CJS:    EQU     H#1     ; COND JUMP SUBROUTINE - PIPELINE
JMAP:   EQU     H#2     ; JUMP MAP (DECODE)
CJP:    EQU     H#3     ; COND JUMP PIPELINE
PUSH:   EQU     H#4     ; PUSH, COND LOAD CNTR & CONTINUE
JSRP:   EQU     H#5     ; COND JUMP SUBROUTINE - REGISTER OR PIPELINE
CJV:    EQU     H#6     ; COND JUMP VECTOR MAP
JRP:    EQU     H#7     ; COND JUMP REGISTER OR PIPELINE
RFCT:   EQU     H#8     ; REPEAT LOOP, ADDR ON STACK, COUNT DOWN COUNTER
RPCT:   EQU     H#9     ; REPEAT LOOP, ADDR IN PIPELINE, COUNT DOWN COUNTER
CRTN:   EQU     H#A     ; COND RETURN FROM SUBROUTINE
CJPP:   EQU     H#B     ; COND JUMP PIPELINE AND POP STACK
LDCT:   EQU     H#C     ; LOAD COUNTER AND CONTINUE
LOOP:   EQU     H#D     ; REPEAT LOOP UNTIL TEST TRUE; ADDR ON STACK
CONT:   EQU     H#E     ; CONTINUE
TWB:    EQU     H#F     ; DEAD-MAN TIMER, REPEAT LOOP, ADDR ON STACK, FAIL
;                         ADDR IN PIPELINE, COUNT-DOWN COUNTER (3-WAY)
;
; ************************************************************************
AM2910: SUB    4VH#E,3VX,12V$X    ; AM2910-COND MUX-BRANCH/COUNTER FIELDS
; ************************************************************************
;
; THE AM2910 FIELDS ARE GROUPED AS A SUB STATEMENT WHICH APPEARS IN ALL Am29116
; DEF STATEMENTS.  THE ACTUAL FIELDS REQUIRED CAN BE ALTERED AS NEEDED
;
;
```

```
; * * * * * * * * * * * * * * * * *
; Am29116 CONTROL LINES
; 16-Bit Bipolar Microprocessor
; * * * * * * * * * * * * * * * * *
;
OEYEN:          EQU             B#0     ; Y BUS ENABLE
OEYDIS:         EQU             B#1
;
DLE.EN:         EQU             B#1     ; DATA LATCH ENABLE
DLE.DIS:        EQU             B#0
;
OETEN:          EQU             B#1     ; T BUS ENABLE
OETDIS:         EQU             B#0
;
SRE.EN:         EQU             B#0     ; STATUS REGISTER ENABLE
SRE.DIS:        EQU             B#1
;
IEN:            EQU             B#0     ; INSTRUCTION ENABLE
IDIS:           EQU             B#1
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
ZZZZZ:  SUB     1VB#0, 1VB#1, 1VB#0, 1VB#0, 1VB#0, 24X
;
;               OEYEN  DLE.EN OETDIS SRE.EN IEN    MISC
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; THE "SLEEPER" CONTAINS THE Am29116 CONTROL LINES AND 24 DON'T CARE POSITIONS
; WHICH THE USER CAN DEFINE TO BE WHATEVER NEEDED TO COMPLETE THE MICROWORD
; THE "SLEEPER" APPEARS IN ALL Am29116 INSTRUCTION DEF STATEMENTS
;
```

```
AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4
* * * * AM29116.DEF FILE - D. E. WHITE - JANUARY 12, 1982 *

 ; ****************************
 ; SINGLE OPERAND INSTRUCTIONS
 ; ****************************
 ;
 ; OPCODES [1]
 ;
MOVE:           EQU             H#C     ; 1100 MOVE
COMP:           EQU             H#D     ; 1101 COMP
INC:            EQU             H#E     ; 1110 INC     INCREMENT
NEG:            EQU             H#F     ; 1111 NEG     INCREMENT COMP
 ;
 ; SOURCE-DESTINATION SELECT [2]
 ;
SORA:           EQU             H#0     ; RAM   ACC
SORY:           EQU             H#2     ; RAM   Y BUS
SORS:           EQU             H#3     ; RAM   STATUS
SOAR:           EQU             H#4     ; ACC   RAM
SODR:           EQU             H#6     ; D     RAM
SOIR:           EQU             H#7     ; I     RAM
SOZR:           EQU             H#8     ; 0     RAM
SOZER:          EQU             H#9     ; D(OE) RAM
SOSER:          EQU             H#A     ; D(SE) RAM
SORR:           EQU             H#B     ; RAM   RAM
 ;
 ; ************************************************************
SOR: DEF AM2910,1V,  B#10,4V,    4V,         5V%, ZZZZZ;SINGLE OPERAND RAM
 ;
 ;              MODE,QUAD,OPCODE,SOURCE-DEST,REGISTER
 ;              [M]        [1]      [2]        [R]
 ; ************************************************************
```

```
;
; SOURCE (R/S) [3]
;
SOA:            EQU             H#4     ; ACC
SOD:            EQU             H#6     ; D
SOI:            EQU             H#7     ; I
SOZ:            EQU             H#8     ; 0
SOZE:           EQU             H#9     ; D(0E)
SOSE:           EQU             H#A     ; D(SE)
;
; DESTINATION [4]
;
NRY:            EQU             D#0     ; Y BUS
NRA:            EQU             D#1     ; ACC
NRS:            EQU             D#4     ; STATUS
NRAS:           EQU             D#5     ; ACC,STATUS
;
; ***************************************************************
SONR: DEF AM2910,1V,  B#11,4V,    4V,    5V%, ZZZZZ ; SINGLE OPERAND NON-RAM
;
;               MODE,QUAD,OPCODE,SOURCE,DESTINATION
;               [M]         [1]    [3]      [4]
; ***************************************************************
```

```
* * * * AM29116.DEF FILE - D. E. WHITE - JANUARY 12, 1982 *

 ; ***********************************
 ; TWO OPERAND INSTRUCTIONS
 ; ***********************************
 ;
 ; OPCODES [5]
 ;
SUBR:           EQU             H#0     ; S minus R
SUBRC:          EQU             H#1     ; S minus R with carry
SUBS:           EQU             H#2     ; R minus S
SUBSC:          EQU             H#3     ; R minus S with carry
ADD:            EQU             H#4     ; R plus S
ADDC:           EQU             H#5     ; R plus S with carry
AND:            EQU             H#6     ; R . S
NAND:           EQU             H#7     ; R . S
EXOR:           EQU             H#8     ; R   S
NOR:            EQU             H#9     ; R + S
OR:             EQU             H#A     ; R + S
EXNOR:          EQU             H#B     ; R   S
 ;
 ;
 ; SOURCE-DESTINATION [6]               ; R     S       DEST
 ;
TORAA:          EQU             H#0     ; RAM   ACC     ACC
TORIA:          EQU             H#2     ; RAM   I       ACC
TODRA:          EQU             H#3     ; D     RAM     ACC
TORAY:          EQU             H#8     ; RAM   ACC     Y BUS
TORIY:          EQU             H#A     ; RAM   I       Y BUS
TODRY:          EQU             H#B     ; D     RAM     Y BUS
TORAR:          EQU             H#C     ; RAM   ACC     RAM
TORIR:          EQU             H#E     ; RAM   I       RAM
TODRR:          EQU             H#F     ; D     RAM     RAM
 ;
 ; ***********************************************************
TOR1: DEF AM2910,1V,  B#00,4V,        4V,    5V%, ZZZZZ ; TWO OPERAND RAM (1)
 ;
 ;              MODE,QUAD,SOURCE-DEST,OPCODE,REGISTER
 ;              [M]             [6]     [5]     [R]
 ; ***********************************************************
 ;
 ; THE [i] IN THE COMMENT BELOW THE VARIABLE-FIELD REFERS TO THE ALLOWED
 ; MNEMONIC GROUP.  EXAMPLE: MODE REFERS VIA [M] TO THE BYTE-WORD SELECT.
 ; EXAMPLE: THE ALLOWED OPCODE SUBSTITUTIONS IN TOR1 COME FROM GROUP [5]
 ; WHILE THE ALLOWED SOURCE-DESTINATIONS COME FROM GROUP [6].
 ;
```

```
;
;
;
;
;
; SOURCE-DESTINATION [7]                        R     S       DEST
;
TODAR:          EQU             H#1     ; D     ACC     RAM
TOAIR:          EQU             H#2     ; ACC   I       RAM
TODIR:          EQU             H#5     ; D     I       RAM
;
; ************************************************************
TOR2: DEF AM2910,1V,  B#10,4V,         4V,    5V%, ZZZZZ ; TWO OPERAND RAM (2)
;
;               MODE,QUAD,SOURCE-DEST,OPCODE,REGISTER
;               [M]           [7]       [5]    [R]
; ************************************************************
;
; SOURCE [8]                              R     S
;
TODA:           EQU             H#1     ; D     ACC
TOAI:           EQU             H#2     ; ACC   I
TODI:           EQU             H#5     ; D     I
; ***********************************************************
TONR: DEF AM2910,1V,   B#11,4V,    4V,    5V%, ZZZZZ ; TWO OPERAND NON-RAM
;
;               MODE, QUAD,SOURCE,OPCODE,DESTINATION
;               [M]          [8]    [5]     [4]
; ***********************************************************
```

```
 ; ************************************************
 ; SHIFT INSTRUCTIONS
 ; ************************************************
 ;
 ; DIRECTION AND INPUT [9]
 ;
SHUPZ:          EQU            H#0      ; UP 0
SHUP1:          EQU            H#1      ; UP 1
SHUPL:          EQU            H#2      ; UP QLINK
SHDNZ:          EQU            H#4      ; DOWN 0
SHDN1:          EQU            H#5      ; DOWN 1
SHDNL:          EQU            H#6      ; DOWN QLINK
SHDNC:          EQU            H#7      ; DOWN QC
SHDNOV:         EQU            H#8      ; DOWN QN QOVR
 ;
 ;
 ; SOURCE [10]
 ;
SHRR:           EQU            H#6      ; RAM   RAM
SHDR:           EQU            H#7      ; D     RAM
 ;
 ;
 ; ***************************************************
SHFTR: DEF AM2910,1V,  B#10,4V,      4V,    5V%, ZZZZZ ; SHIFT RAM
 ;
 ;              MODE,QUAD,SOURCE,DIRECT-INPT,REGISTER
 ;              [M]      [10]      [9]     [R]
 ; ***************************************************
 ;
 ;
 ; SOURCE [11]
 ;
SHA:            EQU            H#6      ; ACC
SHD:            EQU            H#7      ; D
 ;
 ;
 ; ***************************************************
SHFTNR: DEF AM2910,1V,  B#11,4V,    4V,    5V%, ZZZZZ ; SHIFT NON-RAM
 ;
 ;              MODE,QUAD,SOURCE,DIRECT-INP,DESTINATION
 ;              [M]        [11]   [9]      [4](NRY; NRA ONLY)
 ; ***************************************************
 ;
```

```
; **************************************************
;ROTATE INSTRUCTIONS
; **************************************************
;
; SOURCE-DESTINATION [12]
;
RTRA:           EQU             H#C      ; RAM   ACC
RTRY:           EQU             H#E      ; RAM   Y BUS
RTRR:           EQU             H#F      ; RAM   RAM
;
;
; ****************************************************
ROTR1: DEF  AM2910,1V,  B#00,4V,4V,        5V%, ZZZZZ ; ROTATE RAM (1)
;
;               MODE,QUAD,N,SOURCE-DEST,REGISTER
;               [M]      [N]     [12]      [R]
; ****************************************************
;
; SOURCE-DESTINATION [13]
;
RTAR:           EQU             H#0      ; ACC   RAM
RTDR:           EQU             H#1      ; D     RAM
;
;
; ****************************************************
ROTR2: DEF  AM2910,1V,  B#01,4V,4V,        5V%, ZZZZZ ; ROTATE RAM (2)
;
;               MODE,QUAD,N,SOURCE-DEST,REGISTER
;               [M]      [N]    [13]       [R]
; ****************************************************
;
; SOURCE DESTINATION [14]
;
RTDY:           EQU             D#24     ; D     Y BUS
RTDA:           EQU             D#25     ; D     ACC
RTAY:           EQU             D#28     ; ACC   Y BUS
RTAA:           EQU             D#29     ; ACC   ACC
;
;
; ****************************************************
ROTNR: DEF  AM2910,1V,  B#11,4V,H#C,      5V%, ZZZZZ ; ROTATE NON-RAM
;
;               MODE,QUAD,N,FIXED CODE,DESTINATION
;               [M]      [N]               [14]
; ****************************************************
; ***************************************************
; BIT ORIENTED INSTRUCTIONS
; ***************************************************
;
; OPCODES [15]
;
SETNR:          EQU             H#D     ; SET RAM, BIT N
RSTNR:          EQU             H#E     ; RESET RAM, BIT N
TSTNR:          EQU             H#F     ; TEST RAM, BIT N
;
;
; ********************************************************
BOR1: DEF AM2910,1V,  B#11,4V,4V,   5V%, ZZZZZ ; BIT ORIENTED RAM (1)
;
;               MODE,QUAD,N,OPCODE,REGISTER
;               [M]       [N]  [15]    [R]
; ********************************************************
;
;
; OPCODES [16]
;
LD2NR:          EQU             H#C     ; 2^N --- RAM
LDC2NR:         EQU             H#D     ; 2^N --- RAM
A2NR:           EQU             H#E     ; RAM + 2^N - RAM
S2NR:           EQU             H#F     ; RAM - 2^N - RAM
;
;
; ********************************************************
BOR2: DEF AM2910,1V,  B#10,4V,4V,   5V%, ZZZZZ ; BIT ORIENTED RAM (2)
;
;               MODE,QUAD,N,OPCODE,REGISTER
;               [M]       [N]  [16]    [R]
; ********************************************************
```

```
* * * * AM29116.DEF FILE - D. E. WHITE - JANUARY 12, 1982 *

 ;
 ;
 ;
 ;
 ;
 ; OPCODES [17]
 ;
 TSTNA:          EQU             D#0      ; TEST ACC, BIT N
 RSTNA:          EQU             D#1      ; RESET ACC, BIT N
 SETNA:          EQU             D#2      ; SET ACC, BIT N
 A2NA:           EQU             D#4      ; ACC + 2^N -- ACC
 S2NA:           EQU             D#5      ; ACC - 2^N --ACC
 LD2NA:          EQU             H#6      ; 2^N -- ACC
 LDC2NA:         EQU             D#7      ; 2^N -- ACC
 TSTND:          EQU             D#16     ; TEST D, BIT N
 RSTND:          EQU             D#17     ; RESET D, BIT N
 SETND:          EQU             D#18     ; SET D, BIT N
 A2NDY:          EQU             D#20     ; D + 2^N -- Y BUS
 S2NDY:          EQU             D#21     ; D - 2^N -- Y BUS
 LD2NY:          EQU             D#22     ; 2^N -- Y BUS
 LDC2NY:         EQU             D#23     ; 2^N -- Y BUS
 ;
 ;
 ; *********************************************************
 BONR: DEF AM2910,1V,  B#11,4V,B#110,    5V%, ZZZZZ ; BIT ORIENTED NON-RAM
 ;
 ;              MODE,QUAD,N,FIXED CODE,OPCODE
 ;               [M]      [N]           [17]
 ; *********************************************************
; **************************************************
; ROTATE AND MERGE
; **************************************************
;
; SOURCE-DEST SELECT [U,S,MASK-DEST] [18]
;
;                                                ROT   NON-ROT  MASK-DEST
MDAI:           EQU             H#7      ; D     ACC      I
MDAR:           EQU             H#8      ; D     ACC      RAM
MDRI:           EQU             H#9      ; D     RAM      I
MDRA:           EQU             H#A      ; D     RAM      ACC
MARI:           EQU             H#C      ; ACC   RAM      I
MRAI:           EQU             H#E      ; RAM   ACC      I
;
;
; **********************************************************
ROTM: DEF AM2910,1V,  B#01,4V,4V,        5V%, ZZZZZ ;ROTATE AND MERGE
;
;               MODE,QUAD,N,SOURCE-DEST,REGISTER
;               [M]      [N]    [18]        [R]
; **********************************************************
;
;
;
;
;
; **************************************************
; ROTATE AND COMPARE
; **************************************************
;
; ROT.SRC(U)-NON ROT.SRC(S)/DEST-MASK(S)[19]
;
CDAI:           EQU             H#2      ; D     ACC      I
CDRI:           EQU             H#3      ; D     RAM      I
CDRA:           EQU             H#4      ; D     RAM      ACC
CRAI:           EQU             H#5      ; RAM   ACC      I
;
;
; ********************************************
ROTC: DEF AM2910,1V,  B#01,4V,4V,             5V%, ZZZZZ ; ROTATE AND COMPARE
;
;               MODE,QUAD,N,SOURCE-DEST-MASK,REGISTER
;               [M]      [N]       [19]          [R]
; *********************************************
```

```
;
; ***********************************************
; PRIORITIZE
; ***********************************************
;
; SOURCE [20]
;
PRT1A:          EQU             H#7     ; ACC
PR1D:           EQU             H#9     ; D
;
;
; DESTINATION [21]
;
PR1A:           EQU             H#8     ; ACC
PR1Y:           EQU             H#A     ; Y BUS
PR1R:           EQU             H#B     ; RAM
;
; *********************************************
PRT1: DEF AM2910,1V,  B#10,4V,        4V,    5V%, ZZZZZ ; RAM ADDR MASK(S)
;
;               MODE,QUAD,DESTINATION,SOURCE,REG-MASK
;               [M]             [21]     [20]    [R]
; *********************************************
;
;
; DESTINATION [23]
;
PR2A:           EQU             H#0     ; ACC
PR2Y:           EQU             H#2     ; Y BUS
;
; MASK (S) [22]
;
PRA:            EQU             H#8     ; ACC
PRZ:            EQU             H#A     ; 0
PRI:            EQU             H#B     ; I
;
;
; *********************************************
PRT2: DEF AM2910,1V,  B#10,4V,  4V,  5V%, ZZZZZ ; PRIORITIZE RAM
;
;               MODE,QUAD,MASK,DEST,REG-SOURCE
;               [M]       [22] [23]    [R]
; *********************************************
```

```
;
;
;
;
; SOURCE (R) [24]
;
PR3R:           EQU             H#3     ; RAM
PR3A:           EQU             H#4     ; ACC
PR3D:           EQU             H#6     ; D
;
;
; ***********************************************
PRT3: DEF AM2910,1V,  B#10,4V,  4V,    5V%, ZZZZZ ; PRIORITIZE RAM
;
;               MODE,QUAD,MASK,SOURCE,REG-DEST
;               [M]       [22]  [24]    [R]
; ***********************************************
;
;
; SOURCE (R) [25]
;
PRTA:           EQU             H#4     ; ACC
PRTD:           EQU             H#6     ; D
;
;
; **********************************************
PRTNR: DEF  AM2910,1V,  B#11,4V,  4V,    5V%, ZZZZZ ; PRIORITIZE NON-RAM
;
;                  MODE,QUAD,MASK,SOURCE,DESTINATION
;                  [M]       [22]  [25]    [4](NRY,NRA ONLY)
; **********************************************
```

```
;
;
;
;
;
; ****************************************
; CYCLIC REDUNDANCY CHECK
; ****************************************
;
;
;
; **************************************
CRCF: DEF AM2910,B#11001100011,5V%, ZZZZZ ; FORWARD
; **************************************
;
;
; **************************************
CRCR: DEF AM2910,B#11001101001,5V%, ZZZZZ ; REVERSE
; **************************************
;
;
;
;
; ****************************************
;
; NOOP
;
; ****************************************
NOOP: DEF AM2910,H#7140, ZZZZZ             ; NO OPERATION
; ****************************************
;
```

```
 ; ************************************************
 ; STATUS
 ; ************************************************
 ;
 ; OPCODE [26]
 ;
 SONZC:          EQU             5D#3     ; SET OVR,N,C,Z
 SL:             EQU             5D#5     ; SET LINK
 SF1:            EQU             5D#6     ; SET FLAG 1
 SF2:            EQU             5D#9     ; SET FLAG 2
 SF3:            EQU             5D#10    ; SET FLAG 3
 ;
 ;
 ; **************************************************
 SETST: DEF AM2910,B#011,H#BA,5V%, ZZZZZ ; SET STATUS
 ;
 ;                            OPCODE
 ;                             [26]
 ; **************************************************
 ;
 ;
 ;
 ; OPCODE [27]
 ;
 RONCZ:          EQU             D#3      ; RESET OVR,N,C,Z
 RL:             EQU             D#5      ; RESET LINK
 RF1:            EQU             D#6      ; RESET FLAG 1
 RF2:            EQU             D#9      ; RESET FLAG 2
 RF3:            EQU             D#10     ; RESET FLAG 3
 ;
 ;****************************************************
 RSTST: DEF AM2910,B#011,H#AA,5V%, ZZZZZ ; RESET STATUS
 ;
 ;                             OPCODE
 ;                              [27]
 ; ***************************************************
```

```
;
;
; ************************************************
SVSTR: DEF  AM2910,1V,  B#10,H#7A, 5V%,      ZZZZZ ; SAVE STATUS-RAM
;
;                 MODE,QUAD,FIXED,RAM ADDRESS/DEST
;                  [M]               [R]
; ************************************************
;
;
;************************************************
SVSTNR: DEF  AM2910,1V, B#11,H#7A, 5V%,       ZZZZZ ; SAVE STATUS NON-RAM
;
;                 MODE,QUAD,FIXED,DESTINATION
;                  [M]           [4](NRY,NRA ONLY)
; ************************************************
;
;
; ********************************************
; TEST STATUS
; ********************************************
;
; OPCODE (CT)
;
TNOZ:           EQU            D#0     ; TEST (N OVR) + Z
TNO:            EQU            D#2     ; TEST N OVR
TZ:             EQU            D#4     ; TEST Z
TOVR:           EQU            D#6     ; TEST OVR
TLOW:           EQU            D#8     ; TEST LOW
TC:             EQU            D#10    ; TEST C
TZC:            EQU            D#12    ; TEST Z + C
TN:             EQU            D#14    ; TEST N
TL:             EQU            D#16    ; TEST LINK
TF1:            EQU            D#18    ; TEST FLAG 1
TF2:            EQU            D#20    ; TEST FLAG 2
TF3:            EQU            D#22    ; TEST FLAG 3
;
;
; ************************************************
TEST: DEF AM2910,B#011,H#9A,5V%, ZZZZZ ;  TEST STATUS
;
;                FIXED,     OPCODE
;                            [CT]
; ************************************************
; * * * * * * * * * * * * * * * * *
;
; Am2914 INSTRUCTION SET
; Vectored Priority Interrupt Controller
;
; * * * * * * * * * * * * * * * * *
;
MCLR:   EQU     H#0     ; MASTER CLEAR
CLRIN:  EQU     H#1     ; CLEAR ALL INTERRUPTS
CLRMB:  EQU     H#2     ; CLEAR INTERRUPTS FROM M-BUS
CLRMR:  EQU     H#3     ; CLEAR INTERRUPTS FROM MASK REGISTER
CLRVC:  EQU     H#4     ; CLEAR INTERRUPT FROM LAST VECTOR READ
RDVC:   EQU     H#5     ; READ VECTOR
RDSTA:  EQU     H#6     ; READ STATUS REGISTER
RDM:    EQU     H#7     ; READ MASK REGISTER
SETM:   EQU     H#8     ; SET MASK REGISTER
LDSTA:  EQU     H#9     ; LOAD STATUS REGISTER
BCLRM:  EQU     H#A     ; BIT CLEAR MASK REGISTER
BSETM:  EQU     H#B     ; BIT SET MASK REGISTER
CLRM:   EQU     H#C     ; CLEAR MASK REGISTER
DISIN:  EQU     H#D     ; DISABLE INTERRUPT REQUEST
LDM:    EQU     H#E     ; LOAD MASK REGISTER
ENIN:   EQU     H#F     ; ENABLE INTERRUPT REQUEST
;
```

```
 ; * * * * * * * * * * * * * * * * * * *
 ;
 ; Am2940 DMA CONTROL UNIT
 ; DMA Address Generator
 ; TO USE - DELETE THE .40 AND DELETE THE Am2942 INSTRUCTION SET
 ; DUE TO DUPLICATE MNEMONICS
 ;
 ; * * * * * * * * * * * * * * * * * * *
 ; INSTRUCTIONS
 ;
 WRCR.40:       EQU     Q#0      ; WRITE CONTROL REGISTER
 RDCR.40:       EQU     Q#1      ; READ CONTROL REGISTER
 RDWC.40:       EQU     Q#2      ; READ WORD COUNTER
 RDAC.40:       EQU     Q#3      ; READ ADDRESS COUNTER
 REIN.40:       EQU     Q#4      ; REINITIALIZE COUNTERS
 LDAD.40:       EQU     Q#5      ; LOAD ADDRESS
 LDWC.40:       EQU     Q#6      ; LOAD WORD COUNT
 ENCT.40:       EQU     Q#7      ; ENABLE COUNTERS
 ;
 ;CONTROL MODE BYTE
 ;NOTE - BITS 3 THROUGH 7 ARE DON'T CARE
 ;
 WC1I.40:       EQU     8Q#0%    ; WORD COUNT EQUALS ONE, INCREMENT ADDR CNTR
 WCCI.40:       EQU     8Q#1%    ; WORD COUNT COMPARE, INCREMENT ADDR CNTR
 ADCI.40:       EQU     8Q#2%    ; ADDR COMPARE, INCREMENT ADDR CNTR
 WCOI.40:       EQU     8Q#3%    ; WORD CNTR CARRY OUT, INCREMENT ADDR CNTR
 WC1D.40:       EQU     8Q#4%    ; WORD COUNT EQUALS ONE, DECREMENT ADDR CNTR
 WCCD.40:       EQU     8Q#5%    ; WORD COUNT COMPARE, DECREMENT ADDR CNTR
 ADCD.40:       EQU     8Q#6%    ; ADDR COMPARE, DECREMENT ADDR CNTR
 WCOD.40:       EQU     8Q#7%    ; WORD CNTR CARRY OUT, DECREMENT ADDR CNTR
```

```
AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4
* * * * AM29116.DEF FILE - D. E. WHITE - JANUARY 12, 1982 *

 ; **********************************
 ; Am2942 INSTRUCTION SET
 ; Programmable Timer/Counter
 ; DMA Address Generator
 ; *********************************
 ;
 ; DMA INSTRUCTIONS - ALSO REQUIRES INSTRUCTION ENABLE = LOW
 ;
 WRCR:           EQU     H#0     ; WRITE CONTROL REGISTER
 RDCR:           EQU     H#1     ; READ CONTROL REGISTER
 RDWC:           EQU     H#2     ; READ WORD COUNTER
 RDAC:           EQU     H#3     ; READ ADDRESS COUNTER
 REIN:           EQU     H#4     ; REINITIALIZE COUNTERS
 LDAD:           EQU     H#5     ; LOAD ADDRESS
 LDWC:           EQU     H#6     ; LOAD WORD COUNT
 ENCT:           EQU     H#7     ; ENABLE COUNTERS
 ;
 ; DMA INSTRUCTION - INSTRUCTION ENABLE = HIGH
 ;
 ; ALL OF THE ABOVE BECOME INSTRUCTION DISABLE
 ;
 ; TIMER/COUNTER INSTRUCTIONS - INSTRUCTION ENABLE = LOW
 ;
 WCRT:           EQU     H#8     ; WRITE CONTROL REGISTER, T/C
 REAC:           EQU     H#9     ; REINITIALIZE ADDRESS COUNTER
 RWCT:           EQU     H#A     ; READ WORD COUNTER, T/C
 RACT:           EQU     H#B     ; READ ADDRESS COUNTER, T/C
 RAWC:           EQU     H#C     ; REINITIALIZE ADDRESS AND WORD COUNTERS
 LDAT:           EQU     H#D     ; LOAD ADDRESS, T/C
 LWCT:           EQU     H#E     ; LOAD WORD COUNT, T/C
 REWC:           EQU     H#F     ; REINITIALIZE WORD COUNTER
 ;
 ; TIMER/COUNTER INSTRUCTIONS - INSTRUCTION ENABLE = HIGH
 ;
 ; ALL OF THE ABOVE T/C INSTRUCTIONS BECOME INSTRUCTION DISABLE, T/C
 ;
```

```
; ************************************************************************
;
; Am2904 INSTRUCTION SET - PARTIAL ONLY!!!!!
;  BUILD ONLY WHAT YOU NEED!!!
; Status and Shift Control Unit
;
; * * * * * * * * * * * * * * * * * * * * * * * * * *
;
;
; DOWN SHIFTING
;
SDZRZQ:         EQU     H#0             ; Z->RN; Z->QN
SDOROQ:         EQU     H#1             ; 1->RN; 1->QN
SLN.RECOVER:    EQU     H#2             ; 0->RN; R0->Mc; MN->QN
DDOR:           EQU     H#3             ; 1->RN; R0->QN
DDMCR:          EQU     H#4             ; Mc->RN; R0->QN
DLN.RECOVER:    EQU     H#5             ; MN->RN; R0->QN
DDZR:           EQU     H#6             ; 0->RN; R0->QN
DDZRQMC:        EQU     H#7             ; 0->RN; R0->QN; Q0->Mc
SDROTMC:        EQU     H#8             ; ROT.R; R0->Mc; ROT.Q
SDROTC:         EQU     H#9             ; ROT.R WITH Mc; ROT.Q
SDROT:          EQU     H#A             ; ROT.R; ROT.Q
SDIC:           EQU     H#B             ; Ic->RN; R0->QN
DDROTC:         EQU     H#C             ; Mc->RN; R0->QN; Q0->Mc
DDROTMC:        EQU     H#D             ; Q0->RN; R0->QN; Q0->Mc
DDINIOVR:       EQU     H#E             ; IN EXOR IOVR -> RN; R0->QN
DDROT:          EQU     H#F             ; DOUBLE PRECISION ROTATE DOWN
;
;
; UP SHIFTING (INCOMPLETE)
;
SURZQZ:         EQU     H#2             ; R0<-0; Q0<-0
;
```

```
AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4
* * * * AM29116.DEF FILE - D. E. WHITE - JANUARY 12, 1982 *

 ; SHIFT ENABLES
 ;
 SE.EN:           EQU     B#0               ; ENABLE SHIFTING
 SE.DIS:          EQU     B#1               ; DISABLE SHIFTING
 ;
 ;
 ;
 ;
 ; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 ; Am2904 STATUS REGISTER INSTRUCTION CODES
 ; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
 ; MACHINE STATUS REGISTER INSTRUCTION CODES
 ;    I5-I4-I3-I2-I1-I0 AND EZ-EC-EN-EOVR-CEM ENABLES
 ; MICRO STATUS REGISTER INSTRUCTION CODES
 ;    I5-I4-I3-I2-I1-I0 AND CEu ENABLE
 ;
 ; THE FOLLOWING TAKES THESE ALL TOGETHER - YOU MAY WISH TO DO THIS ANOTHER WAY
 ;
 ; ORDER: 543 210 ZCNOVR CEM CEu
 ;        Q#  Q#  H#     B#  B#
 ;
 ONELEVEL:        EQU     12Q#0000          ; Y -> MSR; MSR -> USR
 SET.MSR:         EQU     12Q#0101          ; SET MACRO STATUS ONLY
 SET.USR:         EQU     12Q#0176          ; SET MICRO STATUS ONLY
 SWAP.REG:        EQU     12Q#0200          ; MSR <--> USR
 ;
 LOAD.MSR:        EQU     12Q#2001          ; ALU STATUS -> MSR
 ; THE ABOVE IS ON OF SEVERAL CODES - YOU DON'T NEED THEM ALL!
 ;
 LOAD.USR:        EQU     12Q#2076          ; ALU STATUS -> USR
 ; DITTO!
 ;
 LOAD.BOTH:       EQU     12Q#2000          ; ALU -> MSR, USR
 ; AGAIN DITTO!
 ;
 LDINVRTM:        EQU     12Q#3001          ; ALU -> MSR; Ic INVERTED
 LDINVRTU:        EQU     12Q#3076          ; ALU -> USR; Ic INVERTED
 LOAD.INVERT:     EQU     12Q#3000          ; ALU -> MSR, USR; Ic INVERTED
 ;
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; Am2904 CONDITION CODE OUTPUT INSTRUCTION CODES
; * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; caution!  I5-I4-I3-I2-I1-I0 ARE ALSO USED FOR TESTING!!!!
; ENABLE TESTING VIA OEct ENABLE
; * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
TESTMZ:         EQU     12Q#4477        ; NO STATUS OPERATION
TESTMOVR:       EQU     12Q#4677        ; NO STATUS OPERATION
TESTMC:         EQU     12Q#5277        ; NO STATUS OPERATION
TESTMN:         EQU     12Q#5677        ;
TEST.IOVR:      EQU     12Q#6677        ;
TEST.IC:        EQU     12Q#7277        ;
;
;
; TEST ENABLE
;
OECTEN:         EQU     B#0
OECTDIS:        EQU     B#1
;
;
; INSTRUCTION ENABLE
;
IEN.04:         EQU     B#0
IENDIS:         EQU     B#1
;
```

```
* * * * AM29116.DEF FILE - D. E. WHITE - JANUARY 12, 1982 *

 AM2904:   DEF   42X,    12VQ#2001, 1VB#1,  1VB#0, 4VX,  1VB#1, 3X
 ; DEFAULTS              LOAD.MSR   OECTDIS OEYEN  X     SE.DIS
 ;
 SHIFT.04:    DEF   56X,  4VX, 1B#0, 3X
 ;                       SHIFT SE.EN
 TEST.04:     DEF   42X, 12VQ#7777, 1VB#0, 9X
 ;                      DISABLED   OECTEN
 STATUS.04:   DEF   42X, 12VQ#2001, B#1, 1VB#0, 4X, B#1, 3X
 ;                      LOAD.MSR  NO CT OEYEN     SE.DIS
 ; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
 ; Am2925 CYCLE LENGTH SELECT
 ; System Clock Generator and Driver
 ;
 ; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
 ; THE FOLLOWING ARE THE CYCLE LENGTH CODES (PRELIM)
 ;
CLA:    EQU     Q#0     ; 3 CLOCK PERIODS       100ns AT 30MHz
CLB:    EQU     Q#1     ; 4                     160ns AT 25MHz
CLC:    EQU     Q#5     ; 5                     200ns AT 25MHz
CLD:    EQU     Q#7     ; 6                     200ns AT 30MHz
CLE:    EQU     Q#3     ; 7                     280ns AT 25MHz
CLF:    EQU     Q#2     ; 8                     320ns AT 25MHz
CLG:    EQU     Q#6     ; 9                     300ns AT 30MHz
CLH:    EQU     Q#4     ; 10 CLOCK PERIODS      322ns AT 31MHz
 ;                                    (max crystal frequency is 31MHz)
 ;
 ; OTHER CONTROL LINES FOR THE Am2925
 ; INCOMPLETELY DEFINED AT PRESENT (IN THIS FILE)
 ;
FIRST.25:       EQU     B#1             ;
LAST.25:        EQU     B#0             ;
 ;
HALT:           EQU     B#00            ;
NOHALT:         EQU     B#00            ;
 ;
SINGLSTP:       EQU     B#00            ;
RUN:            EQU     B#00            ;
 ;
WAITREQ:        EQU     B#0             ;
NOWAITRQ:       EQU     B#1             ;
 ;
READY:          EQU     B#0             ;
NOTREADY:       EQU     B#1             ;
 ;
INITIALZE:      EQU     B#0             ;
NO.INIT:        EQU     B#1             ;
 ;
 ; * * * * * * * * * * * * * * * * * * * * * *
 ;
 ; Am2950/51
 ;
 ; Eight-Bit Bidirectional I/O Ports
 ;
 ; * * *
 ;
 ;
 END

 TOTAL PHASE 1 ERRORS =   0
```

```
        ;
        ; D.E.WHITE   OCT. 17, 1980
        ; EDITED DEC. 30, 1981
        ; EDITED JAN. 12, 1982
        ;
        ;
        ;
        ; This file was created just to debug the master file, CONTROLR.DEF.  It
        ; exercises the Am29116 DEF statements to demonstrate their usage.
        ;
        ;
        ;
        ;
        ;
        ; Advanced Micro Devices reserves the right to make changes in its product
        ; without notice in order to improve design or performance characteristics.
        ; The company assumes no responsibility for the use of any circuits or
        ; programs described herein.
        ;
        ; Am29116 Mnemonics Copyright (c) Advanced Micro Devices, Inc.
        ;
        ;
        ;
        ;
        ;
```

```
AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4                                          PAGE    2
DEMO SRC FILE FOR AM29116.DEF - D. E. WHITE - JAN 12, 1982 E

      ;
      ;
      ;
      ; FREE FORMAT CONSTANTS
      ;
 000E          FF1:    EQU     4H#E    ; AM2910 CONTINUE
               FF4:    EQU     5B#01000 ; AM29116 ENABLES
      ;
      ;  Additional mnemonics may be defined in the .SRC file as needed so that the
      ;  .DEF file does not need to be re-assembled.  This should be minimized for
      ;  proper documentation
      ;
      ;     NOTE: the below is a BLANK line - AMDASM accepts it

      ; NO OPERATION
 0008 ;
               NOOP
 0000 ;
```

```
      ;
      ;
      ;
      ; SINGLE OPERAND INSTRUCTIONS
      ;
      ;
0001         SOR     ,,, W, MOVE, SORA, R10          ; ACC <- R10
0002         SOR     ,,, W, INC,  SORA, R31          ; R31 <- R31 + 1
0003         SOR     ,,, W, NEG,  SORA, R31          ; ACC <- NOT R31 + 1
0004         SOR     ,,, W, COMP, SORA, R31          ; ACC <- NOT R31
      ;
0005         SONR    ,,, W, MOVE, SOI,  NRS          ; STATUS <- NEXT WORD
0006         FF      H#E, 15X, H#0123, B#01000, 24X  ; DATA FOR ABOVE
      ;
0007         SONR    ,,, W, NEG,  SOI,  NRA          ; ACC <- NOT I + 1
0008         FF      H#E, 15X, H#0000, B#01000, 24X  ; DATA FOR ABOVE
      ;
      ;      The above is an example of the free-format executable statement.
      ;      It supplies data for the two-microcycle immediate data instructions.
```

```
        ;
        ;
        ; TWO OPERAND INSTRUCTIONS
        ;
        ;
0009          TOR1    ,,, W, TODRR, SUBS, R31        ; R31 <- R31-<DATA LATCH>
        ;
000A          TOR1    ,,, W, TORIY, AND,  R0         ; Y <- R0 AND IMMEDIATE
000B          FF      FF1, 15X, H#ABCD, FF4, 24X     ; DATA FOR ABOVE
        ;
000C          TOR1    ,,, B, TORAA, ADD,  R31        ; ACC <- R31 + ACC
        ;
000D          TOR2    ,,, W, TODIR, SUBS, R0         ; R0 <- DATA - IMMEDIATE
000E          FF      FF1, 15X, H#0002, FF4, 24X     ; DATA FOR THE ABOVE
        ;
000F          TOR2    ,,, W, TODAR, EXOR, R30        ; R30 <- D EXOR ACC
        ;
0010          TONR    ,,, W, TOAI, ADDC,  NRA        ; ACC <= ACC + I + CIN
0011          FF      FF1, 15X, H#5555, FF4, 24X     ; DATA FOR ABOVE
        ;
0012          TONR    ,,, W, TODA, AND,   NRA        ; ACC <- D AND ACC
0013          TONR    ,,, B, TODA, SUBS,  NRA        ; ACC <- D - ACC
        ;
```

```
AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4                                                     PAGE    5
DEMO SRC FILE FOR AM29116.DEF - D. E. WHITE - JAN 12, 1982 E

        ; SINGLE BIT SHIFT INSTRUCTIONS
        ;
        ;
0014          SHFTR   ,,, W, SHRR, SHUP1, R10          ; SHIFT R10 <- 1
0015          SHFTR   ,,, W, SHRR, SHUPL, R10          ; SHIFT R10 <- QLINK
0016          SHFTR   ,,, W, SHRR, SHDNZ, R10          ; SHIFT 0 -> R10
        ;
0017          SHFTNR  ,,, W, SHA,  SHDNOV,NRA          ; SHIFT QN XOR QOVR ->ACC
        ;
        ;
        ;
        ; ROTATE INSTRUCTIONS
        ;
        ;
0018          ROTR1   ,,, W, H#8,  RTRA,  R10          ; ACC <- R10
0019          ROTR1   ,,, W, H#F,  RTRR,  R31          ; R31 <- R31
        ;
001A          ROTR2   ,,, W, H#3,  RTAR,  R10          ; R10 <- ACC
        ;
001B          ROTNR   ,,, W,       H#5,   RTAA         ; ACC <- ACC
001C          ROTNR   ,,, W,       H#E,   RTDA         ; ACC <- DATA LATCH
        ;
        ;
        ;
        ; BIT-ORIENTED INSTRUCTIONS
        ;
        ;
001D          BOR1    ,,, W, H#6,  SETNR, R10          ; SET BIT 6
001E          BOR1    ,,, W, H#8,  RSTNR, R10          ; RESET BIT 8
001F          BOR1    ,,, W, H#F,  TSTNR, R31          ; TEST BIT 15
        ;
0020          BOR2    ,,, W, H#C,  S2NR,  R31          ; R31 <- R31 - 2**12
        ;
0021          BONR    ,,, W, H#3,         RSTNA        ; RESET BIT 3 IN ACC
0022          BONR    ,,, W, H#C,         LD2NA        ; ACC <- 2**12
0023          BONR    ,,, W, H#3,         RSTND        ; RESET BIT 3, DATA LATCH
        ;
```

```
      ; ROTATE AND MERGE INSTRUCTIONS
      ;
      ;
                         ; R6 <- ACC AND R6 OR NOT ACC AND D (ROTATED)
      ;
0024        ROTM    ,,, W, H#6,  MDRA,  R6
      ;
0025        ROTM    ,,, W, H#B,  MARI,  R6
0026        FF      FF1, 15X, H#AAAA, FF4, 24X
      ;
      ;
      ;
      ; ROTATE AND COMPARE INSTRUCTIONS
      ;
      ;
0027        ROTC    ,,, W, H#6,  CDAI,  R5          ; D ACC I (R?)
0028        FF      FF1, 15X, H#1874, FF4, 24X
      ;
0029        ROTC    ,,, W, H#C,  CRAI,  R6
002A        FF      FF1, 15X, H#5656, FF4, 24X
      ;
      ;
      ;
      ; PRIORITIZE INSTRUCTIONS
      ;
      ;
002B        PRT1    ,,, W, PR1A, PRT1A, R6
      ;
002C        PRT2    ,,, W, PRA,  PR2A,  R6
      ;
002D        PRT3    ,,, W, PRA,  PR3R,  R6
      ;
002E        PRTNR   ,,, W, PRA,  PRTA,  NRY
      ;
      ; CRC INSTRUCTIONS (SINGLE CALL)
      ;
      ;
002F        CRCF     ,,, R6                          ; CRC FORWARD
      ;
0030        CRCR     ,,, R6                          ; CRC REVERSE
      ;
      ;
      ;
      ; STATUS INSTRUCTIONS
      ;
      ;
0031        SETST    ,,, SONZC                       ; SET ALU STATUS
0032        SETST    ,,, SF3                         ; SET FLAG 3
      ;
0033        RSTST    ,,, RF2                         ; RESET FLAG 2
0034        RSTST    ,,, RONCZ                       ; RESET ALU STATUS
      ;
0035        SVSTR    ,,, W, R6                       ; R6 <- STATUS
0036        SVSTNR   ,,, W, NRA                      ; ACC <- STATUS
      ;
      ;
      ;
      ; TEST STATUS INSTRUCTIONS
      ;
      ;
0037        TEST     ,,, TZ                          ; TEST Z
0038        TEST     ,,, TF1                         ; TEST FLAG 1
0039        TEST     ,,, TNO                         ; TEST N EXOR OVR
003A        TEST     ,,, TC                          ; TEST C
      ;
      ;
      ;
      ;
      ;
      ;
      END
```

```
AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4
DEMO SRC FILE FOR AM29116.DEF - D. E. WHITE - JAN 12, 1982 E

 0000 1110XXXXXXXXXXXX XXX0111000101000 00001000XXXXXXXX XXXXXXXXXXXXXXXX
 0001 1110XXXXXXXXXXXX XXX1101100000001 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 0002 1110XXXXXXXXXXXX XXX1101110000011 11101000XXXXXXXX XXXXXXXXXXXXXXXX
 0003 1110XXXXXXXXXXXX XXX1101111000011 11101000XXXXXXXX XXXXXXXXXXXXXXXX
 0004 1110XXXXXXXXXXXX XXX1101101000011 11101000XXXXXXXX XXXXXXXXXXXXXXXX
 0005 1110XXXXXXXXXXXX XXX1111100011100 10001000XXXXXXXX XXXXXXXXXXXXXXXX
 0006 1110XXXXXXXXXXXX XXX0000000100100 01101000XXXXXXXX XXXXXXXXXXXXXXXX
 0007 1110XXXXXXXXXXXX XXX1111111011100 00101000XXXXXXXX XXXXXXXXXXXXXXXX
 0008 1110XXXXXXXXXXXX XXX0000000000000 00001000XXXXXXXX XXXXXXXXXXXXXXXX
 0009 1110XXXXXXXXXXXX XXX1001111001011 11101000XXXXXXXX XXXXXXXXXXXXXXXX
 000A 1110XXXXXXXXXXXX XXX1001010011000 00001000XXXXXXXX XXXXXXXXXXXXXXXX
 000B 1110XXXXXXXXXXXX XXX1010101111001 10101000XXXXXXXX XXXXXXXXXXXXXXXX
 000C 1110XXXXXXXXXXXX XXX0000000010011 11101000XXXXXXXX XXXXXXXXXXXXXXXX
 000D 1110XXXXXXXXXXXX XXX1100101001000 00001000XXXXXXXX XXXXXXXXXXXXXXXX
 000E 1110XXXXXXXXXXXX XXX0000000000000 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 000F 1110XXXXXXXXXXXX XXX1100001100011 11001000XXXXXXXX XXXXXXXXXXXXXXXX
 0010 1110XXXXXXXXXXXX XXX1110010010100 00101000XXXXXXXX XXXXXXXXXXXXXXXX
 0011 1110XXXXXXXXXXXX XXX0101010101010 10101000XXXXXXXX XXXXXXXXXXXXXXXX
 0012 1110XXXXXXXXXXXX XXX1110001011000 00101000XXXXXXXX XXXXXXXXXXXXXXXX
 0013 1110XXXXXXXXXXXX XXX0110001001000 00101000XXXXXXXX XXXXXXXXXXXXXXXX
 0014 1110XXXXXXXXXXXX XXX1100110000101 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 0015 1110XXXXXXXXXXXX XXX1100110001001 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 0016 1110XXXXXXXXXXXX XXX1100110010001 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 0017 1110XXXXXXXXXXXX XXX1110110100000 00101000XXXXXXXX XXXXXXXXXXXXXXXX
 0018 1110XXXXXXXXXXXX XXX1001000110001 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 0019 1110XXXXXXXXXXXX XXX1001111111111 11101000XXXXXXXX XXXXXXXXXXXXXXXX
 001A 1110XXXXXXXXXXXX XXX1010011000001 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 001B 1110XXXXXXXXXXXX XXX1110101110011 10101000XXXXXXXX XXXXXXXXXXXXXXXX
 001C 1110XXXXXXXXXXXX XXX1111110110011 00101000XXXXXXXX XXXXXXXXXXXXXXXX
 001D 1110XXXXXXXXXXXX XXX1110110110101 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 001E 1110XXXXXXXXXXXX XXX1111000111001 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 001F 1110XXXXXXXXXXXX XXX1111111111111 11101000XXXXXXXX XXXXXXXXXXXXXXXX
 0020 1110XXXXXXXXXXXX XXX1101100111111 11101000XXXXXXXX XXXXXXXXXXXXXXXX
 0021 1110XXXXXXXXXXXX XXX1110011110000 0101000XXXXXXXXX XXXXXXXXXXXXXXXX
 0022 1110XXXXXXXXXXXX XXX1111100110001 1001000XXXXXXXXX XXXXXXXXXXXXXXXX
 0023 1110XXXXXXXXXXXX XXX1110011110100 0101000XXXXXXXXX XXXXXXXXXXXXXXXX
 0024 1110XXXXXXXXXXXX XXX1010110101000 11001000XXXXXXXX XXXXXXXXXXXXXXXX
 0025 1110XXXXXXXXXXXX XXX1011011110000 11001000XXXXXXXX XXXXXXXXXXXXXXXX
 0026 1110XXXXXXXXXXXX XXX1010101010101 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 0027 1110XXXXXXXXXXXX XXX1010110001000 10101000XXXXXXXX XXXXXXXXXXXXXXXX
 0028 1110XXXXXXXXXXXX XXX0001100001110 10001000XXXXXXXX XXXXXXXXXXXXXXXX
 0029 1110XXXXXXXXXXXX XXX1011100010100 11001000XXXXXXXX XXXXXXXXXXXXXXXX
 002A 1110XXXXXXXXXXXX XXX0101011001010 11001000XXXXXXXX XXXXXXXXXXXXXXXX
 002B 1110XXXXXXXXXXXX XXX1101000011100 11001000XXXXXXXX XXXXXXXXXXXXXXXX
 002C 1110XXXXXXXXXXXX XXX1101000000000 11001000XXXXXXXX XXXXXXXXXXXXXXXX
 002D 1110XXXXXXXXXXXX XXX1101000001100 11001000XXXXXXXX XXXXXXXXXXXXXXXX
 002E 1110XXXXXXXXXXXX XXX1111000010000 00001000XXXXXXXX XXXXXXXXXXXXXXXX
 002F 1110XXXXXXXXXXXX XXX1100110001100 11001000XXXXXXXX XXXXXXXXXXXXXXXX
 0030 1110XXXXXXXXXXXX XXX1100110100100 11001000XXXXXXXX XXXXXXXXXXXXXXXX
 0031 1110XXXXXXXXXXXX XXX0111011101000 01101000XXXXXXXX XXXXXXXXXXXXXXXX
 0032 1110XXXXXXXXXXXX XXX0111011101001 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 0033 1110XXXXXXXXXXXX XXX0111010101001 00101000XXXXXXXX XXXXXXXXXXXXXXXX
 0034 1110XXXXXXXXXXXX XXX0111010101000 01101000XXXXXXXX XXXXXXXXXXXXXXXX
 0035 1110XXXXXXXXXXXX XXX1100111101000 11001000XXXXXXXX XXXXXXXXXXXXXXXX
 0036 1110XXXXXXXXXXXX XXX1110111101000 00101000XXXXXXXX XXXXXXXXXXXXXXXX
 0037 1110XXXXXXXXXXXX XXX0111001101000 10001000XXXXXXXX XXXXXXXXXXXXXXXX
 0038 1110XXXXXXXXXXXX XXX0111001101010 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 0039 1110XXXXXXXXXXXX XXX0111001101000 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 003A 1110XXXXXXXXXXXX XXX0111001101001 01001000XXXXXXXX XXXXXXXXXXXXXXXX
 SYMBOLS

  A2NA      0004
  A2NDY     0014
  A2NR      000E
  ADCD.40   0006
  ADCI.40   0002
  ADD       0004
  ADDC      0005
  AND       0006
  B         0000
  BCLRM     000A
  BSETM     000B
  CDAI      0002
  CDRA      0004
  CDRI      0003
  CJP       0003
  CJPP      000B
  CJS       0001
  CJV       0006
  CLA       0000
  CLB       0001
  CLC       0005
  CLD       0007
  CLE       0003
  CLF       0002
  CLG       0006
  CLH       0004
  CLRIN     0001
  CLRM      000C
  CLRMB     0002
  CLRMR     0003
  CLRVC     0004
  COMP      000D
  CONT      000E
  CRAI      0005
  CRTN      000A
  DDINIOVR  000E
  DDMCR     0004
  DDOR      0003
  DDROT     000F
  DDROTC    000C
  DDROTMC   000D
  DDZR      0006
  DDZRQMC   0007
  DISIN     000D
  DLE.DIS   0000
  DLE.EN    0001
  DLN.RECO  0005
  ENCT      0007
  ENCT.40   0007
  ENIN      000F
  EXNOR     000B
  EXOR      0008
  FF1       000E
  FF4       0008
  FIRST.25  0001
  HALT      0000
```

```
AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4
DEMO SRC FILE FOR AM29116.DEF - D. E. WHITE - JAN 12, 1982 E

 IDIS      0001
 IEN       0000
 IEN.04    0000
 IENDIS    0001
 INC       000E
 INITIALZ  0000
 JMAP      0002
 JRP       0007
 JSRP      0005
 JZ        0000
 LAST.25   0000
 LD2NA     0006
 LD2NR     000C
 LD2NY     0016
 LDAD      0005
 LDAD.40   0005
 LDAT      000D
 LDC2NA    0007
 LDC2NR    000D
 LDC2NY    0017
 LDCT      000C
 LDINVRTM  0601
 LDINVRTU  063E
 LDM       000E
 LDSTA     0009
 LDWC      0006
 LDWC.40   0006
 LOAD.BOT  0400
 LOAD.INV  0600
 LOAD.MSR  0401
 LOAD.USR  043E
 LOOP      000D
 LWCT      000E
 MARI      000C
 MCLR      0000
 MDAI      0007
 MDAR      0008
 MDRA      000A
 MDRI      0009
 MOVE      000C
 MRAI      000E
 N0        0000
 N1        0001
 N2        0002
 N3        0003
 N4        0004
 N5        0005
 N6        0006
 N7        0007
 N8        0008
 N9        0009
 NA        000A
 NAND      0007
 NB        000B
 NC        000C
 ND        000D
 NE        000E
 NEG       000F
```

```
  NF        000F
  NO.INIT   0001
  NOHALT    0000
  NOR       0009
  NOTREADY  0001
  NOWAITRQ  0001
  NRA       0001
  NRAS      0005
  NRS       0004
  NRY       0000
  OECTDIS   0001
  OECTEN    0000
  OETDIS    0000
  OETEN     0001
  OEYDIS    0001
  OEYEN     0000
  ONELEVEL  0000
  OR        000A
  PR1A      0008
  PR1D      0009
  PR1R      000B
  PR1Y      000A
  PR2A      0000
  PR2Y      0002
  PR3A      0004
  PR3D      0006
  PR3R      0003
  PRA       0008
  PRI       000B
  PRT1A     0007
  PRTA      0004
  PRTD      0006
  PRZ       000A
  PUSH      0004
  R0        0000
  R1        0001
  R10       000A
  R11       000B
  R12       000C
  R13       000D
  R14       000E
  R15       000F
  R16       0010
  R17       0011
  R18       0012
  R19       0013
  R2        0002
  R20       0014
  R21       0015
  R22       0016
  R23       0017
  R24       0018
  R25       0019
  R26       001A
  R27       001B
  R28       001C
  R29       001D
  R3        0003
```

```
 R30      001E
 R31      001F
 R4       0004
 R5       0005
 R6       0006
 R7       0007
 R8       0008
 R9       0009
 RACT     000B
 RAWC     000C
 RDAC     0003
 RDAC.40  0003
 RDCR     0001
 RDCR.40  0001
 RDM      0007
 RDSTA    0006
 RDVC     0005
 RDWC     0002
 RDWC.40  0002
 REAC     0009
 READY    0000
 REIN     0004
 REIN.40  0004
 REWC     000F
 RF1      0006
 RF2      0009
 RF3      000A
 RFCT     0008
 RL       0005
 RONCZ    0003
 RPCT     0009
 RSTNA    0001
 RSTND    0011
 RSTNR    000E
 RTAA     001D
 RTAR     0000
 RTAY     001C
 RTDA     0019
 RTDR     0001
 RTDY     0018
 RTRA     000C
 RTRR     000F
 RTRY     000E
 RUN      0000
 RWCT     000A
 S2NA     0005
 S2NDY    0015
 S2NR     000F
 SDIC     000B
 SDOROQ   0001
 SDROT    000A
 SDROTC   0009
 SDROTMC  0008
 SDZRZQ   0000
 SE.DIS   0001
 SE.EN    0000
 SET.MSR  0041
 SET.USR  007E
```

```
  SETM      0008
  SETNA     0002
  SETND     0012
  SETNR     000D
  SF1       0006
  SF2       0009
  SF3       000A
  SHA       0006
  SHD       0007
  SHDN1     0005
  SHDNC     0007
  SHDNL     0006
  SHDNOV    0008
  SHDNZ     0004
  SHDR      0007
  SHRR      0006
  SHUP1     0001
  SHUPL     0002
  SHUPZ     0000
  SINGLSTP  0000
  SL        0005
  SLN.RECO  0002
  SOA       0004
  SOAR      0004
  SOD       0006
  SODR      0006
  SOI       0007
  SOIR      0007
  SONZC     0003
  SORA      0000
  SORR      000B
  SORS      0003
  SORY      0002
  SOSE      000A
  SOSER     000A
  SOZ       0008
  SOZE      0009
  SOZER     0009
  SOZR      0008
  SRE.DIS   0001
  SRE.EN    0000
  SUBR      0000
  SUBRC     0001
  SUBS      0002
  SUBSC     0003
  SURZQZ    0002
  SWAP.REG  0080
  TC        000A
  TEST.IC   0EBF
  TEST.IOV  0DBF
  TESTMC    0ABF
  TESTMN    0BBF
  TESTMOVR  09BF
  TESTMZ    093F
  TF1       0012
  TF2       0014
  TF3       0016
  TL        0010
```

```
   TLOW      0008
   TN        000E
   TNO       0002
   TNOZ      0000
   TOAI      0002
   TOAIR     0002
   TODA      0001
   TODAR     0001
   TODI      0005
   TODIR     0005
   TODRA     0003
   TODRR     000F
   TODRY     000B
   TORAA     0000
   TORAR     000C
   TORAY     0008
   TORIA     0002
   TORIR     000E
   TORIY     000A
   TOVR      0006
   TSTNA     0000
   TSTND     0010
   TSTNR     000F
   TWB       000F
   TZ        0004
   TZC       000C
   W         0001
   WAITREQ   0000
   WC1D.40   0004
   WC1I.40   0000
   WCCD.40   0005
   WCCI.40   0001
   WCOD.40   0007
   WCOI.40   0003
   WCRT      0008
   WRCR      0000
   WRCR.40   0000

 TOTAL PHASE 2 ERRORS =    0
```

A DISK CONTROLLER

DISKCTLR .DEF & .SRC

```
;
; This .DEF file (DISKCTLR.DEF) was created by editing CONTROLR.DEF;
; by adding DEF and EQU statements, deleting some others, and by
; changing the basic microword format.  The bulk of the effort required
; to create such a file was considerably reduced by beginning from the
; "master" file (CONTROLR.DEF) rather than typing a new file from scratch.
;
; This particular .DEF file was created for a specific Am29116-Am9520
; disk controller, described in the AMD application note:
; "A High-Performance Intelligent Disk Controller," by Otis Tabler and
; Brad Kitson, to be released by AMD in early 1982.  The source file
; is DISKCTLR.SRC.
;
; The major difference between this DEF file and the CONTROLR.DEF file is
; the approach to the microprogramming.  This file makes heavy use of
; DEF statement overlays while the other uses the comma-positional
; notation.  The choice is a matter of preference. THE Am29116 MNEMONICS
; AND INSTRUCTION LAYOUT ARE IDENTICAL IN THESE FILES.
;
;
; This file may also be used as a master file which the user can edit to
; suit his/her application.
;
; Anyone finding an error in this file is requested to send a marked listing
; or portion thereof to: AMD APPLICATIONS     or  AMD CUSTOMER EDUCATION CENTER
;                        PO BOX 453  MS#70        PO BOX 453  MS#71
;                        SUNNYVALE, CA 94086      490-A LAKESIDE DRIVE
;                                                 SUNNYVALE, CA 94086
;
; Advanced Micro Devices reserves the right to make changes in its product
; without notice in order to improve design or performance characteristics.
; The company assumes no responsibility for the use of any circuits or
; programs described herein.
;
;
; Am29116 Mnemonics Copyright (c) 1982 Advanced Micro Devices, Inc.
;
;
```

```
;
;
WORD    80
;
; ************************************************
; GENERAL MNEMONICS
; ************************************************
;
; BYTE - WORD MODE SELECT [M] <---------- referenced by DEF statements
;
B:              EQU             1B#0    ; BYTE MODE
W:              EQU             1B#1    ; WORD MODE
;
;
; ************************************************
; N SELECT [N]
;
N0:             EQU             H#0     ; 0
N1:             EQU             H#1     ;
N2:             EQU             H#2     ;
N3:             EQU             H#3     ;
N4:             EQU             H#4     ;
N5:             EQU             H#5     ;
N6:             EQU             H#6     ;
N7:             EQU             H#7     ;
N8:             EQU             H#8     ;
N9:             EQU             H#9     ;
NA:             EQU             H#A     ;
NB:             EQU             H#B     ;
NC:             EQU             H#C     ;
ND:             EQU             H#D     ;
NE:             EQU             H#E     ;
NF:             EQU             H#F     ;
```

```
;
; ***************************************************
; 32 RAM REGISTERS [R]
;
R0:             EQU             5D#0    ; 00000
R1:             EQU             5D#1    ;
R2:             EQU             5D#2    ;
R3:             EQU             5D#3    ;
R4:             EQU             5D#4    ;
R5:             EQU             5D#5    ;
R6:             EQU             5D#6    ;
R7:             EQU             5D#7    ;
R8:             EQU             5D#8    ;
R9:             EQU             5D#9    ;
R10:            EQU             5D#10   ;
R11:            EQU             5D#11   ;
R12:            EQU             5D#12   ;
R13:            EQU             5D#13   ;
R14:            EQU             5D#14   ;
R15:            EQU             5D#15   ;
R16:            EQU             5D#16   ;
R17:            EQU             5D#17   ;
R18:            EQU             5D#18   ;
R19:            EQU             5D#19   ;
R20:            EQU             5D#20   ;
R21:            EQU             5D#21   ;
R22:            EQU             5D#22   ;
R23:            EQU             5D#23   ;
R24:            EQU             5D#24   ;
R25:            EQU             5D#25   ;
R26:            EQU             5D#26   ;
R27:            EQU             5D#27   ;
R28:            EQU             5D#28   ;
R29:            EQU             5D#29   ;
R30:            EQU             5D#30   ;
R31:            EQU             5D#31   ;
;
```

```
;
;
; ****************************
; SINGLE OPERAND INSTRUCTIONS
; ****************************
;
; OPCODES [1]
;
MOVE:          EQU          H#C     ; 1100 MOVE
COMP:          EQU          H#D     ; 1101 COMP
INC:           EQU          H#E     ; 1110 INC     INCREMENT
NEG:           EQU          H#F     ; 1111 NEG     INCREMENT COMP
;
; SOURCE-DESTINATION SELECT [2]
;
SORA:          EQU          H#0     ; RAM   ACC
SORY:          EQU          H#2     ; RAM   Y BUS
SORS:          EQU          H#3     ; RAM   STATUS
SOAR:          EQU          H#4     ; ACC   RAM
SODR:          EQU          H#6     ; D     RAM
SOIR:          EQU          H#7     ; I     RAM
SOZR:          EQU          H#8     ; 0     RAM
SOZER:         EQU          H#9     ; D(0E) RAM
SOSER:         EQU          H#A     ; D(SE) RAM
SORR:          EQU          H#B     ; RAM   RAM
;
; **********************************************************
SOR: DEF 1V,  B#10,4V%D#,    4V%D#,         5V%D#,64X   ; SINGLE OPERAND RAM
;          \            \      \             \
;            MODE,QUAD,OPCODE,SOURCE-DEST,REGISTER
;             [M]          [1]       [2]         [R]    <---- refer to proper EQU groups
; **********************************************************
```

```
;
; SOURCE (R/S) [3]
;
SOA:            EQU            H#4      ; ACC
SOD:            EQU            H#6      ; D
SOI:            EQU            H#7      ; I
SOZ:            EQU            H#8      ; 0
SOZE:           EQU            H#9      ; D(0E)
SOSE:           EQU            H#A      ; D(SE)
;
; DESTINATION [4]
;
NRY:            EQU            D#0      ; Y BUS
NRA:            EQU            D#1      ; ACC
NRS:            EQU            D#4      ; STATUS
NRAS:           EQU            D#5      ; ACC,STATUS
;
; ****************************************************************
SONR: DEF 1V,  B#11,4V%D#,    4V%D#,    5V%D#,64X        ; SINGLE OPERAND NON-RAM
;
;               MODE,QUAD,OPCODE,SOURCE,DESTINATION
;               [M]         [1]    [3]       [4]
; ****************************************************************
```

```
;
;
;
; **********************************
; TWO OPERAND INSTRUCTIONS
; **********************************
;
; OPCODES [5]
;
SUBR:           EQU         H#0     ; S minus R
SUBRC:          EQU         H#1     ; S minus R with carry
SUBS:           EQU         H#2     ; R minus S
SUBSC:          EQU         H#3     ; R minus S with carry
ADD:            EQU         H#4     ; R plus S
ADDC:           EQU         H#5     ; R plus S with carry
AND:            EQU         H#6     ; R . S
NAND:           EQU         H#7     ; R . S
EXOR:           EQU         H#8     ; R   S
NOR:            EQU         H#9     ; R + S
OR:             EQU         H#A     ; R + S
EXNOR:          EQU         H#B     ; R   S
;
;
; SOURCE-DESTINATION [6]            ; R     S     DEST
;
TORAA:          EQU         H#0     ; RAM   ACC   ACC
TORIA:          EQU         H#2     ; RAM   I     ACC
TODRA:          EQU         H#3     ; D     RAM   ACC
TORAY:          EQU         H#8     ; RAM   ACC   Y BUS
TORIY:          EQU         H#A     ; RAM   I     Y BUS
TODRY:          EQU         H#B     ; D     RAM   Y BUS
TORAR:          EQU         H#C     ; RAM   ACC   RAM
TORIR:          EQU         H#E     ; RAM   I     RAM
TODRR:          EQU         H#F     ; D     RAM   RAM
;
; ***********************************************************
TOR1: DEF 1V,  B#00,4V%D#,       4V%D#,    5V%D#,64X   ; TWO OPERAND RAM (1)
;
;               MODE,QUAD,SOURCE-DEST,OPCODE,REGISTER
;               [M]          [6]     [5]    [R]
; ***********************************************************
```

```
;
;
; SOURCE-DESTINATION [7]                          R      S        DEST
;
TODAR:              EQU             H#1      ; D      ACC      RAM
TOAIR:              EQU             H#2      ; ACC    I        RAM
TODIR:              EQU             H#5      ; D      I        RAM
;
; ***********************************************************
TOR2: DEF 1V,  B#10,4V%D#,        4V%D#,    5V%D#,64X    ; TWO OPERAND RAM (2)
;
;                   MODE,QUAD,SOURCE-DEST,OPCODE,REGISTER
;                   [M]           [7]       [5]    [R]
; ***********************************************************
;
; SOURCE    [8]                                  R      S
;
TODA:               EQU             H#1      ; D      ACC
TOAI:               EQU             H#2      ; ACC    I
TODI:               EQU             H#5      ; D      I
;
; **********************************************************
TONR: DEF 1V,   B#11,4V%D#,    4V%D#,    5V%D#,64X       ; TWO OPERAND NON-RAM
;
;                   MODE, QUAD,SOURCE,OPCODE,DESTINATION
;                   [M]          [8]    [5]     [4]
; **********************************************************
; ***************************************************
; SHIFT INSTRUCTIONS
; ***************************************************
;
; DIRECTION AND INPUT [9]
;
SHUPZ:          EQU             H#0     ; UP 0
SHUP1:          EQU             H#1     ; UP 1
SHUPL:          EQU             H#2     ; UP QLINK
SHDNZ:          EQU             H#4     ; DOWN 0
SHDN1:          EQU             H#5     ; DOWN 1
SHDNL:          EQU             H#6     ; DOWN QLINK
SHDNC:          EQU             H#7     ; DOWN QC
SHDNOV:         EQU             H#8     ; DOWN QN QOVR
;
;
; SOURCE [10]
;
SHRR:           EQU             H#6     ; RAM   RAM
SHDR:           EQU             H#7     ; D     RAM
;
;
; ******************************************************
SHFTR: DEF 1V,  B#10,4V%D#,         4V%D#,    5V%D#,64X  ; SHIFT RAM
;
;               MODE,QUAD,SOURCE,DIRECT-INPT,REGISTER
;               [M]        [10]        [9]     [R]
; ******************************************************
;
;
; SOURCE [11]
;
SHA:            EQU             H#6     ; ACC
SHD:            EQU             H#7     ; D
;
;
; ******************************************************
SHFTNR: DEF 1V,  B#11,4V%D#,    4V%D#,    5V%D#,64X      ; SHIFT NON-RAM
;
;               MODE,QUAD,SOURCE,DIRECT-INP,DESTINATION
;               [M]         [11]   [9]        [4](NRY; NRA ONLY)
; ******************************************************
```

```
;
; ***********************************************
;ROTATE INSTRUCTIONS
; ***********************************************
;
; SOURCE-DESTINATION [12]
;
RTRA:           EQU             H#C       ; RAM   ACC
RTRY:           EQU             H#E       ; RAM   Y BUS
RTRR:           EQU             H#F       ; RAM   RAM
;
;
; *************************************************
ROTR1: DEF  1V,  B#00,4V%D#,4V%D#,        5V%D#,64X      ; ROTATE RAM (1)
;
;               MODE,QUAD,N,SOURCE-DEST,REGISTER
;               [M]      [N]     [12]     [R]
; *************************************************
;
; SOURCE-DESTINATION [13]
;
RTAR:           EQU             H#0       ; ACC   RAM
RTDR:           EQU             H#1       ; D     RAM
;
;
; *************************************************
ROTR2: DEF  1V,  B#01,4V%D#,4V%D#,        5V%D#,64X      ; ROTATE RAM (2)
;
;               MODE,QUAD,N,SOURCE-DEST,REGISTER
;               [M]      [N]    [13]      [R]
; *************************************************
```

```
;
; SOURCE DESTINATION [14]
;
RTDY:           EQU             D#24    ; D     Y BUS
RTDA:           EQU             D#25    ; D     ACC
RTAY:           EQU             D#28    ; ACC   Y BUS
RTAA:           EQU             D#29    ; ACC   ACC
;
;
; ****************************************************
ROTNR: DEF  1V,  B#11,4V%D#,H#C,      5V%D#,64X  ; ROTATE NON-RAM
;
;               MODE,QUAD,N,FIXED CODE,DESTINATION
;               [M]     [N]             [14]
; ****************************************************
; ***************************************************
; BIT ORIENTED INSTRUCTIONS
; ***************************************************
;
; OPCODES [15]
;
SETNR:          EQU             H#D     ; SET RAM, BIT N
RSTNR:          EQU             H#E     ; RESET RAM, BIT N
TSTNR:          EQU             H#F     ; TEST RAM, BIT N
;
;
; ********************************************************
BOR1: DEF 1V,  B#11,4V%D#,4V%D#,   5V%D#,64X     ; BIT ORIENTED RAM (1)
;
;               MODE,QUAD,N,OPCODE,REGISTER
;               [M]      [N]  [15]    [R]
; ********************************************************
;
;
; OPCODES [16]
;
LD2NR:          EQU             H#C     ; 2^N --- RAM
LDC2NR:         EQU             H#D     ; 2^N --- RAM
A2NR:           EQU             H#E     ; RAM + 2^N - RAM
S2NR:           EQU             H#F     ; RAM - 2^N - RAM
;
;
; ********************************************************
BOR2: DEF 1V,  B#10,4V%D#,4V%D#,   5V%D#,64X     ; BIT ORIENTED RAM (2)
;
;               MODE,QUAD,N,OPCODE,REGISTER
;               [M]      [N]  [16]    [R]
; ********************************************************
```

```
;
; OPCODES [17]
;
TSTNA:          EQU             D#0     ; TEST ACC, BIT N
RSTNA:          EQU             D#1     ; RESET ACC, BIT N
SETNA:          EQU             D#2     ; SET ACC, BIT N
A2NA:           EQU             D#4     ; ACC + 2^N -- ACC
S2NA:           EQU             D#5     ; ACC - 2^N --ACC
LD2NA:          EQU             H#6     ; 2^N -- ACC
LDC2NA:         EQU             D#7     ; 2^N -- ACC
TSTND:          EQU             D#16    ; TEST D, BIT N
RSTND:          EQU             D#17    ; RESET D, BIT N
SETND:          EQU             D#18    ; SET D, BIT N
A2NDY:          EQU             D#20    ; D + 2^N -- Y BUS
S2NDY:          EQU             D#21    ; D - 2^N -- Y BUS
LD2NY:          EQU             D#22    ; 2^N -- Y BUS
LDC2NY:         EQU             D#23    ; 2^N -- Y BUS
;
;
; ********************************************************
BONR: DEF 1V,  B#11,4V%D#,B#1100,    5V%D#,64X   ; BIT ORIENTED NON-RAM
;
;               MODE,QUAD,N,FIXED CODE,OPCODE
;               [M]     [N]            [17]
; ********************************************************
```

```
 ; **************************************************
 ; ROTATE AND MERGE
 ; **************************************************
 ;
 ; SOURCE-DEST SELECT [U,S,MASK-DEST] [18]
 ;
 ;                                                 ROT   NON-ROT  MASK-DEST
 MDAI:           EQU             H#7       ; D     ACC      I
 MDAR:           EQU             H#8       ; D     ACC      RAM
 MDRI:           EQU             H#9       ; D     RAM      I
 MDRA:           EQU             H#A       ; D     RAM      ACC
 MARI:           EQU             H#C       ; ACC   RAM      I
 MRAI:           EQU             H#E       ; RAM   ACC      I
 ;
 ;
 ; **********************************************************
 ROTM: DEF 1V,  B#01,4V%D#,4V%D#,        5V%D#,64X        ;ROTATE AND MERGE
 ;
 ;               MODE,QUAD,N,SOURCE-DEST,REGISTER
 ;               [M]      [N]    [18]         [R]
 ; **********************************************************
 ;
 ; **************************************************
 ; ROTATE AND COMPARE
 ; **************************************************
 ;
 ; ROT.SRC(U)-NON ROT.SRC(S)/DEST-MASK(S)[19]
 ;
 CDAI:           EQU             H#2       ; D     ACC      I
 CDRI:           EQU             H#3       ; D     RAM      I
 CDRA:           EQU             H#4       ; D     RAM      ACC
 CRAI:           EQU             H#5       ; RAM   ACC      I
 ;
 ;
 ; ********************************************
 ROTC: DEF 1V,  B#01,4V%D#,4V%D#,             5V%D#,64X    ; ROTATE AND COMPARE
 ;
 ;               MODE,QUAD,N,SOURCE-DEST-MASK,REGISTER
 ;               [M]      [N]        [19]         [R]
 ; *********************************************
```

```
; ***********************************************
; PRIORITIZE
; ***********************************************
;
; SOURCE [20]
;
PRT1A:          EQU            H#7      ; ACC
PR1D:           EQU            H#9      ; D
;
;
; DESTINATION [21]
;
PR1A:           EQU            H#8      ; ACC
PR1Y:           EQU            H#A      ; Y BUS
PR1R:           EQU            H#B      ; RAM
;
; ********************************************
PRT1: DEF 1V,  B#10,4V%D#,          4V%D#,    5V%D#,64X         ; RAM ADDR MASK(S)
;
;               MODE,QUAD,DESTINATION,SOURCE,REG-MASK
;               [M]           [21]      [20]     [R]
; ********************************************
;
;
; DESTINATION [23]
;
PR2A:           EQU            H#0      ; ACC
PR2Y:           EQU            H#2      ; Y BUS
;
; MASK (S) [22]
;
PRA:            EQU            H#8      ; ACC
PRZ:            EQU            H#A      ; 0
PRI:            EQU            H#B      ; I
;
;
; ********************************************
PRT2: DEF 1V,  B#10,4V%D#,  4V%D#,  5V%D#,64X    ; PRIORITIZE RAM
;
;               MODE,QUAD,MASK,DEST,REG-SOURCE
;               [M]        [22] [23]    [R]
; ********************************************
```

```
 ; SOURCE (R) [24]
 ;
 PR3R:           EQU              H#3     ; RAM
 PR3A:           EQU              H#4     ; ACC
 PR3D:           EQU              H#6     ; D
 ;
 ;
 ; ***********************************************
 PR3T: DEF 1V,  B#10,4V%D#,  4V%D#,    5V%D#,64X  ; PRIORITIZE RAM
 ;
 ;               MODE,QUAD,MASK,SOURCE,REG-DEST
 ;               [M]       [22]  [24]    [R]
 ; ***********************************************
 ;
 ;
 ; SOURCE (R) [25]
 ;
 PRTA:           EQU              H#4     ; ACC
 PRTD:           EQU              H#6     ; D
 ;
 ;
 ; **********************************************
 PRTNR: DEF  1V,  B#11,4V%D#,  4V%D#,    5V%D#,64X        ; PRIORITIZE NON-RAM
 ;
 ;                  MODE,QUAD,MASK,SOURCE,DESTINATION
 ;                  [M]       [22]  [25]    [4](NRY,NRA ONLY)
 ; **********************************************
 ;
 ; ***********************************************
 ; CYCLIC REDUNDANCY CHECK
 ; ***********************************************
 ;
 ; ********************************************
CRCF: DEF B#11001100011,5V%D#,64X        ; FORWARD
 ; ********************************************
 ;
 ; ********************************************
CRCR: DEF B#11001101001,5V%D#,64X        ; REVERSE
 ; ********************************************
 ;
 ; *********************************************
 ;
 ; NOOP
 ;
 ; *********************************************
NOOP: DEF H#7140,64X    ; NO OPERATION
 ; *********************************************
```

```
;
; ***************************************************
; STATUS
; ***************************************************
;
; OPCODE [26]
;
SONZC:          EQU             5D#3    ; SET OVR,N,C,Z
SL:             EQU             5D#5    ; SET LINK
SF1:            EQU             5D#6    ; SET FLAG 1
SF2:            EQU             5D#9    ; SET FLAG 2
SF3:            EQU             5D#10   ; SET FLAG 3
;
;
; ***************************************************
SETST: DEF B#011,H#BA,5V%D#,64X  ; SET STATUS
;
;                          OPCODE
;                           [26]
; ***************************************************
```

```
;
; OPCODE [27]
;
RONCZ:          EQU             D#3     ; RESET OVR,N,C,Z
RL:             EQU             D#5     ; RESET LINK
RF1:            EQU             D#6     ; RESET FLAG 1
RF2:            EQU             D#9     ; RESET FLAG 2
RF3:            EQU             D#10    ; RESET FLAG 3
;
;*****************************************************
RSTST: DEF B#011,H#AA,5V%D#,64X         ; RESET STATUS
;
;                           OPCODE
;                           [27]
; ****************************************************
;
; ****************************************************
SVSTR: DEF  1V,  B#10,H#7A, 5V%D#,64X   ; SAVE STATUS-RAM
;
;                    MODE,QUAD,FIXED,RAM ADDRESS/DEST
;                     [M]                [R]
; ****************************************************
;
;
;*****************************************************
SVSTNR: DEF  1V, B#11,H#7A, 5V%D#,64X    ; SAVE STATUS NON-RAM
;
;                    MODE,QUAD,FIXED,DESTINATION
;                    [M]            [4](NRY,NRA ONLY)
; ****************************************************
```

```
;
; **************************************************
; TEST STATUS
; **************************************************
;
; OPCODE (CT)
;
TNOZ:           EQU             D#0     ; TEST (N OVR) + Z
TNO:            EQU             D#2     ; TEST N OVR
TZ:             EQU             D#4     ; TEST Z
TOVR:           EQU             D#6     ; TEST OVR
TLOW:           EQU             D#8     ; TEST LOW
TC:             EQU             D#10    ; TEST C
TZC:            EQU             D#12    ; TEST Z + C
TN:             EQU             D#14    ; TEST N
TL:             EQU             D#16    ; TEST LINK
TF1:            EQU             D#18    ; TEST FLAG 1
TF2:            EQU             D#20    ; TEST FLAG 2
TF3:            EQU             D#22    ; TEST FLAG 3
;
;
; ******************************************************
TEST: DEF B#011,H#9A,5V%D#,64X    ;  TEST STATUS
;
;                FIXED,     OPCODE
;                            [CT]
; ******************************************************
```

```
;
; added DEF and EQU statements
;********************************
;
;        IMMEDIATE OPERAND
;
IMME:    DEF      16V%D#, 64X
;
;        CT MULTIPLEXER CONTROL
;
CT:      DEF      16X, 4V%D#, 60X
NOZ:     EQU      H#0
NO:      EQU      H#1
Z:       EQU      H#2
OVR:     EQU      H#3
LOW:     EQU      H#4
C:       EQU      H#5
ZC:      EQU      H#6
N:       EQU      H#7
L:       EQU      H#8
F1:      EQU      H#9
F2:      EQU      H#A
F3:      EQU      H#B
;
;        STATUS REGISTER ENABLE
;
SRE:     DEF      20X, B#1, 59X
NOSRE:   DEF      20X, B#0, 59X
;
;        OUTPUT ENABLE Y
;
OEY:     DEF      21X, B#0, 58X
NOOEY:   DEF      21X, B#1, 58X
;
;        INSTRUCTION ENABLE
;
IEN:     DEF      22X, B#0, 57X
NOIEN:   DEF      22X, B#1, 57X
;
;        D-I-LATCH ENABLE
;
DLE:     DEF      23X, B#1, 56X
NODLE:   DEF      23X, B#0, 56X
```

```
 ;
 ;
 ;
 ;------------------------------------------------------
 ;     Am2910 COMMANDS AND BRANCH ADDRESSES
 ; note use of DEF statements - overlay in SRC file
 ;------------------------------------------------------
JZ:     DEF     24X, H#0, 10V$D#1023, 42X
CJS:    DEF     24X, H#1, 10V$D#1023, 42X
JS:     DEF     24X, H#1, 10V$D#1023, 6Q#36, 36X        ;  UNCONDITIONAL JUMP TO SUBR.
JMAP:   DEF     24X, H#2, 10V$D#1023, 42X
CJP:    DEF     24X, H#3, 10V$D#1023, 42X
JP:     DEF     24X, H#3, 10V$D#1023, 6Q#36, 36X        ;  UNCONDITIONAL JUMP
PUSH:   DEF     24X, H#4, 10V$D#1023, 42X
JSRP:   DEF     24X, H#5, 10V$D#1023, 42X
CJV:    DEF     24X, H#6, 10V$D#1023, 42X
JRP:    DEF     24X, H#7, 10V$D#1023, 42X
RFCT:   DEF     24X, H#8, 10V$D#1023, 42X
RPCT:   DEF     24X, H#9, 10V$D#1023, 42X
CRTN:   DEF     24X, H#A, 10V$D#1023, 42X
RTN:    DEF     24X, H#A, 10V$D#1023, 6Q#36, 36X        ;  UNCONDITIONAL RETURN
CJPP:   DEF     24X, H#B, 10V$D#1023, 42X
LDCT:   DEF     24X, H#C, 10V$D#1023, 42X
LOOP:   DEF     24X, H#D, 10V$D#1023, 42X
CONT:   DEF     24X, H#E, 10V$D#1023, 42X
TWB:    DEF     24X, H#F, 10V$D#1023, 42X
 ;                        ^
 ;                        |
 ;                        |
 ; NOTE: For proper assembly, a "$" must be used in any field which
 ; will be used to accept a symbolic address in the SRC file.
 ;
 ;
 ;
```

```
;
;
;
;
;       Am2910 CONDITION CODE SELECTIONS
;
IF:       DEF     38X, 5V%D#, B#0, 36X
IFNOT:    DEF     38X, 5V%D#, B#1, 36X
;
;
AE20:     EQU     5Q#10:              ; AM9520 ALIGNMENT ERROR FLAG
CT16:     EQU     5Q#11:              ; AM29116 CONDITIONAL TEST FLAG
EP20:     EQU     5Q#12:              ; AM9520 ERROR PATTERN FLAG
ER20:     EQU     5Q#13:              ; AM9520 ERROR DETECTED FLAG
FAIL:     EQU     5Q#14:              ; UNCONDITIONAL FAILURE OF "TEST"
RDYI:     EQU     5Q#15:              ; NOT READY INPUT (DATA UNAVAILABLE FROM FIFOS)
RDYO:     EQU     5Q#16:              ; NOT READY OUTPUT (FIFOS FULL)
SUCC:     EQU     5Q#17:              ; UNCONDITIONAL SUCCESS OF "TEST"
ATTN:     EQU     5Q#20:              ; ATTENTION
BACK:     EQU     5Q#21:              ; BUS ACKNOWLEDGE
BUSY:     EQU     5Q#22:              ; BUSY
INDX:     EQU     5Q#23:              ; INDEX
SAMD:     EQU     5Q#24:              ; SECTOR / ADDRESS MARK DETECTED
PM2:      EQU     5Q#25:              ; AM9520 PATTERN MATCH 2 FLAG
PM3:      EQU     5Q#26:              ; AM9520 PATTERN MATCH 3 FLAG
PM4:      EQU     5Q#27:              ; AM9520 PATTERN MATCH 4 FLAG
```

```
;
;       MISCELLANEOUS CONTROL SIGNALS
;
ADMC:   DEF     44X, B#0, 35X   ; ADDRESS MARK CONTROL
BFCB:   DEF     45X, B#0, 34X   ; MEMORY BUS FROM DRIVE CONTROL BUS
BFTP:   DEF     46X, B#0, 33X   ; MEMORY BUS FROM TRANSLATE PROM
BF03:   DEF     47X, B#0, 32X   ; MEMORY BUS FROM 9403AS
BF16:   DEF     48X, B#0, 31X   ; MEMORY BUS FROM AM29116
BF2L:   DEF     49X, B#0, 30X   ; MEMORY BUS FROM AM9520 - LOWER BYTE
BF2U:   DEF     50X, B#0, 29X   ; MEMORY BUS FROM AM9520 - UPPER BYTE
BOUT:   DEF     51X, B#0, 28X   ; (DISK) BUS DIRECTION OUT (FROM CONTROLLER)
BT03:   DEF     52X, B#0, 27X   ; MEMORY BUS TO 9403AS
BT16:   DEF     53X, B#0, 26X   ; MEMORY BUS TO AM29116
BT2L:   DEF     54X, B#0, 25X   ; MEMORY BUS TO AM9520 - LOWER BYTE
BT2U:   DEF     55X, B#0, 24X   ; MEMORY BUS TO AM9520 - UPPER BYTE
BT20:   DEF     56X, B#0, 23X   ; MEMORY BUS TO AM9520 - CONTROL INFORMATION
CE2L:   DEF     57X, B#0, 22X   ; CLOCK ENABLE AM9520 TO LOWER-BYTE BUS INT.
CE20:   DEF     58X, B#0, 21X   ; CLOCK ENABLE MEMORY BUS TO AM9520 TRANSFER
CP20:   DEF     59X, B#0, 20X   ; CLOCK PULSE (ACTUAL WAVEFORM) FOR AM9520
CREQ:   DEF     60X, B#0, 19X   ; COMMAND REQUEST
INPT:   DEF     61X, B#0, 18X   ; INPUT SERIAL DATA TO 9403AS
JMPI:   DEF     62X, B#01, 16X  ; JUMP INDIRECT AM29116 REGISTER
NOJMPI: DEF     62X, B#10, 16X  ; NO INDIRECT JUMP
MADR:   DEF     64X, B#0, 15X   ; MEMORY ACCESS
MREA:   DEF     65X, B#0, 14X   ; MEMORY ADDRESS
MWRT:   DEF     66X, B#0, 13X   ; MEMORY WRITE
OUPT:   DEF     67X, B#0, 12X   ; OUTPUT SERIAL DATA FROM 9403AS
PENB:   DEF     68X, B#0, 11X   ; PARAMETER ENABLE
PFPM:   DEF     69X, B#0, 10X   ; SET 9520 P BITS FROM 9520 PM BITS
PF03:   DEF     70X, B#0, 9X    ; PARALLEL FETCH FROM 9403AS
PL03:   DEF     71X, B#0, 8X    ; PARALLEL LOAD INTO 9403AS
PREQ:   DEF     72X, B#0, 7X    ; PARAMETER REQUEST
RDGA:   DEF     73X, B#0, 6X    ; READ GATE
RFIF:   DEF     74X, B#0, 5X    ; RESET FIFO
SAST:   DEF     75X, B#0, 4X    ; SELECT / ATTENTION STROBE
WRGA:   DEF     76X, B#0, 3X    ; WRITE GATE
```

```
;
ASCEBC: EQU     Q#0             ; ASCII TO EBCDIC SUBSET PREFIX
BCDEBC: EQU     Q#1             ; BCD TO EBCDIC SUBSET PREFIX
EBCASC: EQU     Q#2             ; EBCDIC SUBSET TO ASCII PREFIX
EBCBCD: EQU     Q#3             ; EBCDIC SUBSET TO BCD PREFIX
;
XLAT:   DEF     77X, 3V%D#      ; TRANSLATE PREFIX
;
;
        END

TOTAL PHASE 1 ERRORS =   0
```

```
AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4
AM29116 / AM9520 - BASED DISK CONTROLLER

       ;
       ; CREATED 9/81 TABLER-KITSON
       ;
       ;
       ;
       ; This SRC file was created for the AMD application note:
       ; "A High-Performance Intelligent Disk Controller"
       ; by Otis Tabler and Brad Kitson.
       ;
       ; Mnemonics and word format are defined in DISKCTLR.DEF
       ;
       ; Advanced Micro Devices reserves the right to make changes in its
       ; product without notice in order to improve design or performance
       ; characteristics.  The company assumes no responsibility for the
       ; use of any circuits or programs described herein.
       ;
       ; Am29116 Mnemonics Copyright (c) 1982 Advanced Micro Devices
       ;
       ;
       ;
       ;
```

```
;
;    SECTOR READ / WRITE SUBROUTINE
;*******************************************
;
;       INPUTS:
;
;               FUNCTION CODE IN R0:
;
;                       0 TO READ SECTOR
;
;                       +1 TO WRITE SECTOR
;
;               HEAD NUMBER IN MSB OF R1
;
;               MSB OF TRACK NUMBER IN LSB OF R1
;
;               LSB OF TRACK NUMBER IN MSB OF R2
;
;               SECTOR NUMBER IN LSB OF R2
;
;               START ADDRESS OF RAM SECTOR BUFFER IN R3
;
```

```
        ;
        ;       OUTPUT:
        ;
        ;               R0 CONTAINS:
        ;
        ;                       0       IF THE FUNCTION SPECIFIED WAS COMPLETED
        ;                               EITHER WITHOUT ERROR OR WITH A
        ;                               SUCCESSFULLY CORRECTED READ ERROR
        ;
        ;                       +1      IF THE SECTOR'S HEADER IS BAD
        ;
        ;                       +2      IF AN UNCORRECTABLE ERROR WAS DETECTED IN
        ;                               READING THE SECTOR'S DATA SEGMENT
        ;
```

AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4
AM29116 / AM9520 - BASED DISK CONTROLLER

```
        ;
        ;  ADDITIONAL MNEMONICS
        ; *********************
        ;
        ;
 C001 CRCMSK: EQU     16H#C001   ; CRCF POLYNOMIAL MASK
 0010 CRCNIT: EQU     16         ; CRCF NUMBER OF ITERATIONS (D#16 <-- default bas
      NSPASS: EQU     64         ; NUMBER OF SECTOR PASSES (SET THIS EQUAL
 0040 ;                            TO THE NUMBER OF SECTORS PER TRACK.)
      RDITCT: EQU     65         ; READ ITERATION COUNT, EQUAL TO THE NUMBER
      ;                          ; OF 16-BIT WORDS (DATA PLUS MODIFIED FIRE
 0041 ;                          ; CODE) PER SECTOR, DIVIDED BY TWO, MINUS 1.
 0016 PF1:    EQU     22         ; PERIOD FACTOR ONE
 000D PF2:    EQU     13         ; PERIOD FACTOR TWO
 0059 PF3:    EQU     89         ; PERIOD FACTOR THREE
      PF4:    EQU     23         ; PERIOD FACTOR FOUR
 0017 ;
 E723 A1LSW:  EQU     H#E723     ; A1 CONSTANT(LEAST SIG. WORD)
 0006 A1MSW:  EQU     6          ; A1 CONS.(MOST SIG. WORD)
 BFA8 A2LSW:  EQU     H#BFA8     ; A2 CONS.(LEAST SIG. WORD)
 D530 A3LSW:  EQU     H#D530     ; A3' CONS.(LEAST SIG. WORD)
      A4LSW:  EQU     H#A928     ; A4' CONS.(LEAST SIG. WORD)
 A928 ;
 7100 KL128:  EQU     H#7100     ; K(LEAST SIG. WORD) SHIFTED UP
                                 ; BY SEVEN PLACES
 0477 KM128:  EQU     H#0477     ; K(MOST SIG. WORD) SHIFTED UP
                                 ; BY SEVEN PLACES.
 EEE2 KLSW:   EQU     H#EEE2     ; K(LEAST SIG. WORD)
      KMSW:   EQU     8          ; K(MOST SIG. WORD)
 0008 ;
```

```
     ;
     ;
     ;      IF THE FUNCTION CODE IN R0 EQUALS +1 (WRITE SECTOR), PRECALCULATE
     ;      THE MODIFIED FIRE CODE'S PARTIAL CHECKSUM FOR THE FIRST HALF OF
     ;      THE DATA SEGMENT TO BE WRITTEN.
     ;
0000 SECTIO: BOR2    W,0,S2NR,R0
     /      &NODLE  &NOIEN  &NOOEY  &NOSRE ;<------  note use of overlayed
     /      &CT     Z       &NOJMPI        ;         DEF statements
     /      &IFNOT  CT16    &CJP    CFCODE ;         signified by "&"
     ;
     ;
     ;      RESET THE AM9520 AND THEN PLACE IT IN COMPUTE CHECK BITS MODE.
     ;      INITIALIZE COUNTER FOR CHECK BITS PRECALCULATION LOOP.
     ;
0001        SONR    W,MOVE,SOI,NRY
     /      &NODLE  &NOIEN  &OEY    &NOSRE
     /      &NOJMPI
     /      &CONT
     ;
0002        IMME    H#0000
     /      &NODLE  &NOIEN  &OEY    &NOSRE
     /      &BT20   &NOJMPI
     /      &CONT
     ;
0003        SONR    W,MOVE,SOI,NRY
     /      &NODLE  &NOIEN  &OEY    &NOSRE
     /      &NOJMPI
     /      &CONT
     ;
0004        IMME    H#0010
     /      &NODLE  &NOIEN  &OEY    &NOSRE
     /      &BT20   &NOJMPI
     /      &LDCT   127
     ;
     ;
     ;      (R3) TO R4
     ;
0005        SOR     W,MOVE,SORY,R3
     /      &DLE    &NOIEN  &OEY    &NOSRE
     /      &NOJMPI
     /      &CONT
     ;
0006        SOR     W,MOVE,SODR,R4
     /      &NODLE  &IEN    &NOOEY  &NOSRE
     /      &NOJMPI
     /      &CONT
```

```
       ;
       ;       BEGIN CHECK BITS PRECALCULATION LOOP.
       ;       (R4) TO THE MAR.
       ;
 0007 PCPREL:  SOR     W,MOVE,SORY,R4
       /       &NODLE  &NOIEN  &OEY    &NOSRE
       /       &MADR   &NOJMPI
       /       &CONT
       ;
       ;
       ;       CLOCK THE LESS SIGNIFICANT BYTE OF ((MAR)) INTO THE AM9520.
       ;
               NOOP
 0008  /       &NODLE  &NOIEN  &NOOEY  &NOSRE
       /       &BT2L   &NOJMPI
       /       &CONT
       ;
               NOOP
 0009  /       &NODLE  &NOIEN  &NOOEY  &NOSRE
       /       &CP20   &NOJMPI
       /       &CONT
       ;
       ;
       ;       NOOP FOR TIMING PURPOSES
       ;
               NOOP
 000A  /       &NODLE  &NOIEN  &NOOEY  &NOSRE
       /       &NOJMPI
       /       &CONT
       ;
       ;
       ;       CLOCK THE MORE SIGNIFICANT BYTE OF ((MAR)) INTO THE AM9520.
       ;
               NOOP
 000B  /       &NODLE  &NOIEN  &NOOEY  &NOSRE
       /       &BT2U   &NOJMPI
       /       &CONT
       ;
       ;
       ;       INCREMENT (R4).
       ;       END CHECK BITS PRECALCULATION LOOP.
       ;
 000C          SOR     W,INC,SORR,R4
       /       &NODLE  &IEN    &NOOEY  &NOSRE
       /       &CP20   &NOJMPI
       /       &RPCT   PCPREL
```

```
      ;
      ;        PLACE THE AM9520 IN WRITE CHECK BITS MODE.
      ;        INITIALIZE COUNTER FOR STORE CHECK BITS IN BUFFER LOOP.
      ;
 000D          SONR    W,MOVE,SOI,NRY
      /        &NODLE  &NOIEN  &OEY    &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
 000E          IMME    H#0011
      /        &NODLE  &NOIEN  &OEY    &NOSRE
      /        &BT20   &NOJMPI
      /        &LDCT   2
      ;
      ;
      ;        BEGIN STORE CHECK BITS IN BUFFER LOOP.
      ;        (R4) TO THE MAR.
      ;        CLOCK OUT NEXT MODIFIED FIRE CODE BYTE TO THE
      ;        LESS-SIGNIFICANT MEMORY BUS INTERFACE REGISTER.
      ;
 000F SCBIBL:  SOR     W,MOVE,SORY,R4
      /        &NODLE  &NOIEN  &OEY    &NOSRE
      /        &CP20   &MADR   &NOJMPI
      /        &CONT
      ;
               NOOP
 0010 /        &NODLE  &NOIEN  &NOOEY  &NOSRE
      /        &CE2L   &NOJMPI
      /        &CONT
      ;
      ;
      ;        NOOP FOR TIMING PURPOSES
      ;
               NOOP
 0011 /        &NODLE  &NOIEN  &NOOEY  &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
      ;
      ;        CLOCK OUT NEXT MODIFIED FIRE CODE BYTE TO THE
      ;        MORE-SIGNIFICANT MEMORY BUS INTERFACE REGISTER.
      ;
               NOOP
 0012 /        &NODLE  &NOIEN  &NOOEY  &NOSRE
      /        &CP20   &NOJMPI
      /        &CONT
      ;
      ;
      ;        NOOP FOR TIMING PURPOSES
      ;
               NOOP
 0013 /        &NODLE  &NOIEN  &NOOEY  &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
```

```
        ;       (BUS INTERFACE REGISTER PAIR) TO (MAR).
        ;       INCREMENT (R4).
        ;       END STORE CHECK BITS IN BUFFER LOOP.
        ;
 0014           SOR     W,INC,SORR,R4
        /       &NODLE  &IEN    &NOOEY  &NOSRE
        /       &BF2L   &BF2U   &MWRT   &NOJMPI
        /       &RPCT   SCBIBL
        ;
        ;
        ;       ZERO THE UPPER BYTE OF (R5) AND THEN CLOCK THE 7TH AND LAST
        ;       BYTE OF THE MODIFIED FIRE CODE INTO ITS LOWER BYTE.
        ;
 0015           SOR     W,MOVE,SOZR,R5
        /       &NODLE  &IEN    &NOOEY  &NOSRE
        /       &CP20   &NOJMPI
        /       &CONT
        ;
                NOOP
 0016   /       &NODLE  &NOIEN  &NOOEY  &NOSRE
        /       &CE2L   &NOJMPI
        /       &CONT
        ;
 0017           SOR     B,MOVE,SODR,R5
        /       &DLE    &IEN    &NOOEY  &NOSRE
        /       &BF2L   &BT16   &NOJMPI
        /       &CONT
        ;
        ;
        ;       (R4) TO MAR.
        ;
 0018           SOR     W,MOVE,SORY,R4
        /       &NODLE  &NOIEN  &OEY    &NOSRE
        /       &MADR   &NOJMPI
        /       &CONT
        ;
        ;
        ;       (R5) TO (MAR).
        ;
 0019           SOR     W,MOVE,SORY,R5
        /       &NODLE  &NOIEN  &OEY    &NOSRE
        /       &BF16   &MWRT   &NOJMPI
        /       &CONT
        ;
```

```
      ;
      ;        CONVERT THE FUNCTION CODE IN R0 TO A MICROCODE BRANCH ADDRESS.
      ;
 001A CFCODE: SONR    W,MOVE,SOI,NRA
      /        &NODLE  &IEN    &NOOEY   &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
 001B          IMME    BRTABL
      /        &NODLE  &IEN    &NOOEY   &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
 001C          TOR1    W,ADD,TORAA,R0
      /        &NODLE  &IEN    &NOOEY   &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
      ;
      ;        CRCF POLYNOMIAL MASK TO ACC.
      ;
 001D          SONR    W,MOVE,SOI,NRA
      /        &NODLE  &IEN    &NOOEY   &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
 001E          IMME    CRCMSK
      /        &NODLE  &IEN    &NOOEY   &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
      ;
      ;        CLEAR REGISTER USED TO ACCUMULATE CRCF.
      ;
 001F          SOR     W,MOVE,SOZR,R4
      /        &NODLE  &IEN    &NOOEY   &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
      ;
      ;        COPY HEAD BYTE AND TRACK BYTE 1 TO R5.
      ;        SET CRCF LOOP COUNTER.
      ;
 0020          SOR     W,MOVE,SORY,R1
      /        &DLE    &IEN    &NOOEY   &NOSRE
      /        &NOJMPI
      /        &LDCT   CRCNIT
      ;
 0021          SOR     W,MOVE,SODR,R5
      /        &NODLE  &IEN    &NOOEY   &NOSRE
      /        &NOJMPI
      /        &CONT
```

```
       ;
       ;         SHIFT R5 AND SET QLINK.
       ;
 0022 CRCFL1: SHFTR   W,SHUPZ,SHRR,R5
      /          &NODLE  &IEN    &NOOEY  &NOSRE
      /          &NOJMPI
      /          &CONT
       ;
       ;
       ;         ACCUMULATE CRCF.
       ;
 0023            CRCF    R4
      /          &NODLE  &IEN    &NOOEY  &NOSRE
      /          &NOJMPI
      /          &RPCT   CRCFL1
       ;
       ;
       ;         COPY TRACK BYTE 2 AND SECTOR BYTE TO R5.
       ;         SET CRCF LOOP COUNTER.
       ;
 0024            SOR     W,MOVE,SORY,R2
      /          &DLE    &IEN    &NOOEY  &NOSRE
      /          &NOJMPI
      /          &LDCT   CRCNIT
       ;
 0025            SOR     W,MOVE,SODR,R5
      /          &NODLE  &IEN    &NOOEY  &NOSRE
      /          &NOJMPI
      /          &CONT
       ;
       ;
       ;         SHIFT R5 AND SET QLINK.
       ;
 0026 CRCFL2: SHFTR   W,SHUPZ,SHRR,R5
      /          &NODLE  &IEN    &NOOEY  &NOSRE
      /          &NOJMPI
      /          &CONT
       ;
       ;
       ;         ACCUMULATE CRCF.
       ;
 0027            CRCF    R4
      /          &NODLE  &IEN    &NOOEY  &NOSRE
      /          &NOJMPI
      /          &RPCT   CRCFL2
      ;
      ;        INITIALIZE SECTOR PASS LOOP COUNTER.
      ;        ENTER INPUT MODE.
      ;        TURN ON READ GATE.
      ;
      ;
               NOOP
 0028 /        &NODLE   &NOIEN   &NOOEY   &NOSRE
      /        &INPT    &NOJMPI  &RDGA
      /        &LDCT    NSPASS
      ;
      ;
      ;        BEGIN SECTOR PASS LOOP.
      ;        TURN ON ADDRESS MARK CONTROL.
      ;        RESET BYTE SYNC ACQUISITION CIRCUITRY AND FIFO ARRAY.
      ;
      SECTL1:  NOOP
 0029 /        &NODLE   &NOIEN   &NOOEY   &NOSRE
      /        &ADMC    &INPT    &NOJMPI  &RDGA    &RFIF
      /        &CONT
      ;
      ;
      ;        PASS WHEN ADDRESS MARK DETECTED.
      ;
               NOOP
 002A /        &NODLE   &NOIEN   &NOOEY   &NOSRE
      /        &ADMC    &INPT    &NOJMPI  &RDGA
      /        &IFNOT   SAMD     &CJP     $
      ;
      ;
      ;        TURN OFF ADDRESS MARK CONTROL.
      ;        PASS WHEN INPUT AVAILABLE FROM FIFO ARRAY.
      ;
               NOOP
 002B /        &NODLE   &NOIEN   &NOOEY   &NOSRE
      /        &INPT    &NOJMPI  &RDGA
      /        &IFNOT   RDYI     &CJP     $
```

```
       ;
       ;         INPUT RECORDED HEAD NUMBER AND MSB OF RECORDED TRACK NUMBER
       ;         TO R5 AND D-LATCH.
       ;         PASS WHEN INPUT AVAILABLE FROM FIFO ARRAY.
       ;
 002C            SOR      W,MOVE,SODR,R5
       /         &DLE     &IEN     &NOOEY   &NOSRE
       /         &BF03    &BT16    &INPT    &NOJMPI &RDGA
       /         &IFNOT   RDYI     &CJP     $
       ;
       ;
       ;         COMPARE THE CONTENTS OF R1 AND R5.
       ;         IF THEY DISAGREE, EXAMINE THE NEXT SECTOR.
       ;
 002D            TOR1     W,EXOR,TODRR,R1
       /         &NODLE   &NOIEN   &NOOEY   &SRE    &CT     Z
       /         &INPT    &NOJMPI &RDGA
       /         &IF      CT16     &CJP     SECTL2
       ;
       ;
       ;         INPUT LSB OF RECORDED TRACK NUMBER AND RECORDED SECTOR NUMBER
       ;         TO R5 AND D-LATCH.
       ;         PASS WHEN INPUT AVAILABLE FROM FIFO ARRAY.
       ;
                 NOOP
 002E  /         &NODLE   &NOIEN   &NOOEY   &NOSRE
       /         &PF03    &INPT    &NOJMPI &RDGA
       /         &CONT
       ;
 002F            SOR      W,MOVE,SODR,R5
       /         &DLE     &IEN     &NOOEY   &NOSRE
       /         &BF03    &BT16    &INPT    &NOJMPI &RDGA
       /         &IFNOT   RDYI     &CJP     $
```

```
        ;
        ;       COMPARE THE CONTENTS OF R2 AND R5;
        ;       IF THEY DISAGREE, EXAMINE THE NEXT SECTOR.
        ;
 0030           TOR1    W,EXOR,TODRR,R2
        /       &NODLE  &NOIEN  &NOOEY  &SRE    &CT     Z
        /       &PF03   &INPT   &NOJMPI &RDGA
        /       &IF     CT16    &CJP    SECTL2
        ;
        ;
        ;       INPUT RECORDED CRCF BYTES 1 AND 0 TO R5 AND D-LATCH.
        ;
 0031           SOR     W,MOVE,SODR,R5
        /       &DLE    &IEN    &NOOEY  &NOSRE
        /       &BF03   &BT16   &INPT   &NOJMPI &RDGA
        /       &CONT
        ;
        ;
        ;       COMPARE THE TWO CRCFS.
        ;       IF THEY AGREE, PROCEED TO READ OR WRITE AS SPECIFIED BY R0.
        ;       OTHERWISE, ASSUME BAD HEADER, TRUE ID UNKNOWN,
        ;       AND CONTINUE LOOP.
        ;
 0032           TOR1    W,EXOR,TODRR,R4
        /       &NODLE  &NOIEN  &NOOEY  &SRE    &CT     Z
        /       &INPT   &NOJMPI &RDGA
        /       &IF     CT16    &CJP    MATCH1
        ;
        ;
        ;       TURN OFF READ GATE.
        ;       LEAVE INPUT MODE.
        ;       END SECTOR PASS LOOP.
        ;       NOTICE WE HAVE THREE MICROINSTRUCTION CLOCKS LEFT BEFORE
        ;       IT IS TIME TO BEGIN WRITING OR RE-SYNC AND BEGIN READING.
        ;
 0033 SECTL2:   SOR     W,MOVE,SORY,R0
        /       &NODLE  &NOIEN  &NOOEY  &NOSRE
        /       &NOJMPI
        /       &RPCT   SECTL1
        ;
        ;
        ;       IF SECTOR SEARCH COUNT EXHAUSTED, LOAD +1 INTO R0 AND RETURN
        ;
 0034           BOR2    W,LD2NR,0,R0
        /       &NODLE  &IEN    &NOOEY  &NOSRE
        /       &NOJMPI
        /       &RTN
```

```
       ;
       ;         SUCCESSFUL MATCH.
       ;
 0035 MATCH1: SOR      W,MOVE,SORY,R0
       /      &NODLE   &NOIEN   &OEY     &NOSRE
       /      &JMPI
       /      &CONT
       ;
 0036 BRTABL: SOR      W,NEG,SORA,R3
       /      &NODLE   &IEN     &NOOEY   &NOSRE
       /      &NOJMPI  &RFIF
       /      &JP      RDSEC1
       ;
 0037         SOR      W,NEG,SORA,R3
       /      &NODLE   &IEN     &NOOEY   &NOSRE
       /      &NOJMPI  &RFIF
       /      &JP      WRSEC1
```

```
      ;
      ;       READ SECTOR
      ;
      ;       (R3) TO R4 AND MAR.
      ;       ENTER INPUT MODE AGAIN.
      ;       TURN READ GATE BACK ON.
      ;       INITIALIZE COUNTER FOR READ DATA SEGMENT LOOP.
      ;       DURING THAT LOOP, AM9520 READ HIGH SPEED IS PERFORMED ON THE
      ;       FIRST HALF OF THE SEGMENT BUFFER.  THE SECOND HALF OF THE BUFFER
      ;       IS PROCESSED BY THE AM9520 IN THE READ HIGH SPEED COMPLETION LOOP.
      ;
 0038 RDSEC1: SOR     W,NEG,SOAR,R4
      /       &NODLE  &IEN    &OEY    &NOSRE
      /       &INPT   &NOJMPI &MADR   &RDGA   &RFIF
      /       &LDCT   RDITCT
      ;
      ;
      ;       (R3) - 1 TO R5.
      ;       PASS WHEN INPUT AVAILABLE FROM FIFO ARRAY.
      ;
 0039         SOR     W,COMP,SOAR,R5
      /       &NODLE  &IEN    &NOOEY  &NOSRE
      /       &INPT   &NOJMPI &RDGA
      /       &IFNOT  RDYI    &CJP    $
```

```
       ;
       ;        RESET THE AM9520 AND THEN PLACE IT IN READ HIGH SPEED MODE.
       ;
 003A           SONR    W,MOVE,SOI,NRY
       /        &NODLE  &NOIEN  &OEY    &NOSRE
       /        &INPT   &NOJMPI &RDGA
       /        &CONT
       ;
 003B           IMME    H#0003
       /        &NODLE  &NOIEN  &OEY    &NOSRE
       /        &INPT   &NOJMPI &RDGA
       /        &CONT
       ;
 003C           SONR    W,MOVE,SOI,NRY
       /        &NODLE  &NOIEN  &OEY    &NOSRE
       /        &INPT   &NOJMPI &RDGA
       /        &CONT
       ;
 003D           IMME    H#0013
       /        &NODLE  &NOIEN  &OEY    &NOSRE
       /        &BT20   &INPT   &NOJMPI &RDGA
       /        &CONT
       ;
                NOOP
 003E  /        &NODLE  &NOIEN  &NOOEY  &NOSRE
       /        &INPT   &NOJMPI &PF03   &RDGA
       /        &CONT
```

```
      ;
      ;        BEGIN READ DATA SEGMENT LOOP.
      ;        TRANSFER NEXT WORD FROM FIFO ARRAY TO (MAR).
      ;        PASS WHEN FIFO INPUT AGAIN BECOMES AVAILABLE.
      ;
      RDSEC2:  NOOP
 003F /        &NODLE   &NOIEN   &NOOEY   &NOSRE
      /        &BF03    &INPT    &NOJMPI  &MWRT    &RDGA
      /        &IFNOT   RDYI     &CJP     $
      ;
      ;
      ;        INCREMENT (R5) AND TRANSFER THIS TO THE MAR.
      ;
 0040          SOR      W,INC,SORR,R5
      /        &NODLE   &IEN     &OEY     &NOSRE
      /        &INPT    &NOJMPI  &MADR    &RDGA
      /        &CONT
      ;
      ;
      ;        CLOCK LESS SIGNIFICANT BYTE OF ((MAR)) INTO THE AM9520.
      ;        INCREMENT (R4) AND TRANSFER THIS TO THE MAR.
      ;
 0041          SOR      W,INC,SORR,R4
      /        &NODLE   &IEN     &OEY     &NOSRE
      /        &BT2L    &CP20    &INPT    &NOJMPI &MADR    &MREA    &PF03    &RDGA
      /        &CONT
      ;
      ;
      ;        TRANSFER NEXT WORD FROM FIFO ARRAY TO (MAR).
      ;        PASS WHEN FIFO INPUT AGAIN BECOMES AVAILABLE.
      ;
               NOOP
 0042 /        &NODLE   &NOIEN   &NOOEY   &NOSRE
      /        &BF03    &INPT    &NOJMPI  &MWRT    &RDGA
      /        &IFNOT   RDYI     &CJP     $
```

```
        ;
        ;         TRANSFER (R5) TO THE MAR AGAIN.
        ;
 0043             SOR     W,MOVE,SORR,R5
        /         &NODLE  &IEN    &OEY    &NOSRE
        /         &INPT   &NOJMPI &MADR   &RDGA
        /         &CONT
        ;
        ;
        ;         THIS TIME, CLOCK THE MORE SIGNIFICANT BYTE OF ((MAR))
        ;         INTO THE AM9520.
        ;         END READ DATA SEGMENT LOOP.
        ;
 0044             SOR     W,INC,SORR,R4
        /         &NODLE  &IEN    &OEY    &NOSRE
        /         &BT2U   &CP20   &INPT   &NOJMPI &MADR   &MREA   &PF03   &RDGA
        /         &RPCT   RDSEC2
        ;
        ;
        ;         INITIALIZE COUNTER FOR READ HIGH SPEED COMPLETION LOOP.
        ;
                  NOOP
 0045   /         &NODLE  &NOIEN  &NOOEY  &NOSRE
        /         &NOJMPI
        /         &LDCT   RDITCT
```

```
      ;
      ;        BEGIN READ HIGH SPEED COMPLETION LOOP.
      ;        INCREMENT (R5) AND TRANSFER THIS TO THE MAR.
      ;
 0046 RDSEC3:  SOR     W,INC,SORR,R5
      /        &NODLE  &IEN    &OEY    &NOSRE
      /        &NOJMPI &MADR
      /        &CONT
      ;
      ;
      ;        NOOP FOR TIMING PURPOSES
      ;
               NOOP
 0047 /        &NODLE  &NOIEN  &NOOEY  &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
      ;
      ;        CLOCK LESS SIGNIFICANT BYTE OF ((MAR)) INTO THE AM9520.
      ;
               NOOP
 0048 /        &NODLE  &NOIEN  &NOOEY  &NOSRE
      /        &BT2L   &CP20   &NOJMPI &MREA
      /        &CONT
      ;
      ;
      ;        NOOP FOR TIMING PURPOSES
      ;
               NOOP
 0049 /        &NODLE  &NOIEN  &NOOEY  &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
      ;
      ;        NOOP FOR TIMING PURPOSES
      ;
               NOOP
 004A /        &NODLE  &NOIEN  &NOOEY  &NOSRE
      /        &NOJMPI
      /        &CONT
```

```
        ;
        ;       CLOCK MORE SIGNIFICANT BYTE OF ((MAR)) INTO THE AM9520.
        ;       END READ HIGH SPEED COMPLETION LOOP.
        ;
                NOOP
 004B   /       &NODLE   &NOIEN   &NOOEY   &NOSRE
        /       &BT2U    &CP20    &NOJMPI  &MREA
        /       &RPCT    RDSEC3
        ;
        ;
        ;       WAS AN ERROR DETECTED BY THE AM9520?
        ;
                NOOP
 004C   /       &NODLE   &NOIEN   &NOOEY   &NOSRE
        /       &NOJMPI
        /       &IF      ER20     &CJP     RDSEC4
        ;
        ;
        ;       NO; LOAD 0 INTO R0 AND RETURN.
        ;
 004D           SOR      W,MOVE,SOZR,R0
        /       &NODLE   &IEN     &NOOEY   &NOSRE
        /       &NOJMPI
        /       &RTN
        ;
        ;
        ;       YES; IF THE ERROR IS A CORRECTABLE ONE, LOCATE AND CORRECT IT.
        ;       THE ERROR IS LOCATED USING THE CORRECT HIGH SPEED EQUATION:
        ;
        ;               L=NK - (M1A1 + M2A2 + M3A3 + M4A4)
        ;
        ;       WHERE  K,A1,A2,A3,A4 ARE CONSTANTS
        ;       AND  M1,M2,M3,M4 MUST BE CALCULATED.
        ;
        ;       THE ERROR IS CORRECTED BY PERFORMING AN EXOR
        ;       FUNCTION ON THE BURST ERROR IN MEMORY WITH AN
        ;       ERROR PATTERN(EP) PROVIDED BY THE BEP(9520).
        ;
        ;       INITIALIZE CORRECT HIGH SPEED,SET P0=1,& REP=1
 004E RDSEC4: SONR    W,MOVE,SOI,NRY
      /       &NODLE  &NOIEN  &OEY    &NOSRE
      /       &NOJMPI
      /       &CONT
      ;
 004F         IMME    H#001F
      /       &NODLE  &NOIEN  &OEY    &NOSRE
      /       &NOJMPI &BT20   &BF16
      /       &CONT
      ;
      ;
      ;       CLEAR R8(M1).
      ;
 0050         SOR     W,MOVE,SOZR,R8
      /       &NODLE  &IEN    &NOOEY  &NOSRE
      /       &NOJMPI
      /       &CONT
      ;
      ;
      ;       PERIOD FACTOR 1(PF1) TO ACC.
      ;
 0051         SONR    W,MOVE,SOI,NRA
      /       &NODLE  &IEN    &NOOEY  &NOSRE
      /       &NOJMPI
      /       &CONT
      ;
 0052         IMME    PF1
      /       &NODLE  &IEN    &NOOEY  &NOSRE
      /       &NOJMPI
      /       &CONT
      ;
      ;         TEST FOR ERROR PATTERN PRESENT.
      ;
      M1:       NOOP
 0053 /         &NODLE  &NOIEN  &NOOEY  &NOSRE
      /         &NOJMPI
      /         &IF     EP20    &CJP    M234I
      ;
      ;
      ;         EP NOT PRESENT,
      ;         DECREMENT ACC.
      ;         TEST FOR ALIGNMENT EXCEPTION(AE).
      ;
 0054           BONR    W,0,S2NA
      /         &NODLE  &IEN    &NOOEY  &SRE
      /         &NOJMPI &CP20
      /         &IF     AE20    &CJP    AE
      ;
      ;
      ;         AE NOT PRESENT,
      ;         ADD 8 TO R8.
      ;
 0055           BOR2    W,3,A2NR,R8
      /         &NODLE  &IEN    &NOOEY  &NOSRE
      /         &NOJMPI
      /         &JP     LINK
      ;
      ;
      ;         AE PRESENT;
      ;         INC R8.
      ;
 0056 AE:       BOR2    W,0,A2NR,R8
      /         &NODLE  &IEN    &NOOEY  &NOSRE
      /         &NOJMPI
      /         &CONT
```

```
        ;
        ;       TEST FOR PERIOD FACTOR EXCEEDED (UNCORRECTABLE ERROR).
        ;
        LINK:   NOOP
 0057 /         &NODLE  &NOIEN  &NOOEY  &NOSRE  &CT     N
      /         &NOJMPI
      /         &IFNOT  CT16    &CJP    M1
        ;
        ;
        ;       PERIOD FACTOR EXCEEDED (UNCORRECTABLE ERROR);
        ;       SET R0 = 2.
        ;       RETURN.
        ;
 0058 ERR:      BOR2    W,1,LD2NR,R0
      /         &NODLE  &IEN    &OEY    &NOSRE
      /         &NOJMPI &BT20   &BF16
      /         &RTN
        ;
        ;
        ;       EP PRESENT, CALCULATE M2,M3,M4.
        ;
        ;       PERIOD FACTOR 2(PF2) TO R9(M2).
        ;       PERIOD FACTOR 3(PF3) TO R10(M3).
        ;       PERIOD FACTOR 4(PF4) TO R11(M4).
        ;
 0059 M234I:    SOR     W,MOVE,SOI,R9
      /         &NODLE  &IEN    &NOOEY  &NOSRE
      /         &NOJMPI
      /         &CONT
        ;
 005A           IMME    PF2
      /         &NODLE  &IEN    &NOOEY  &NOSRE
      /         &NOJMPI
      /         &CONT
```

```
005B          SOR      W,MOVE,SOI,R10
     /        &NODLE   &IEN     &NOOEY   &NOSRE
     /        &NOJMPI
     /        &CONT
     ;
005C          IMME     PF3
     /        &NODLE   &IEN     &NOOEY   &NOSRE
     /        &NOJMPI
     /        &CONT
     ;
005D          SOR      W,MOVE,SOI,R11
     /        &NODLE   &IEN     &NOOEY   &NOSRE
     /        &NOJMPI
     /        &CONT
     ;
005E          IMME     PF4
     /        &NODLE   &IEN     &NOOEY   &NOSRE
     /        &NOJMPI
     /        &CONT
     ;
     ;
     ;        JUMP TABLE ADDRESS(TAB1) TO R12.
     ;
005F          SOR      W,MOVE,SOI,R12
     /        &NODLE   &IEN     &NOOEY   &NOSRE
     /        &NOJMPI
     /        &CONT
     ;
0060          IMME     TAB1
     /        &NODLE   &IEN     &NOOEY   &NOSRE
     /        &NOJMPI
     /        &CONT
     ;
     ;
     ;        JUMP INDIRECT TO TAB1 VIA PM2-4.
     ;
0061          ROTM     W,15,MDRI,R12
     /        &DLE     &IEN     &NOOEY   &NOSRE
     /        &NOJMPI  &PFPM    &BF2U    &BT16
     /        &CONT
     ;
0062          IMME     H#0007
     /        &NODLE   &IEN     &OEY     &NOSRE
     /        &JMPI    &PFPM    &BF2U    &BT16
     /        &JP      $
```

```
      ;
      ;        R9 = PF2 - R9.
      ;
 0063 MFIX:    TOR1     W,TORIR,SUBR,R9
      /        &NODLE  &IEN    &NOOEY  &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
 0064          IMME     PF2
      /        &NODLE  &IEN    &NOOEY  &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
      ;
      ;        R10 = PF3 - R10.
      ;
 0065          TOR1     W,TORIR,SUBR,R10
      /        &NODLE  &IEN    &NOOEY  &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
 0066          IMME     PF3
      /        &NODLE  &IEN    &NOOEY  &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
      ;
      ;        R11 = PF4 - R11.
      ;
 0067          TOR1     W,TORIR,SUBR,R11
      /        &NODLE  &IEN    &NOOEY  &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
 0068          IMME     PF4
      /        &NODLE  &IEN    &NOOEY  &NOSRE
      /        &NOJMPI
      /        &CONT
```

```
       ;
       ;        0 TO R7 (LOCATION ACC MSW,LAC).
       ;
 0069 M1A1:     SOR      W,MOVE,SOZR,R7
       /        &NODLE   &IEN     &NOOEY   &NOSRE
       /        &NOJMPI
       /        &CONT
       ;
       ;
       ;        A1 TO R12(LSW),R13(MSW).
       ;
 006A           SOR      W,MOVE,SOI,R12
       /        &NODLE   &IEN     &NOOEY   &NOSRE
       /        &NOJMPI
       /        &CONT
       ;
 006B           IMME     A1LSW
       /        &NODLE   &IEN     &NOOEY   &NOSRE
       /        &NOJMPI
       /        &CONT
       ;
 006C           SOR      W,MOVE,SOI,R13
       /        &NODLE   &IEN     &NOOEY   &NOSRE
       /        &NOJMPI
       /        &CONT
       ;
 006D           IMME     A1MSW
       /        &NODLE   &IEN     &NOOEY   &NOSRE
       /        &NOJMPI
       /        &LDCT    4
       ;
       ;
       ;        R8 TO D.
       ;        LAC = M1A1, MULTIPLY M1A1.
       ;
 006E           SOR      W,MOVE,SORY,R8
       /        &DLE     &NOIEN   &OEY     &NOSRE
       /        &NOJMPI
       /        &JS      MUL
```

```
      ;
      ;         A2 TO R12,R13.
      ;
 006F M2A2:     IMME     A2LSW
      /         &NODLE   &IEN     &NOOEY   &NOSRE
      /         &NOJMPI
      /         &CONT
      ;
 0070           BOR2     W,3,LD2NR,R13
      /         &NODLE   &IEN     &NOOEY   &NOSRE
      /         &NOJMPI
      /         &LDCT    3
      ;
      ;
      ;         R9 TO D.
      ;         LAC = LAC + M2A2.
      ;
 0071           SOR      W,MOVE,SORY,R9
      /         &DLE     &NOIEN   &OEY     &NOSRE
      /         &NOJMPI
      /         &JS      MUL
      ;
      ;
      ;         A3'(A3 - 4K) TO R12,R13.
      ;
 0072 M3A3:     IMME     A3LSW
      /         &NODLE   &IEN     &NOOEY   &NOSRE
      /         &NOJMPI
      /         &CONT
      ;
 0073           BOR2     W,1,LD2NR,R13
      /         &NODLE   &IEN     &NOOEY   &NOSRE
      /         &NOJMPI
      /         &LDCT    6
      ;
      ;
      ;         R10 TO D.
      ;         LAC = LAC + M3A3.
      ;
 0074           SOR      W,MOVE,SORY,R10
      /         &DLE     &NOIEN   &OEY     &NOSRE
      /         &NOJMPI
      /         &JS      MUL
```

```
       ;
       ;        A4'(A4 - 4K) TO R12,R13.
       ;
 0075 M4A4:     IMME     A4LSW
       /        &NODLE   &IEN     &NOOEY   &NOSRE
       /        &NOJMPI
       /        &CONT
       ;
 0076           BOR2     W,3,LD2NR,R13
       /        &NODLE   &IEN     &NOOEY   &NOSRE
       /        &NOJMPI
       /        &LDCT    4
       ;
       ;
       ;        R11 TO D.
       ;        LAC = LAC + M4A4.
       ;
 0077           SOR      W,MOVE,SORY,R11
       /        &DLE     &NOIEN   &OEY     &NOSRE
       /        &NOJMPI
       /        &JS      MUL
       ;
       ;
       ;        PRESHIFTED DIVISOR(K) TO R12,R13,D.
       ;        LAC = REM(M1A1 + M2A2 + M3A3 + M4A4) / K.
       ;        LAC = -L + K.
       ;
 0078           IMME     KL128
       /        &NODLE   &IEN     &NOOEY   &NOSRE
       /        &NOJMPI
       /        &CONT
       ;
 0079           SOR      W,MOVE,SOI,R13
       /        &DLE     &IEN     &OEY     &NOSRE
       /        &NOJMPI
       /        &LDCT    6
       ;
 007A           IMME     KM128
       /        &DLE     &IEN     &OEY     &NOSRE
       /        &NOJMPI
       /        &JS      DIV
```

```
     ;
     ;       L = LAC = K - LAC.
     ;
007B SUBK:   IMME    KLSW
     /       &NODLE  &IEN    &NOOEY  &NOSRE
     /       &NOJMPI
     /       &CONT
     ;
007C         TOR1    W,TORIR,SUBRC,R7
     /       &NODLE  &IEN    &NOOEY  &NOSRE
     /       &NOJMPI
     /       &CONT
     ;
007D         IMME    KMSW
     /       &NODLE  &IEN    &NOOEY  &NOSRE
     /       &NOJMPI
     /       &CONT
     ;
     ;
     ;       0 TO R8.
     ;
007E XORMEM: SOR     W,MOVE,SOZR,R8
     /       &NODLE  &IEN    &NOOEY  &NOSRE
     /       &NOJMPI
     /       &CONT
     ;
     ;
     ;       0 TO R9.
     ;
007F         SOR     W,MOVE,SOZR,R9
     /       &NODLE  &IEN    &NOOEY  &NOSRE
     /       &NOJMPI
     /       &CONT
     ;
     ;
     ;       0 TO ACC.
     ;
0080         SOR     W,MOVE,SORA,R9
     /       &NODLE  &IEN    &NOOEY  &NOSRE
     /       &NOJMPI
     /       &CONT
```

```
       ;
       ;        ROTATE R6 DOWN BY FOUR TO OBTAIN WORD ADDRESS
       ;        AND STORE IN ACC.
       ;
 0081           ROTM    W,12,MRAI,R6
       /        &NODLE  &IEN    &NOOEY  &NOSRE
       /        &NOJMPI
       /        &CONT
       ;
 0082           IMME    H#0FFF
       /        &NODLE  &IEN    &NOOEY  &NOSRE
       /        &NOJMPI
       /        &CONT
       ;
       ;
       ;        MASK UPPER 12 BITS OF R6 TO OBTAIN FIRST BIT OF
       ;        BURST ERROR AND STORE IN R6.
       ;
 0083           TOR1    W,TORIR,AND,R6
       /        &NODLE  &IEN    &NOOEY  &NOSRE
       /        &NOJMPI
       /        &CONT
       ;
 0084           IMME    H#000F
       /        &NODLE  &IEN    &NOOEY  &NOSRE
       /        &NOJMPI
       /        &CONT
       ;
       ;
       ;        JUMP INDIRECT TO TAB2 VIA R6.
       ;
 0085           TOR1    W,TORIR,ADD,R6
       /        &NODLE  &IEN    &NOOEY  &NOSRE
       /        &NOJMPI
       /        &CONT
       ;
 0086           IMME    TAB2
       /        &NODLE  &IEN    &OEY    &NOSRE
       /        &JMPI
       /        &CONT
```

```
     ;
     ;         MADR = ACC = R4 - ACC.
     ;
0087 XOR:      TOR1     W,TORAA,SUBS,R4
     /         &NODLE   &IEN     &NOOEY   &NOSRE
     /         &NOJMPI
     /         &CONT
     ;
0088           BONR     W,0,S2NA
     /         &NODLE   &IEN     &OEY     &NOSRE
     /         &MADR    &NOJMPI
     /         &CONT
     ;
     ;
     ;         R8 = R8 XOR MEM.
     ;
0089           TOR1     W,TODRR,EXOR,R8
     /         &DLE     &IEN     &NOOEY   &NOSRE
     /         &NOJMPI  &BT16
     /         &CONT
     ;
     ;
     ;         R8 TO MEMORY.
     ;
008A           SOR      W,MOVE,SORY,R8
     /         &NODLE   &NOIEN   &OEY     &NOSRE
     /         &NOJMPI  &MWRT    &BF16
     /         &CONT
     ;
     ;
     ;         MADR = ACC + 1.
     ;
008B           SONR     W,INC,SOA,NRY
     /         &NODLE   &NOIEN   &OEY     &NOSRE
     /         &NOJMPI  &MADR
     /         &CONT
     ;
     ;
     ;         R9 = R9 XOR MEM.
     ;
008C           TOR1     W,TODRR,EXOR,R9
     /         &DLE     &IEN     &NOOEY   &NOSRE
     /         &NOJMPI  &BT16
     /         &CONT
```

```
         ;
         ;        R9 TO MEMORY.
         ;
 008D             SOR     W,MOVE,SORY,R9
         /        &NODLE  &NOIEN  &OEY    &NOSRE
         /        &NOJMPI &MWRT   &BF16
         /        &CONT
         ;
         ;
         ;        0 TO R0 (ERROR CORRECTED FLAG).
         ;        RETURN.
         ;
 008E             SOR     W,MOVE,SOZR,R0
         /        &NODLE  &IEN    &OEY    &NOSRE
         /        &NOJMPI &BT20   &BF16
         /        &RTN
```

```
      ;
      ;         TABLE 1 IS USED TO CALCULATE
      ;         M2,M3,M4 AND DETECT UNCORRECTABLE ERRORS.
      ;         EACH MICROINSTRUCTION PATH DECREMENTS THE
      ;         APPROPRIATE REGISTER(S) AND CHECKS FOR VALUES
      ;         EXCEEDING THE 56-BIT POLYNOMIAL PERIOD
      ;         FACTOR LIMITS.
      ;
                ALIGN   8
 0090 ;
 0090 TAB1:     BOR2    W,0,S2NR,R9
      /         &NODLE  &IEN    &NOOEY  &SRE
      /         &NOJMPI &CP20   &PFPM
      /         &JP     PM234
      ;
 0091           BOR2    W,0,S2NR,R11
      /         &NODLE  &IEN    &NOOEY  &SRE
      /         &NOJMPI &CP20   &PFPM
      /         &JP     TM34
      ;
 0092           BOR2    W,0,S2NR,R9
      /         &NODLE  &IEN    &NOOEY  &SRE
      /         &NOJMPI &CP20   &PFPM
      /         &JP     TM24
      ;
 0093           BOR2    W,0,S2NR,R11
      /         &NODLE  &IEN    &NOOEY  &SRE
      /         &NOJMPI &CP20   &PFPM
      /         &JP     TM1
      ;
 0094           BOR2    W,0,S2NR,R10
      /         &NODLE  &IEN    &NOOEY  &SRE
      /         &NOJMPI &CP20   &PFPM
      /         &JP     TM23
      ;
 0095           BOR2    W,0,S2NR,R10
      /         &NODLE  &IEN    &NOOEY  &SRE
      /         &NOJMPI &CP20   &PFPM
      /         &JP     TM1
      ;
 0096           BOR2    W,0,S2NR,R9
      /         &NODLE  &IEN    &NOOEY  &SRE
      /         &NOJMPI &CP20   &PFPM
      /         &JP     TM1
      ;
 0097           SOR     W,MOVE,SOZR,R6
      /         &NODLE  &IEN    &NOOEY  &SRE
      /         &NOJMPI &PFPM
      /         &JP     MFIX
      TM34:    NOOP
0098  /        &NODLE   &NOIEN    &NOOEY    &NOSRE
      /        &NOJMPI  &CP20     &PFPM
      /        &JP      PM34
      ;
      TM24:    NOOP
0099  /        &NODLE   &NOIEN    &NOOEY    &NOSRE
      /        &NOJMPI  &CP20     &PFPM
      /        &JP      PM24
      ;
      TM23:    NOOP
009A  /        &NODLE   &NOIEN    &NOOEY    &NOSRE
      /        &NOJMPI  &CP20     &PFPM
      /        &JP      PM23
      ;
      TM1:     NOOP
009B  /        &NODLE   &NOIEN    &NOOEY    &NOSRE
      /        &NOJMPI  &CP20     &PFPM
      /        &CONT
      ;
               NOOP
009C  /        &NODLE   &NOIEN    &NOOEY    &NOSRE
      /        &NOJMPI  &CP20     &PFPM
      /        &JP      PM1
      ;
009D  PM234:   BOR2     W,0,S2NR,R11
      /        &NODLE   &IEN      &NOOEY    &SRE      &CT       N
      /        &NOJMPI  &CP20     &PFPM
      /        &IF      CT16      &CJP      ERR
      ;
009E  PM34:    BOR2     W,0,S2NR,R10
      /        &NODLE   &IEN      &NOOEY    &SRE      &CT       N
      /        &NOJMPI  &CP20     &PFPM
      /        &IF      CT16      &CJP      ERR
      ;
009F  PM1:     ROTM     W,15,MDRI,R12
      /        &DLE     &IEN      &NOOEY    &NOSRE    &CT       N
      /        &NOJMPI  &PFPM     &BT16     &BF2U
      /        &IF      CT16      &CJP      LINK
      ;
00A0           IMME     H#0007
      /        &NODLE   &IEN      &OEY      &NOSRE
      /        &JMPI    &PFPM     &BT16     &BF2U
      /        &JP      $
```

```
 00A1 PM24:    BOR2     W,0,S2NR,R11
      /        &NODLE   &IEN     &NOOEY   &SRE     &CT      N
      /        &NOJMPI  &CP20    &PFPM
      /        &IFNOT   CT16     &CJP     PM1
      ;
               NOOP
 00A2 /        &NODLE   &NOIEN   &NOOEY   &NOSRE
      /        &NOJMPI
      /        &JP      ERR
      ;
 00A3 PM23:    BOR2     W,0,S2NR,R9
      /        &NODLE   &IEN     &NOOEY   &SRE     &CT      N
      /        &NOJMPI  &CP20    &PFPM
      /        &IFNOT   CT16     &CJP     PM1
      ;
               NOOP
 00A4 /        &NODLE   &NOIEN   &NOOEY   &NOSRE
      /        &NOJMPI
      /        &JP      ERR
      ;
      ;
      ;        TABLE 2 IS USED FOR ROTATING MEMORY WORD(S)
      ;        VIA ROTATE AND MERGE INSTRUCTION(S) SO THAT
      ;        BURST ERRORS CAN BE ALIGNED WITH THE ERROR
      ;        PATTERN PROVIDED BY THE BEP(9520).
      ;
 00A5 TAB2:    ROTM     W,9,MDRI,R8
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      REP1
      ;
 00A6          ROTM     W,10,MDRI,R8
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      REP2
      ;
 00A7          ROTM     W,11,MDRI,R8
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      REP3
      ;
 00A8          ROTM     W,12,MDRI,R8
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      REP4
 00A9          ROTM     W,13,MDRI,R8
       /       &DLE     &IEN     &NOOEY   &NOSRE
       /       &NOJMPI  &BT16    &BF2U    &BF2L
       /       &JP      REP5
       ;
 00AA          ROTM     W,14,MDRI,R8
       /       &DLE     &IEN     &NOOEY   &NOSRE
       /       &NOJMPI  &BT16    &BF2U    &BF2L
       /       &JP      REP6
       ;
 00AB          ROTM     W,15,MDRI,R8
       /       &DLE     &IEN     &NOOEY   &NOSRE
       /       &NOJMPI  &BT16    &BF2U    &BF2L
       /       &JP      REP7
       ;
 00AC          ROTM     W,0,MDRI,R8
       /       &DLE     &IEN     &NOOEY   &NOSRE
       /       &NOJMPI  &BT16    &BF2U    &BF2L
       /       &JP      REP8
       ;
 00AD          ROTM     W,1,MDRI,R8
       /       &DLE     &IEN     &NOOEY   &NOSRE
       /       &NOJMPI  &BT16    &BF2U    &BF2L
       /       &JP      REP9
       ;
 00AE          ROTM     W,2,MDRI,R8
       /       &DLE     &IEN     &NOOEY   &NOSRE
       /       &NOJMPI  &BT16    &BF2U    &BF2L
       /       &JP      REP10
       ;
 00AF          ROTM     W,3,MDRI,R8
       /       &DLE     &IEN     &NOOEY   &NOSRE
       /       &NOJMPI  &BT16    &BF2U    &BF2L
       /       &JP      REP11
       ;
 00B0          ROTM     W,4,MDRI,R8
       /       &DLE     &IEN     &NOOEY   &NOSRE
       /       &NOJMPI  &BT16    &BF2U    &BF2L
       /       &JP      REP12
       ;
 00B1          ROTM     W,5,MDRI,R8
       /       &DLE     &IEN     &NOOEY   &NOSRE
       /       &NOJMPI  &BT16    &BF2U    &BF2L
       /       &JP      REP13
 00B2          ROTM     W,6,MDRI,R8
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      REP14
      ;
 00B3          ROTM     W,7,MDRI,R8
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      REP15
      ;
 00B4          ROTM     W,8,MDRI,R8
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      REP16
 00B5 REP1:    IMME     H#0001
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      RM1
      ;
 00B6 REP2:    IMME     H#0003
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      RM2
      ;
 00B7 REP3:    IMME     H#0007
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      RM3
      ;
 00B8 REP4:    IMME     H#000F
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      RM4
      ;
 00B9 REP5:    IMME     H#001F
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      RM5
      ;
 00BA REP6:    IMME     H#003F
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      RM6
 00BB REP7:    IMME     H#007F
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      RM7
      ;
 00BC REP8:    IMME     H#00FF
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      RM8
      ;
 00BD REP9:    IMME     H#01FF
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      RM9
      ;
 00BE REP10:   IMME     H#03FF
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      RM10
      ;
 00BF REP11:   IMME     H#07FF
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      RM11
      ;
 00C0 REP12:   IMME     H#0FFF
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      XOR
      ;
 00C1 REP13:   IMME     H#1FFE
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      XOR
      ;
 00C2 REP14:   IMME     H#3FFC
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      XOR
      ;
 00C3 REP15:   IMME     H#7FF8
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      XOR
 00C4 REP16:    IMME      H#FFF0
      /         &DLE      &IEN      &NOOEY    &NOSRE
      /         &NOJMPI   &BT16     &BF2U     &BF2L
      /         &JP       XOR
      ;
 00C5 RM1:      ROTM      W,9,MDRI,R9
      /         &DLE      &IEN      &NOOEY    &NOSRE
      /         &NOJMPI   &BT16     &BF2U     &BF2L
      /         &JP       REP17
      ;
 00C6 RM2:      ROTM      W,10,MDRI,R9
      /         &DLE      &IEN      &NOOEY    &NOSRE
      /         &NOJMPI   &BT16     &BF2U     &BF2L
      /         &JP       REP18
      ;
 00C7 RM3:      ROTM      W,11,MDRI,R9
      /         &DLE      &IEN      &NOOEY    &NOSRE
      /         &NOJMPI   &BT16     &BF2U     &BF2L
      /         &JP       REP19
      ;
 00C8 RM4:      ROTM      W,12,MDRI,R9
      /         &DLE      &IEN      &NOOEY    &NOSRE
      /         &NOJMPI   &BT16     &BF2U     &BF2L
      /         &JP       REP20
      ;
 00C9 RM5:      ROTM      W,13,MDRI,R9
      /         &DLE      &IEN      &NOOEY    &NOSRE
      /         &NOJMPI   &BT16     &BF2U     &BF2L
      /         &JP       REP21
      ;
 00CA RM6:      ROTM      W,14,MDRI,R9
      /         &DLE      &IEN      &NOOEY    &NOSRE
      /         &NOJMPI   &BT16     &BF2U     &BF2L
      /         &JP       REP22
      ;
 00CB RM7:      ROTM      W,15,MDRI,R9
      /         &DLE      &IEN      &NOOEY    &NOSRE
      /         &NOJMPI   &BT16     &BF2U     &BF2L
      /         &JP       REP23
      ;
 00CC RM8:      ROTM      W,0,MDRI,R9
      /         &DLE      &IEN      &NOOEY    &NOSRE
      /         &NOJMPI   &BT16     &BF2U     &BF2L
      /         &JP       REP24
 00CD RM9:     ROTM     W,1,MDRI,R9
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      REP25
      ;
 00CE RM10:    ROTM     W,2,MDRI,R9
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      REP26
      ;
 00CF RM11:    ROTM     W,3,MDRI,R9
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      REP27
      ;
 00D0 REP17:   IMME     H#FFE0
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      XOR
      ;
 00D1 REP18:   IMME     H#FFC0
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      XOR
      ;
 00D2 REP19:   IMME     H#FF80
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      XOR
      ;
 00D3 REP20:   IMME     H#FF00
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      XOR
      ;
 00D4 REP21:   IMME     H#FE00
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      XOR
      ;
 00D5 REP22:   IMME     H#FC00
      /        &DLE     &IEN     &NOOEY   &NOSRE
      /        &NOJMPI  &BT16    &BF2U    &BF2L
      /        &JP      XOR
 00D6 REP23:  IMME     H#F800
       /      &DLE     &IEN     &NOOEY   &NOSRE
       /      &NOJMPI  &BT16    &BF2U    &BF2L
       /      &JP      XOR
       ;
 00D7 REP24:  IMME     H#F000
       /      &DLE     &IEN     &NOOEY   &NOSRE
       /      &NOJMPI  &BT16    &BF2U    &BF2L
       /      &JP      XOR
       ;
 00D8 REP25:  IMME     H#E000
       /      &DLE     &IEN     &NOOEY   &NOSRE
       /      &NOJMPI  &BT16    &BF2U    &BF2L
       /      &JP      XOR
       ;
 00D9 REP26:  IMME     H#C000
       /      &DLE     &IEN     &NOOEY   &NOSRE
       /      &NOJMPI  &BT16    &BF2U    &BF2L
       /      &JP      XOR
       ;
 00DA REP27:  IMME     H#8000
       /      &DLE     &IEN     &NOOEY   &NOSRE
       /      &NOJMPI  &BT16    &BF2U    &BF2L
       /      &JP      XOR
```

```
        ;
        ;       SUBROUTINE MULTIPLY
        ;
        ;       THIS SUBROUTINE MULTIPLIES A DOUBLE PRECISION
        ;       WORD BY A SINGLE PRECISION WORD AND ASSUMES
        ;       A DOUBLE PRECISION ANSWER.  THE MULTIPLIER
        ;       IS IN D, THE MULTIPLICAND IS IN R12,R13, AND
        ;       THE ANSWER APPEARS IN R6,R7(LAC). NOTE, UPON
        ;       RETURNING TO THE MAIN PROGRAM THE SUBROUTINE
        ;       INITIATES AN IMMEDIATE MOVE TO R12.
        ;
 00DB MUL:      SHFTR   W,SHDR,SHDNZ,R8
        /       &NODLE  &IEN    &NOOEY  &SRE
        /       &NOJMPI
        /       &CONT
        ;
 00DC           SOR     W,MOVE,SORY,R13
        /       &DLE    &NOIEN  &OEY    &NOSRE
        /       &NOJMPI
        /       &CONT
        ;
 00DD CYC:      SOR     W,MOVE,SORA,R12
        /       &NODLE  &IEN    &NOOEY  &NOSRE  &CT     L
        /       &NOJMPI
        /       &IFNOT  CT16    &CJP    NOTQ
        ;
 00DE           TOR1    W,TORAR,ADD,R6
        /       &NODLE  &IEN    &NOOEY  &SRE
        /       &NOJMPI
        /       &CONT
        ;
 00DF           TOR1    W,TODRR,ADDC,R7
        /       &NODLE  &IEN    &NOOEY  &NOSRE
        /       &NOJMPI
        /       &CONT
 00E0 NOTQ:    SHFTR   W,SHRR,SHUPZ,R12
      /        &NODLE  &IEN    &NOOEY  &SRE
      /        &NOJMPI
      /        &CONT
      ;
 00E1          SHFTR   W,SHRR,SHUPL,R13
      /        &DLE    &IEN    &OEY    &NOSRE
      /        &NOJMPI
      /        &CONT
      ;
 00E2          SHFTR   W,SHRR,SHDNZ,R8
      /        &NODLE  &IEN    &NOOEY  &SRE
      /        &NOJMPI
      /        &RPCT   CYC
      ;
 00E3          SOR     W,MOVE,SOI,R12
      /        &NODLE  &IEN    &NOOEY  &NOSRE
      /        &NOJMPI
      /        &RTN
      ;
      ;         SUBROUTINE DIVIDE
      ;         THIS SUBROUTINE DIVIDES A DOUBLE PRECISION
      ;         NUMBER BY A DOUBLE PRECISION NUMBER LEAVING
      ;         ONLY A REMAINDER.  THE DIVISOR IS IN R12,R13,D
      ;         AND THE DIVIDEND/REMAINDER APPEARS IN R6,R7.
      ;         NOTE, UPON RETURN AN IMMEDIATE SUBTRACT IS
      ;         INITIATED.
      ;
 00E4 DIV:      SOR     W,MOVE,SORA,R12
      /         &NODLE  &IEN    &NOOEY  &NOSRE
      /         &NOJMPI
      /         &CONT
      ;
 00E5 CYC1:     TOR1    W,TORAR,SUBS,R6
      /         &NODLE  &IEN    &NOOEY  &SRE
      /         &NOJMPI
      /         &CONT
      ;
 00E6           TOR1    W,TODRR,SUBSC,R7
      /         &NODLE  &IEN    &NOOEY  &SRE
      /         &NOJMPI
      /         &CONT
      ;
                NOOP
 00E7 /         &NODLE  &IEN    &NOOEY  &NOSRE  &CT     N
      /         &NOJMPI
      /         &IFNOT  CT16    &CJP    POS
      ;
 00E8           TOR1    W,TORAR,ADD,R6
      /         &NODLE  &IEN    &NOOEY  &SRE
      /         &NOJMPI
      /         &CONT
      ;
 00E9           TOR1    W,TODRR,ADDC,R7
      /         &NODLE  &IEN    &NOOEY  &NOSRE
      /         &NOJMPI
      /         &CONT
 00EA POS:     SHFTR   W,SHRR,SHDNZ,R13
      /        &DLE    &IEN     &OEY     &SRE
      /        &NOJMPI
      /        &CONT
      ;
 00EB          SHFTR   W,SHA,SHDNL,R12
      /        &NODLE  &IEN     &NOOEY   &NOSRE
      /        &NOJMPI
      /        &RPCT   CYC1
      ;
 00EC          TOR1    W,TORIR,SUBR,R6
      /        &NODLE  &IEN     &NOOEY   &SRE
      /        &NOJMPI
      /        &RTN
      ;
      ;        WRITE SECTOR
      ;
      ;        (R3) TO R4 AND MAR.
      ;        ENTER OUTPUT MODE. (TURNS ON WRITE CLOCK.)
      ;        INITIALIZE COUNTER FOR OUTPUT DATA PREAMBLE LOOP.
      ;
 00ED WRSEC1:  SOR     W,NEG,SOAR,R4
      /        &NODLE  &IEN    &NOOEY  &NOSRE
      /        &MADR   &NOJMPI &OUPT
      /        &LDCT   5
      ;
      ;
      ;        BEGIN WRITE DATA PREAMBLE LOOP.
      ;        TURN ON WRITE GATE.
      ;        PASS WHEN FIFO ARRAY READY FOR OUTPUT.
      ;
      WRSEC2:  NOOP
 00EE /        &NODLE  &NOIEN  &NOOEY  &NOSRE
      /        &NOJMPI &OUPT   &WRGA
      /        &IFNOT  RDYO    &CJP    $
      ;
      ;
      ;        OUTPUT H#0000 TO FIFO ARRAY.
      ;
 00EF          SONR    W,MOVE,SOZ,NRY
      /        &NODLE  &NOIEN  &OEY    &NOSRE
      /        &NOJMPI &OUPT   &WRGA
      /        &CONT
      ;
      ;
      ;        END WRITE DATA PREAMBLE LOOP.
      ;
 00F0          SONR    W,MOVE,SOZ,NRY
      /        &NODLE  &NOIEN  &OEY    &NOSRE
      /        &BF16   &BT03   &NOJMPI &OUPT   &PL03   &WRGA
      /        &RPCT   WRSEC2
      ;
      ;
      ;        OUTPUT LAST DATA PREAMBLE BYTE AND THE H#FE DATA SYNC BYTE.
      ;        INITIALIZE COUNTER FOR WRITE DATA SEGMENT LOOP.
      ;        PASS WHEN FIFO ARRAY AGAIN READY FOR OUTPUT.
      ;
 00F1 WRSEC3:  SONR    W,MOVE,SOI,NRY
      /        &DLE    &NOIEN  &OEY    &NOSRE
      /        &NOJMPI &OUPT   &WRGA
      /        &LDCT   131
      ;
 00F2          IMME    H#FE00
      /        &DLE    &NOIEN  &OEY    &NOSRE
      /        &NOJMPI &OUPT   &WRGA
      /        &IFNOT  RDYO    &CJP    WRSEC3
      ;
 00F3          SONR    W,MOVE,SOD,NRY
      /        &NODLE  &NOIEN  &OEY    &NOSRE
      /        &BF16   &BT03   &NOJMPI &OUPT   &PL03   &WRGA
      /        &CONT
      ;
      ;       BEGIN WRITE DATA SEGMENT LOOP.
      ;       (R4) TO MAR.
      ;
 00F4 WRSEC4: SOR     W,MOVE,SORY,R4
      /       &NODLE  &NOIEN  &OEY    &NOSRE
      /       &MADR   &NOJMPI &OUPT   &WRGA
      /       &CONT
      ;
      ;
      ;       INCREMENT (R4).
      ;
 00F5         SOR     W,INC,SORR,R4
      /       &NODLE  &IEN    &NOOEY  &NOSRE
      /       &NOJMPI &OUPT   &WRGA
      /       &CONT
      ;
      ;
      ;       ((MAR)) TO FIFO ARRAY.
      ;       END WRITE DATA SEGMENT LOOP.
      ;
              NOOP
 00F6 /       &NODLE  &NOIEN  &NOOEY  &NOSRE
      /       &BT03   &MREA   &NOJMPI &OUPT   &PL03   &WRGA
      /       &RPCT   WRSEC4
      ;
      ;
      ;       CLEAR (R4).
      ;       INITIALIZE COUNTER FOR WRITE DATA POSTAMBLE LOOP.
      ;
 00F7         SOR     W,MOVE,SOZR,R4
      /       &NODLE  &IEN    &NOOEY  &NOSRE
      /       &NOJMPI &OUPT   &WRGA
      /       &LDCT   1
```

```
        ;
        ;       BEGIN WRITE DATA POSTAMBLE LOOP.
        ;       OUTPUT H#0000 TO FIFO ARRAY.
        ;
 00F8 WRSEC5:   SOR     W,MOVE,SORY,R4
        /       &NODLE  &NOIEN  &OEY    &NOSRE
        /       &NOJMPI &OUPT   &WRGA
        /       &IFNOT  RDYO    &CJP    $
        ;
 00F9           SOR     W,MOVE,SORY,R4
        /       &NODLE  &NOIEN  &OEY    &NOSRE
        /       &BF16   &BT03   &NOJMPI &OUPT   &PL03   &WRGA
        /       &CONT
        ;
        ;
        ;       (NOOP FOR TIMING PURPOSES)
        ;       END WRITE DATA POSTAMBLE LOOP.
        ;
                NOOP
 00FA   /       &NODLE  &NOIEN  &NOOEY  &NOSRE
        /       &NOJMPI &OUPT   &WRGA
        /       &RPCT   WRSEC5
        ;
        ;
        ;       TURN OFF WRITE GATE.
        ;
                NOOP
 00FB   /       &NODLE  &NOIEN  &NOOEY  &NOSRE
        /       &NOJMPI &OUPT
        /       &CONT
        ;
        ;
        ;       THEN LEAVE OUTPUT MODE. (TURNS OFF WRITE CLOCK.)
        ;       LOAD 0 INTO R0.
        ;       RETURN.
        ;
 00FC           SOR     W,MOVE,SOZR,R0
        /       &NODLE  &IEN    &NOOEY  &NOSRE
        /       &NOJMPI
        /       &RTN
        ;
                END
```

```
0000 1100000111100000 0010011000110000 011010010011XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0001 1111100011100000 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0002 0000000000000000 XXXX001011101111 111111XXXXXXXXXX XXXXXXXX0XXXXX10 XXXXXXXXXXXXXXXX
0003 1111100011100000 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0004 0000000000010000 XXXX001011000001 111111XXXXXXXXXX XXXXXXXX0XXXXX10 XXXXXXXXXXXXXXXX
0005 1101100001000011 XXXX001111101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0006 1101100011000100 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0007 1101100001000100 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 0XXXXXXXXXXXXXXX
0008 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXX0XXXXXXX10 XXXXXXXXXXXXXXXX
0009 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXXXX0XX10 XXXXXXXXXXXXXXXX
000A 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
000B 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXX0XXXXXX10 XXXXXXXXXXXXXXXX
000C 1101110101100100 XXXX010010010000 000111XXXXXXXXXX XXXXXXXXXXX0XX10 XXXXXXXXXXXXXXXX
000D 1111100011100000 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
000E 0000000000010001 XXXX001011000000 000010XXXXXXXXXX XXXXXXXX0XXXXX10 XXXXXXXXXXXXXXXX
000F 1101100001000100 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXX0XX10 0XXXXXXXXXXXXXXX
0010 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXX0XXXX10 XXXXXXXXXXXXXXXX
0011 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0012 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXXXX0XX10 XXXXXXXXXXXXXXXX
0013 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0014 1101110101100100 XXXX010010010000 001111XXXXXXXXXX X00XXXXXXXXXXX10 XX0XXXXXXXXXXXXX
0015 1101100100000101 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXX0XX10 XX0XXXXXXXXXXXXX
0016 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXX0XXXX10 XXXXXXXXXXXXXXXX
0017 0101100011000101 XXXX010111101111 111111XXXXXXXXXX X0XXX0XXXXXXXX10 XXXXXXXXXXXXXXXX
0018 1101100001000100 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 0XXXXXXXXXXXXXXX
0019 1101100001000101 XXXX001011101111 111111XXXXXXXXXX 0XXXXXXXXXXXXX10 XX0XXXXXXXXXXXXX
001A 1111100011100001 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
001B 0000000000110110 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
001C 1000100000000000 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
001D 1111100011100001 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
001E 1100000000000001 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
001F 1101100100000100 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
```

```
0020 1101100001000001 XXXX010111000000 010000XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0021 1101100011000101 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0022 1100000011000101 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0023 1100110001100100 XXXX010010010000 100010XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0024 1101100001000010 XXXX010111000000 010000XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0025 1101100011000101 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0026 1100000011000101 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0027 1100110001100100 XXXX010010010000 100110XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0028 0111000101000000 XXXX011011000001 000000XXXXXXXXXX XXXXXXXXXXXXX010 XXXXXXXXX0XXXXXX
0029 0111000101000000 XXXX011011101111 111111XXXXXX0XXX XXXXXXXXXXXXX010 XXXXXXXXX00XXXXX
002A 0111000101000000 XXXX011000110000 1010101010010XXX XXXXXXXXXXXXX010 XXXXXXXXX0XXXXXX
002B 0111000101000000 XXXX011000110000 101011011011XXXX XXXXXXXXXXXXX010 XXXXXXXXX0XXXXXX
002C 1101100011000101 XXXX010100110000 101100011011XXX0 XXXXX0XXXXXXX010 XXXXXXXXX0XXXXXX
002D 1001000111100001 0010111000110000 110011010010XXXX XXXXXXXXXXXXX010 XXXXXXXXX0XXXXXX
002E 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXXXXXX010 XXXXXX0XX0XXXXXX
002F 1101100011000101 XXXX010100110000 101111011011XXX0 XXXXX0XXXXXXX010 XXXXXXXXX0XXXXXX
0030 1001000111100010 0010111000110000 110011010010XXXX XXXXXXXXXXXXX010 XXXXXX0XX0XXXXXX
0031 1101100011000101 XXXX010111101111 111111XXXXXXXXX0 XXXXX0XXXXXXX010 XXXXXXXXX0XXXXXX
0032 1001000111100100 0010111000110000 110101010010XXXX XXXXXXXXXXXXX010 XXXXXXXXX0XXXXXX
0033 1101100001000000 XXXX011010010000 101001XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0034 1101100000000000 XXXX010010101111 111111011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0035 1101100001000000 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX01 XXXXXXXXXXXXXXXX
0036 1101111000000011 XXXX010000110000 111000011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXX0XXXXX
0037 1101111000000011 XXXX010000110011 101101011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXX0XXXXX
0038 1101111010000100 XXXX000011000001 000001XXXXXXXXXX XXXXXXXXXXXXX010 0XXXXXXXX00XXXXX
0039 1101101010000101 XXXX010000110000 111001011011XXXX XXXXXXXXXXXXX010 XXXXXXXXX0XXXXXX
```

```
003A 1111100011100000 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXXXX010 XXXXXXXXX0XXXXXX
003B 0000000000000011 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXXXX010 XXXXXXXXX0XXXXXX
003C 1111100011100000 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXXXX010 XXXXXXXXX0XXXXXX
003D 0000000000010011 XXXX001011101111 111111XXXXXXXXXX XXXXXXXX0XXXX010 XXXXXXXXX0XXXXXX
003E 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXXXXXX010 XXXXXX0XX0XXXXXX
003F 0111000101000000 XXXX011000110000 111111011011XXX0 XXXXXXXXXXXXX010 XX0XXXXXX0XXXXXX
0040 1101110101100101 XXXX000011101111 111111XXXXXXXXXX XXXXXXXXXXXXX010 0XXXXXXXX0XXXXXX
0041 1101110101100100 XXXX000011101111 111111XXXXXXXXXX XXXXXX0XXXX0X010 00XXXX0XX0XXXXXX
0042 0111000101000000 XXXX011000110001 000010011011XXX0 XXXXXXXXXXXXX010 XX0XXXXXX0XXXXXX
0043 1101100101100101 XXXX000011101111 111111XXXXXXXXXX XXXXXXXXXXXXX010 0XXXXXXXX0XXXXXX
0044 1101110101100100 XXXX000010010000 111111XXXXXXXXXX XXXXXXX0XXX0X010 00XXXX0XX0XXXXXX
0045 0111000101000000 XXXX011011000001 000001XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0046 1101110101100101 XXXX000011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 0XXXXXXXXXXXXXXX
0047 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0048 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXX0XXXX0XX10 X0XXXXXXXXXXXXXX
0049 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
004A 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
004B 0111000101000000 XXXX011010010001 000110XXXXXXXXXX XXXXXXX0XXX0XX10 X0XXXXXXXXXXXXXX
004C 0111000101000000 XXXX011000110001 001110010110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
004D 1101100100000000 XXXX010010101111 111111011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
004E 1111100011100000 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
004F 0000000000011111 XXXX001011101111 111111XXXXXXXXXX 0XXXXXXX0XXXXX10 XXXXXXXXXXXXXXXX
0050 1101100100001000 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0051 1111100011100001 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0052 0000000000010110 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0053 0111000101000000 XXXX011000110001 011001010100XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0054 1110000110000101 XXXX110000110001 010110010000XXXX XXXXXXXXXXX0XX10 XXXXXXXXXXXXXXXX
0055 1100011111001000 XXXX010000110001 010111011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0056 1100000111001000 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0057 0111000101000000 0111011000110001 010011010011XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0058 1100001110000000 XXXX000010101111 111111011110XXXX 0XXXXXXX0XXXXX10 XXXXXXXXXXXXXXXX
0059 1101100011101001 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
```

```
005A 0000000000001101 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
005B 1101100011101010 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
005C 0000000001011001 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
005D 1101100011101011 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
005E 0000000000010111 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
005F 1101100011101100 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0060 0000000010010000 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0061 1011111100101100 XXXX010111101111 111111XXXXXXXXXX XX0XX0XXXXXXXX10 XXXXX0XXXXXXXXXX
0062 0000000000000111 XXXX000000110001 100010011110XXXX XX0XX0XXXXXXXX01 XXXXX0XXXXXXXXXX
0063 1001110000001001 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0064 0000000000001101 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0065 1001110000001010 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0066 0000000001011001 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0067 1001110000001011 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0068 0000000000010111 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0069 1101100100000111 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
006A 1101100011101100 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
006B 1110011100100011 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
006C 1101100011101101 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
006D 0000000000000110 XXXX010011000000 000100XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
006E 1101100001001000 XXXX001100010011 011011011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
006F 1011111110101000 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0070 1100011110001101 XXXX010011000000 000011XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0071 1101100001001001 XXXX001100010011 011011011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0072 1101010100110000 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0073 1100001110001101 XXXX010011000000 000110XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
```

```
0074 1101100001001010 XXXX001100010011 0110110111110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0075 1010100100101000 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0076 1100011110001101 XXXX010011000000 000100XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0077 1101100001001011 XXXX001100010011 011011011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0078 0111000100000000 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0079 1101100011101101 XXXX000111000000 000110XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
007A 0000010001110111 XXXX000100010011 100100011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
007B 1110111011100010 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
007C 1001110000100111 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
007D 0000000000001000 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
007E 1101100100001000 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
007F 1101100100001001 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0080 1101100000001001 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0081 1011100111000110 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0082 0000111111111111 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0083 1001110011000110 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0084 0000000000001111 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0085 1001110010000110 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0086 0000000010100101 XXXX000011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX01 XXXXXXXXXXXXXXXX
0087 1000000001000100 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
0088 1110000110000101 XXXX000011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 0XXXXXXXXXXXXXXX
0089 1001111100001000 XXXX010111101111 111111XXXXXXXXXX XXXXX0XXXXXXXX10 XXXXXXXXXXXXXXXX
008A 1101100001001000 XXXX001011101111 111111XXXXXXXXXX 0XXXXXXXXXXXXX10 XX0XXXXXXXXXXXXX
008B 1111110010000000 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 0XXXXXXXXXXXXXXX
008C 1001111100001001 XXXX010111101111 111111XXXXXXXXXX XXXXX0XXXXXXXX10 XXXXXXXXXXXXXXXX
008D 1101100001001001 XXXX001011101111 111111XXXXXXXXXX 0XXXXXXXXXXXXX10 XX0XXXXXXXXXXXXX
008E 1101100100000000 XXXX000010101111 111111011110XXXX 0XXXXXXX0XXXXX10 XXXXXXXXXXXXXXXX
0090 1100000111101001 XXXX110000110010 011101011110XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
0091 1100000111101011 XXXX110000110010 011000011110XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
0092 1100000111101001 XXXX110000110010 011001011110XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
0093 1100000111101011 XXXX110000110010 011011011110XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
0094 1100000111101010 XXXX110000110010 011010011110XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
0095 1100000111101010 XXXX110000110010 011011011110XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
0096 1100000111101001 XXXX110000110010 011011011110XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
0097 1101100100000110 XXXX110000110001 100011011110XXXX XXXXXXXXXXXXXX10 XXXXX0XXXXXXXXXX
0098 0111000101000000 XXXX011000110010 011110011110XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
0099 0111000101000000 XXXX011000110010 100001011110XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
```

```
009A 0111000101000000 XXXX011000110010 100011011110XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
009B 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
009C 0111000101000000 XXXX011000110010 011111011110XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
009D 1100000111101011 0111110000110001 011000010010XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
009E 1100000111101010 0111110000110001 011000010010XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
009F 1011111100101100 0111010100110001 010111010010XXXX XX0XX0XXXXXXXX10 XXXXX0XXXXXXXXXX
00A0 0000000000000111 XXXX000000110010 100000011110XXXX XX0XX0XXXXXXXX01 XXXXX0XXXXXXXXXX
00A1 1100000111101011 0111110000110010 011111010011XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
00A2 0111000101000000 XXXX011000110001 011000011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00A3 1100000111101001 0111110000110010 011111010011XXXX XXXXXXXXXXX0XX10 XXXXX0XXXXXXXXXX
00A4 0111000101000000 XXXX011000110001 011000011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00A5 1011001100101000 XXXX010100110010 110101011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00A6 1011010100101000 XXXX010100110010 110110011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00A7 1011011100101000 XXXX010100110010 110111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00A8 1011100100101000 XXXX010100110010 111000011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00A9 1011101100101000 XXXX010100110010 111001011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00AA 1011110100101000 XXXX010100110010 111010011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00AB 1011111100101000 XXXX010100110010 111011011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00AC 1010000100101000 XXXX010100110010 111100011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00AD 1010001100101000 XXXX010100110010 111101011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00AE 1010010100101000 XXXX010100110010 111110011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
```

```
 00AF 1010011100101000 XXXX010100110010 111111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00B0 1010100100101000 XXXX010100110011 000000011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00B1 1010101100101000 XXXX010100110011 000001011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00B2 1010110100101000 XXXX010100110011 000010011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00B3 1010111100101000 XXXX010100110011 000011011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00B4 1011000100101000 XXXX010100110011 000100011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00B5 0000000000000001 XXXX010100110011 000101011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00B6 0000000000000011 XXXX010100110011 000110011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00B7 0000000000000111 XXXX010100110011 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00B8 0000000000001111 XXXX010100110011 001000011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00B9 0000000000011111 XXXX010100110011 001001011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00BA 0000000000111111 XXXX010100110011 001010011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00BB 0000000001111111 XXXX010100110011 001011011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00BC 0000000011111111 XXXX010100110011 001100011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00BD 0000000111111111 XXXX010100110011 001101011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00BE 0000001111111111 XXXX010100110011 001110011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00BF 0000011111111111 XXXX010100110011 001111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00C0 0000111111111111 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00C1 0001111111111110 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00C2 0011111111111100 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00C3 0111111111111000 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00C4 1111111111110000 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00C5 1011001100101001 XXXX010100110011 010000011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00C6 1011010100101001 XXXX010100110011 010001011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00C7 1011011100101001 XXXX010100110011 010010011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00C8 1011100100101001 XXXX010100110011 010011011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00C9 1011101100101001 XXXX010100110011 010100011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00CA 1011110100101001 XXXX010100110011 010101011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00CB 1011111100101001 XXXX010100110011 010110011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00CC 1010000100101001 XXXX010100110011 010111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00CD 1010001100101001 XXXX010100110011 011000011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00CE 1010010100101001 XXXX010100110011 011001011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
 00CF 1010011100101001 XXXX010100110011 011010011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
```

```
00D0 1111111111100000 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00D1 1111111111000000 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00D2 1111111110000000 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00D3 1111111100000000 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00D4 1111111000000000 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00D5 1111110000000000 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00D6 1111100000000000 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00D7 1111000000000000 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00D8 1110000000000000 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00D9 1100000000000000 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00DA 1000000000000000 XXXX010100110010 000111011110XXXX X00XX0XXXXXXXX10 XXXXXXXXXXXXXXXX
00DB 1100111010001000 XXXX110011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00DC 1101100001001101 XXXX001111101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00DD 1101100000001100 1000010000110011 1000000100111XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00DE 1001100010000110 XXXX110011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00DF 1001111010100111 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00E0 1100110000001100 XXXX110011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00E1 1100110001001101 XXXX000111101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00E2 1100110010001000 XXXX110010010011 011101XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00E3 1101100011101100 XXXX010010101111 111111011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00E4 1101100000001100 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00E5 1001100001000110 XXXX110011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00E6 1001111001100111 XXXX110011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00E7 0111000101000000 0111010000110011 101010010011XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
00E8 1001100010000110 XXXX110011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
```

```
 00E9 1001111010100111 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
 00EA 1100110010001101 XXXX100111101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
 00EB 1100110011001100 XXXX010010010011 100101XXXXXXXXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
 00EC 1001110000000110 XXXX110010101111 111111011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
 00ED 1101111010000100 XXXX010011000000 000101XXXXXXXXXX XXXXXXXXXXXXXX10 0XX0XXXXXXXXXXXX
 00EE 0111000101000000 XXXX011000110011 101110011101XXXX XXXXXXXXXXXXXX10 XXX0XXXXXXXX0XXX
 00EF 1111100100000000 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXX0XXXXXXXX0XXX
 00F0 1111100100000000 XXXX001010010011 101110XXXXXXXXXX 0XXX0XXXXXXXXX10 XXX0XXX0XXXX0XXX
 00F1 1111100011100000 XXXX001111000010 000011XXXXXXXXXX XXXXXXXXXXXXXX10 XXX0XXXXXXXX0XXX
 00F2 1111111000000000 XXXX001100110011 110001011101XXXX XXXXXXXXXXXXXX10 XXX0XXXXXXXX0XXX
 00F3 1111100011000000 XXXX001011101111 111111XXXXXXXXXX 0XXX0XXXXXXXXX10 XXX0XXX0XXXX0XXX
 00F4 1101100001000100 XXXX001011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 0XX0XXXXXXXX0XXX
 00F5 1101110101100100 XXXX010011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXX0XXXXXXXX0XXX
 00F6 0111000101000000 XXXX011010010011 110100XXXXXXXXXX XXXX0XXXXXXXXX10 X0X0XXX0XXXX0XXX
 00F7 1101100100000100 XXXX010011000000 000001XXXXXXXXXX XXXXXXXXXXXXXX10 XXX0XXXXXXXX0XXX
 00F8 1101100001000100 XXXX001000110011 111000011101XXXX XXXXXXXXXXXXXX10 XXX0XXXXXXXX0XXX
 00F9 1101100001000100 XXXX001011101111 111111XXXXXXXXXX 0XXX0XXXXXXXXX10 XXX0XXX0XXXX0XXX
 00FA 0111000101000000 XXXX011010010011 111000XXXXXXXXXX XXXXXXXXXXXXXX10 XXX0XXXXXXXX0XXX
 00FB 0111000101000000 XXXX011011101111 111111XXXXXXXXXX XXXXXXXXXXXXXX10 XXX0XXXXXXXXXXXX
 00FC 1101100100000000 XXXX010010101111 111111011110XXXX XXXXXXXXXXXXXX10 XXXXXXXXXXXXXXXX
```

SYMBOLS

```
 A1LSW     E723
 A1MSW     0006
 A2LSW     BFA8
 A2NA      0004
 A2NDY     0014
 A2NR      000E
 A3LSW     D530
 A4LSW     A928
 ADD       0004
 ADDC      0005
 AE        0056
 AE20      0008
 AND       0006
 ASCEBC    0000
 ATTN      0010
 B         0000
 BACK      0011
 BCDEBC    0001
 BRTABL    0036
 BUSY      0012
 C         0005
 CDAI      0002
 CDRA      0004
 CDRI      0003
 CFCODE    001A
 COMP      000D
 CRAI      0005
 CRCFL1    0022
 CRCFL2    0026
 CRCMSK    C001
 CRCNIT    0010
 CT16      0009
 CYC       00DD
 CYC1      00E5
 DIV       00E4
 EBCASC    0002
 EBCBCD    0003
 EP20      000A
 ER20      000B
 ERR       0058
 EXNOR     000B
 EXOR      0008
 F1        0009
 F2        000A
 F3        000B
 FAIL      000C
 INC       000E
 INDX      0013
 KL128     7100
 KLSW      EEE2
 KM128     0477
 KMSW      0008
 L         0008
 LD2NA     0006
 LD2NR     000C
 LD2NY     0016
```

```
   LDC2NA    0007
   LDC2NR    000D
   LDC2NY    0017
   LINK      0057
   LOW       0004
   M1        0053
   M1A1      0069
   M234I     0059
   M2A2      006F
   M3A3      0072
   M4A4      0075
   MARI      000C
   MATCH1    0035
   MDAI      0007
   MDAR      0008
   MDRA      000A
   MDRI      0009
   MFIX      0063
   MOVE      000C
   MRAI      000E
   MUL       00DB
   N         0007
   N0        0000
   N1        0001
   N2        0002
   N3        0003
   N4        0004
   N5        0005
   N6        0006
   N7        0007
   N8        0008
   N9        0009
   NA        000A
   NAND      0007
   NB        000B
   NC        000C
   ND        000D
   NE        000E
   NEG       000F
   NF        000F
   NO        0001
   NOR       0009
   NOTQ      00E0
   NOZ       0000
   NRA       0001
   NRAS      0005
   NRS       0004
   NRY       0000
   NSPASS    0040
   OR        000A
   OVR       0003
   PCPREL    0007
   PF1       0016
   PF2       000D
   PF3       0059
   PF4       0017
   PM1       009F
   PM2       0015
```

```
  PM23      00A3
  PM234     009D
  PM24      00A1
  PM3       0016
  PM34      009E
  PM4       0017
  POS       00EA
  PR1A      0008
  PR1D      0009
  PR1R      000B
  PR1Y      000A
  PR2A      0000
  PR2Y      0002
  PR3A      0004
  PR3D      0006
  PR3R      0003
  PRA       0008
  PRI       000B
  PRT1A     0007
  PRTA      0004
  PRTD      0006
  PRZ       000A
  R0        0000
  R1        0001
  R10       000A
  R11       000B
  R12       000C
  R13       000D
  R14       000E
  R15       000F
  R16       0010
  R17       0011
  R18       0012
  R19       0013
  R2        0002
  R20       0014
  R21       0015
  R22       0016
  R23       0017
  R24       0018
  R25       0019
  R26       001A
  R27       001B
  R28       001C
  R29       001D
  R3        0003
  R30       001E
  R31       001F
  R4        0004
  R5        0005
  R6        0006
  R7        0007
  R8        0008
  R9        0009
  RDITCT    0041
  RDSEC1    0038
  RDSEC2    003F
  RDSEC3    0046
  RDSEC4    004E
  RDYI      000D
  RDYO      000E
  REP1      00B5
  REP10     00BE
  REP11     00BF
  REP12     00C0
  REP13     00C1
  REP14     00C2
  REP15     00C3
  REP16     00C4
  REP17     00D0
  REP18     00D1
  REP19     00D2
  REP2      00B6
  REP20     00D3
  REP21     00D4
  REP22     00D5
  REP23     00D6
  REP24     00D7
  REP25     00D8
  REP26     00D9
  REP27     00DA
  REP3      00B7
  REP4      00B8
  REP5      00B9
  REP6      00BA
  REP7      00BB
  REP8      00BC
  REP9      00BD
  RF1       0006
  RF2       0009
  RF3       000A
  RL        0005
  RM1       00C5
  RM10      00CE
  RM11      00CF
  RM2       00C6
  RM3       00C7
  RM4       00C8
  RM5       00C9
  RM6       00CA
  RM7       00CB
  RM8       00CC
  RM9       00CD
  RONCZ     0003
  RSTNA     0001
  RSTND     0011
  RSTNR     000E
  RTAA      001D
  RTAR      0000
  RTAY      001C
  RTDA      0019
  RTDR      0001
  RTDY      0018
  RTRA      000C
  RTRR      000F
  RTRY      000E
```

S2NA	0005
S2NDY	0015
S2NR	000F
SAMD	0014
SCBIBL	000F
SECTIO	0000
SECTL1	0029
SECTL2	0033
SETNA	0002
SETND	0012
SETNR	000D
SF1	0006
SF2	0009
SF3	000A
SHA	0006
SHD	0007
SHDN1	0005
SHDNC	0007
SHDNL	0006
SHDNOV	0008
SHDNZ	0004
SHDR	0007
SHRR	0006
SHUP1	0001
SHUPL	0002
SHUPZ	0000
SL	0005
SOA	0004
SOAR	0004
SOD	0006
SODR	0006
SOI	0007
SOIR	0007
SONZC	0003
SORA	0000
SORR	000B
SORS	0003
SORY	0002
SOSE	000A
SOSER	000A
SOZ	0008
SOZE	0009
SOZER	0009
SOZR	0008
SUBK	007B
SUBR	0000
SUBRC	0001
SUBS	0002
SUBSC	0003
SUCC	000F
TAB1	0090
TAB2	00A5
TC	000A
TF1	0012
TF2	0014
TF3	0016
TL	0010
TLOW	0008

```
 TM1       009B
 TM23      009A
 TM24      0099
 TM34      0098
 TN        000E
 TNO       0002
 TNOZ      0000
 TOAI      0002
 TOAIR     0002
 TODA      0001
 TODAR     0001
 TODI      0005
 TODIR     0005
 TODRA     0003
 TODRR     000F
 TODRY     000B
 TORAA     0000
 TORAR     000C
 TORAY     0008
 TORIA     0002
 TORIR     000E
 TORIY     000A
 TOVR      0006
 TSTNA     0000
 TSTND     0010
 TSTNR     000F
 TZ        0004
 TZC       000C
 W         0001
 WRSEC1    00ED
 WRSEC2    00EE
 WRSEC3    00F1
 WRSEC4    00F4
 WRSEC5    00F8
 XOR       0087
 XORMEM    007E
 Z         0002
 ZC        0006

TOTAL PHASE 2 ERRORS =    0
```

AM29CPU.DEF

& AM29203.SRC

```
;       *****************************************************
;
;
; ******************************************
; Am2900 FAMILY MNEMONICS - AM29CPU.DEF FILE
; ******************************************
;
; This file was created to serve as a master cpu file.  The user must
; edit it as required for his/her particular application.
;
;
;
; Anyone finding errors in this file is requested to send a marked listing
; or portion thereof to:  AMD CUSTOMER EDUCATION CENTER
;                         490-A LAKESIDE DRIVE
;                         PO BOX 453  MS#71
;                         SUNNYVALE, CA  94086
;
;
; Advanced Micro Devices reserves the right to make changes in its product
; without notice in order to improve design and performance characteristics.
; The company assumes no responsibility for the use of any circuits or programs
; described herein.
```

```
;
;
;
 WORD 64
;
; 13 DECEMBER 1976 JRM
; UPDATED SEPT 28, 1977
; UPDATED APRIL 16, 1981 DEW
; UPDATED MAY 15, 1981 DEW
; UPDATED JAN 12, 1982 DEW
;
;       INDEX:
;       Am2901 [1], [2], [3]
;       Am2914 [5]
;       Am2930 [6]
;       Am2932 [7]
;       Am2940 [8]
;       Am2903 [9], [10], [11], [12]
;       Am29203 [9], [11], [13], [14]
;       THREE-ADDRESS EXPANDED MEMORY SAMPLE [15], [16], [17], [18]
;       MISCELLANEOUS FIELDS
;               REGISTERS [R]
;               CARRY BIT (NOT Am2904) [19]
;       Am2910 [20]
;       Am29811 ADDED INSTRUCTION
;       Am2925 MICROPROGRAMMABLE CLOCK CYCLE SELECT + CONTROL LINES [21]
;       Am2904 SHIFT EXAMPLES
;       Am2904 STATUS REGISTER EXAMPLES
;       Am2904 CONDITION CODE OUTPUT EXAMPLES
;       MORE MISCELLANEOUS FIELDS
;               OUTPUT ENABLE
;               INSTRUCTION ENABLE
;               CONDITION CODE MUX - DATA MONITOR PROBLEM ADDITION
;       EXAMPLES OF DEF STATEMENTS - TWO ADDRESS - NONEXPANDED MEMORY
```

```
;
; * * * * * * * * * * * * * * * * * * * * * * * * *
;
; Am2901 INSTRUCTION SET
;
; * * * * * * * * * * * * * * * * * * * * * * * * *
;
; Am2901 SOURCE OPERANDS (R S)                        [1]
;
AQ:      EQU     Q#0
AB:      EQU     Q#1
ZQ:      EQU     Q#2
ZB:      EQU     Q#3
ZA:      EQU     Q#4
DA:      EQU     Q#5
DQ:      EQU     Q#6
DZ:      EQU     Q#7
;
; Am2901 ALU FUNCTIONS (R FUNCTION S)                 [2]
; *** TO USE: DELETE THE .01 AND DELETE THE Am2903/29203 INSTRUCTION SETS ***
;
ADD.01:          EQU     Q#0
SUBR.01:         EQU     Q#1
SUBS.01:         EQU     Q#2
OR.01:           EQU     Q#3
AND.01:          EQU     Q#4
NOTRS.01:        EQU     Q#5
EXOR.01:         EQU     Q#6
EXNOR.01:        EQU     Q#7
;
; Am2901 DESTINATION CONTROL                          [3]
;
QREG.01:         EQU     Q#0
NOP.01:          EQU     Q#1
RAMA.01:         EQU     Q#2
RAMF.01:         EQU     Q#3
RAMQD.01:        EQU     Q#4
RAMD.01:         EQU     Q#5
RAMQU.01:        EQU     Q#6
RAMU.01:         EQU     Q#7
;
```

```
;
;
; EXAMPLE SUB STATEMENTS
;
ALU:    SUB     3VQ#0, 3VQ#0, 3VQ#1, 2VB#00     ; Am2901 FIELDS ONLY
;               SOURCE FUNCT  DEST   CARRY
;
REGS:   SUB     4VH#0, 4VH#0                    ; ALU REGISTER ADDRESSES
;               RA     RB
;
SAMPLE: DEF     19X, ALU, REGS, 2VB#00, 24X     ; USAGE OF SUB STATEMENTS
;               SEQ             MUX  etc.
;
; Although the file is written with DEF statements which include the
; very same fields, SUB statements are included here for a reference
; example.  The use of SUB statements to partially define a microword
; allows a SUB to be defined immediatly after the mnemonics of the
; individual fields.  The DEF statement may be defined after all of the
; SUB statements to which it refers are defined.  The DEF statements are then
; more readable. The comments which would appear with a DEF statement would
; then appear with the SUB statement (what the field is, what the default
; represents, etc.).  This file has been written with both versions to
; provide a reference.  Whichever method is chosen, DOCUMENT THE STATEMENTS.
;
; NOTE THAT, if the sequencer fields had been defined, the 19X above could
; have been replaced by the sequencer SUB statement name.
```

```
;
; * * * * * * * * * * * * * * * * 
;
; Am2914 INSTRUCTION SET                           [5]
;
; * * * * * * * * * * * * * * * * 
;
MCLR:    EQU     H#0     ; MASTER CLEAR
CLRIN:   EQU     H#1     ; CLEAR ALL INTERRUPTS
CLRMB:   EQU     H#2     ; CLEAR INTERRUPTS FROM M-BUS
CLRMR:   EQU     H#3     ; CLEAR INTERRUPTS FROM MASK REGISTER
CLRVC:   EQU     H#4     ; CLEAR INTERRUPT FROM LAST VECTOR READ
RDVC:    EQU     H#5     ; READ VECTOR
RDSTA:   EQU     H#6     ; READ STATUS REGISTER
RDM:     EQU     H#7     ; READ MASK REGISTER
SETM:    EQU     H#8     ; SET MASK REGISTER
LDSTA:   EQU     H#9     ; LOAD STATUS REGISTER
BCLRM:   EQU     H#A     ; BIT CLEAR MASK REGISTER
BSETM:   EQU     H#B     ; BIT SET MASK REGISTER
CLRM:    EQU     H#C     ; CLEAR MASK REGISTER
DISIN:   EQU     H#D     ; DISABLE INTERRUPT REQUEST
LDM:     EQU     H#E     ; LOAD MASK REGISTER
ENIN:    EQU     H#F     ; ENABLE INTERRUPT REQUEST
;
```

```
 ; * * * * * * * * * * * * * * * * * * 
 ;
 ; Am2930 PROGRAM CONTROL UNIT                      [6]
 ;
 ; * * * * * * * * * * * * * * * * * * 
 ;
 ; NON-CONDITIONAL INSTRUCTIONS
 ;
PRST:   EQU     5H#00:  ; RESET
FPC:    EQU     5H#01:  ; FETCH PC
FR:     EQU     5H#02:  ; FETCH R
FD:     EQU     5H#03:  ; FETCH D
FRD:    EQU     5H#04:  ; FETCH R PLUS D
FPD:    EQU     5H#05:  ; FETCH PC PLUS D
FPR:    EQU     5H#06:  ; FETCH PC PLUS R
FSD:    EQU     5H#07:  ; FETCH S PLUS D
FPLR:   EQU     5H#08:  ; FETBH PC, LOAD R
FRDR:   EQU     5H#09:  ; FETCH R PLUS D, LOAD R
PLDR:   EQU     5H#0A:  ; LOAD R
PSHP:   EQU     5H#0B:  ; PUSH PC
PSHD:   EQU     5H#0C:  ; PUSH D
POPS:   EQU     5H#0D:  ; POP S
POPP:   EQU     5H#0E:  ; POP PC
PHLD:   EQU     5H#0F:  ; HOLD
;
; CONDITIONAL INSTRUCTIONS - FAIL TEST, EXECUTE FPC
;
JMPR:   EQU     5H#10:  ; JUMP R
JMPD:   EQU     5H#11:  ; JUMP D
JMPZ:   EQU     5H#12:  ; JUMP ZERO
JPRD:   EQU     5H#13:  ; JUMP R PLUS D
JPPD:   EQU     5H#14:  ; JUMP PC PLUS D
JPPR:   EQU     5H#15:  ; JUMP PC PLUS R
JSBR:   EQU     5H#16:  ; JUMP SUBROUTINE R
JSBD:   EQU     5H#17:  ; JUMP SUBROUTINE D
JSBZ:   EQU     5H#18:  ; JUMP SUBROUTINE ZERO
JSRD:   EQU     5H#19:  ; JUMP SUBROUTINE R PLUS D
JSPD:   EQU     5H#1A:  ; JUMP SUBROUTINE PC PLUS D
JSPR:   EQU     5H#1B:  ; JUMP SUBROUTINE PC PLUS R
RTS:    EQU     5H#1C:  ; RETURN S
RTSD:   EQU     5H#1D:  ; RETURN S PLUS D
CHLD:   EQU     5H#1E:  ; HOLD
PSUS:   EQU     5H#1F:  ; SUSPEND
;
```

```
 ; * * * * * * * * * * * * * * * * * *
 ;
 ; Am2932 PROGRAM CONTROL UNIT                        [7]
 ;
 ; * * * * * * * * * * * * * * * * * *
 ;
 ; TO USE: DELETE THE .32  FROM Am2932 MNEMONICS IF NEEDED
 ; AND DELETE THE Am2930 INSTRUCTION SET
 ;
PRST.32:        EQU     H#0     ; RESET
PSUS.32:        EQU     H#1     ; SUSPEND
PSHD.32:        EQU     H#2     ; PUSH D
POPS.32:        EQU     H#3     ; POP STACK
FPC.32:         EQU     H#4     ; FETCH PC
JMPD.32:        EQU     H#5     ; JUMP D
PSHP.32:        EQU     H#6     ; PUSH PC
RTS.32:         EQU     H#7     ; RETURN STACK
FR.32:          EQU     H#8     ; FETCH R
FPR.32:         EQU     H#9     ; FETCH PC PLUS R
FPLR.32:        EQU     H#A     ; FETCH PC, LOAD R
JMPR.32:        EQU     H#B     ; JUMP R
JPPR.32:        EQU     H#C     ; JUMP PC PLUS R
JSBR.32:        EQU     H#D     ; JUMP SUBROUTINE R
JSPR.32:        EQU     H#E     ; JUMP SUBROUTINE PC PLUS R
PLDR.32:        EQU     H#F     ; LOAD R
```

```
; * * * * * * * * * * * * * * * * * * *
;
; Am2940 DMA CONTROL UNIT                         [8]
;
; * * * * * * * * * * * * * * * * * * *
; INSTRUCTIONS
;
WRCR:   EQU     Q#0     ; WRITE CONTROL REGISTER
RDCR:   EQU     Q#1     ; READ CONTROL REGISTER
RDWC:   EQU     Q#2     ; READ WORD COUNTER
RDAC:   EQU     Q#3     ; READ ADDRESS COUNTER
REIN:   EQU     Q#4     ; REINITIALIZE COUNTERS
LDAD:   EQU     Q#5     ; LOAD ADDRESS
LDWC:   EQU     Q#6     ; LOAD WORD COUNT
ENCT:   EQU     Q#7     ; ENABLE COUNTERS
;
; CONTROL MODE BYTE
; NOTE - BITS 3 THROUGH 7 ARE DON'T CARE
;
WC1I:   EQU     8Q#0%   ; WORD COUNT EQUALS ONE, INCREMENT ADDRESS COUNTER
WCCI:   EQU     8Q#1%   ; WORD COUNT COMPARE, INCREMENT ADDRESS COUNTER
ADCI:   EQU     8Q#2%   ; ADDRESS COMPARE, INCREMENT ADDRESS COUNTER
WCOI:   EQU     8Q#3%   ; WORD COUNTER CARRY OUT, INCREMENT ADDRESS COUNTER
WC1D:   EQU     8Q#4%   ; WORD COUNT EQUALS ONE, DECREMENT ADDRESS COUNTER
WCCD:   EQU     8Q#5%   ; WORD COUNT COMPARE, DECREMENT ADDRESS COUNTER
ADCD:   EQU     8Q#6%   ; ADDRESS COMPARE, DECREMENT ADDRESS COUNTER
WCOD:   EQU     8Q#7%   ; WORD COUNTER CARRY OUT, DECREMENT ADDRESS COUNTER
;
```

```
 ; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
 ; Am2903 INSTRUCTION SET
 ;
 ; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
 ; ALU SOURCE OPERANDS (EA, I0, OEB)                   [9]
 ; 16 REGISTER - TWO ADDRESS VERSION
 ;
 ; NOTE: USE FOR BOTH THE Am2903 AND THE Am29203 * * *
 ;
 ; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
 ;
RAMAB:          EQU     Q#0             ; RAM A PORT, RAM B PORT
RAMADB:         EQU     Q#1             ; RAM A PORT, DATA BUS B
RAMAQ:          EQU     Q#2             ; OR Q#3 - RAM A PORT, Q REGISTER
DARAMB:         EQU     Q#4             ; DATA BUS A, RAM B PORT
DADB:           EQU     Q#5             ; DATA BUS A, DATA BUS B
DAQ:            EQU     Q#6             ; OR Q#7 - DATA BUS A, Q REGISTER
 ; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
 ; ALU FUNCTIONS - NORMAL MODE (I4, I3, I2, I1, I0 ALL NOT 0)
 ; Am2903 ONLY !  * * * TO USE, DELETE THE .03 AND TAG OR
 ; DELETE THE Am29203 FUNCTIONS * * *
 ;                                                     [10]
 ; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
SPECL.03:       EQU     H#0             ; I0 MUST BE LOW
HIGH.03:        EQU     H#0             ; I0 MUST BE HIGH (Q REGISTER SELECT)
SUBR.03:        EQU     H#1             ; F = S - R - 1 + Cin
SUBS.03:        EQU     H#2             ; F = R - S - 1 + Cin
ADD.03:         EQU     H#3             ; F = R + S + Cin
INCRS.03:       EQU     H#4             ; F = S + Cin
INCSNON.03:     EQU     H#5             ; F = NOT S + Cin
INCRR.03:       EQU     H#6             ; F = R + Cin
INCRNON.03:     EQU     H#7             ; F = NOT R + Cin
LOW.03:         EQU     H#8             ; F = LOW
NOTRS.03:       EQU     H#9             ; F = NOT R AND S
EXNOR.03:       EQU     H#A             ; F = R EXNOR S
EXOR.03:        EQU     H#B             ; F = R EXOR S
AND.03:         EQU     H#C             ; F = R AND S
NOR.03:         EQU     H#D             ; F = R NOR S
NAND.03:        EQU     H#E             ; F = R NAND S
OR.03:          EQU     H#F             ; F = R OR S
```

```
; * * * * * * * * * * * * * * * * * * * * * * *
;
; ALU DESTINATION CONTROL ( I8 - I7 - I6 - I5)          [11]
;     NORMAL FUNCTIONS
;
; NOTE: USE FOR BOTH THE Am2903 AND THE Am29203 * * * *
;
; * * * * * * * * * * * * * * * * * * * * * * *
;
RAMDA:          EQU     H#0          ; F TO RAM, ARITHMETIC DOWN SHIFT
RAMDL:          EQU     H#1          ; F TO RAM, LOGICAL DOWN SHIFT
RAMQDA:         EQU     H#2          ; DOUBLE PRECISION ARITHMETIC DOWN SHIFT
RAMQDL:         EQU     H#3          ; DOUBLE PRECISION LOGICAL DOWN SHIFT
RAM:            EQU     H#4          ; F TO RAM WITH PARITY
QD:             EQU     H#5          ; F TO Y, DOWN SHIFT Q
LOADQ:          EQU     H#6          ; F TO Q WITH PARITY
RAMQ:           EQU     H#7          ; F TO RAM AND Q WITH PARITY
RAMUPA:         EQU     H#8          ; F TO RAM, ARITHMETIC UP SHIFT
RAMUPL:         EQU     H#9          ; F TO RAM, LOGICAL UP SHIFT
RAMQUPA:        EQU     H#A          ; DOUBLE PRECISION ARITHMETIC UP SHIFT
RAMQUPL:        EQU     H#B          ; DOUBLE PRECISION LOGICAL UP SHIFT
YBUS:           EQU     H#C          ; F TO Y ONLY
QUP:            EQU     H#D          ; F TO Y, UP SHIFT Q
SIGNEXT:        EQU     H#E          ; SIO0 TO Yi
RAMEXT:         EQU     H#F          ; F TO Y, SIGN EXTEND LEAST SIG. BYTE
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; SPECIAL FUNCTIONS (I8-I7-I6-I5)                [12]
;
;
; Am2903 FUNCTIONS ONLY
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
MULT.03:        EQU     H#0             ; UNSIGNED MULTIPLY
TWOMULT.03:     EQU     H#2             ; TWO'S COMPLEMENT MULTIPLY
TWOLAST.03:     EQU     H#6             ; TWO'S COMPLEMENT MULTIPLY LAST STEP
INCRMNT.03:     EQU     H#4             ; INCREMENT BY 1 + Cin
SGNTWO.03:      EQU     H#5             ; SIGN MAGNITUDE-TWO'S COMPL CONVERSION
SLN.03:         EQU     H#8             ; SINGLE LENGTH NORMALIZE
DLN.03:         EQU     H#A             ; DOUBLE LENGTH NORMALIZE
DIVFRST.03:     EQU     H#A             ; TWO'S COMPLEMENT DIVIDE FIRST STEP
DIVIDE.03:      EQU     H#C             ; TWO'S COMPLEMENT DIVIDE MIDDLE STEPS
DIVLAST.03:     EQU     H#E             ; TWO'S COMPLEMENT DIVIDE LAST STEP
;
```

```
 ; ***********************************************************************
 ; NEW!   NEW!   NEW!   NEW!   NEW!   NEW!   NEW!   NEW!   NEW!   NEW!
 ;
 ; Am29203 INSTRUCTION SET - 5/15/81 - DEW
 ;
 ; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
 ; Am29203 SOURCE SELECT - SEE Am2903 SOURCE SELECTION (SAME)
 ; TWO - ADDRESS OPERATION ONLY
 ;
 ; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
 ; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
 ; Am29203 ALU FUNCTIONS - NORMAL MODE ( I4, I3, I2, I1, I0 ALL NOT 0 )
 ;                                              [13]
 ;
 ; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
 ; NOTE DIFFERENCE FROM Am2903 - ** FIVE ** INSTRUCTION FIELDS USED
 ; IN THE DATA SHEET TABLE - THEREFORE RESTRICTIONS ON THE SOURCE SELECTION
 ; APPLY.   THE FUNCTIONS ARE GROUPED ACCORDING TO THE RESTRICTION
 ;
 ;
 ; * * THE FOLLOWING DO NOT HAVE RESTRICTIONS ON THE SOURCE SELECTION * *
 ;
SUBR:          EQU     H#1          ; Fi = Si - Ri - 1 + Cin
SUBS:          EQU     H#2          ; Fi = Ri - Si - 1 + Cin
ADD:           EQU     H#3          ; Fi = Ri + Si + Cin
INCRS:         EQU     H#4          ; Fi = Si + Cin
INCRSNON:      EQU     H#5          ; Fi = ~Si + Cin
NOTRS:         EQU     H#9          ; Fi = NOT Ri AND Si
EXNOR:         EQU     H#A          ; Fi = Ri EXNOR Si
EXOR:          EQU     H#B          ; Fi = Ri EXOR Si
AND:           EQU     H#C          ; Fi = Ri AND Si
NOR:           EQU     H#D          ; Fi = Ri NOR Si
NAND:          EQU     H#E          ; Fi = Ri NAND Si
OR:            EQU     H#F          ; Fi = Ri OR Si
```

```
;
; * * THE FOLLOWING REQUIRE THAT RAMAQ OR DAQ BE THE SELECTED SOURCE * *
;
HIGH:           EQU     H#0             ; Fi = HIGH
INCRR:          EQU     H#6             ; Fi = Ri + Cin
INCRNON:        EQU     H#7             ; Fi = ~Ri + Cin
LOW:            EQU     H#8             ; Fi = LOW
;
; * * THE FOLLOWING REQUIRE THAT Q IS ** NOT ** IN THE SELECTED SOURCE * *
;
SPECL:          EQU     H#0             ; SPECIAL FUNCTIONS
RESRVD.1:       EQU     H#6             ; RESERVED FOR LATER USE
RESRVD.2:       EQU     H#7             ; RESERVED FOR LATER USE
SPECIL.2:       EQU     H#8             ; SPECIAL FUNCTIONS (MULTI-BCD)
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; DESTINATION CONTROL - SEE Am2903 DESTINATION (SAME)
;
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; Am29203 SPECIAL FUNCTIONS                     [14]
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; INSTRUCTION LINES I8, I7, I6, I5
;
; THE FOLLOWING ARE USED WITH SPECL:
;
MULT:           EQU     H#0     ; UNSIGNED MULTIPLY
TWOMULT:        EQU     H#2     ; TWO'S COMPLEMENT MULTIPLY
TWOLAST:        EQU     H#6     ; TWO'S COMPLEMENT MULTIPLY - LAST STEP
DECRMNT:        EQU     H#3     ; DECREMENT BY 1 OR 2
INCRMNT:        EQU     H#4     ; INCREMENT BY 1 OR 2
SGNTWO:         EQU     H#5     ; SIGN MAGNITUDE-TWO'S COMPLEMENT CONVERSION
SLN:            EQU     H#8     ; SINGLE LENGTH NORMALIZE
DLN:            EQU     H#A     ; DOUBLE LENGTH NORMALIZE
DIVFRST:        EQU     H#A     ; TWO'S COMPLEMENT DIVIDE - FIRST STEP
DIVIDE:         EQU     H#C     ; TWO'S COMPLEMENT DIVIDE - MIDDLE STEPS
DIVLAST:        EQU     H#E     ; TWO'S COMPLEMENT DIVIDE - LAST STEP
BCD.BIN:        EQU     H#1     ; BCD TO BINARY CONVERSION
BCD.DIV2:       EQU     H#7     ; BCD DIVIDE BY TWO
BIN.BCD:        EQU     H#9     ; BINARY TO BCD CONVERSION
BCD.ADD:        EQU     H#B     ; BCD ADD
BCD.SUBS:       EQU     H#D     ; BCD SUBTRACT Fi = Ri - Si - 1 + Cin [BCD]
BCD.SUBR:       EQU     H#F     ; BCD SUBTRACT Fi = Si - Ri - 1 + Cin [BCD]
;
; THE FOLLOWING ARE USED WITH SPECIL.2:
;
MULTIBCD:       EQU     H#1     ; MULTIPRECISION BCD TO BINARY CONVERSION
MULTIBIN:       EQU     H#9     ; MULTIPRECISION BINARY TO BCD CONVERSION
;
```

```
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
;
; DEFINITION FILE FOR FIGURE 29, PAGE 2-57, 1980 DATA BOOK
;
; EXPANDED MEMORY FOR THE Am2903 USING THE Am29705
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; ONLY THE SOURCE FIELDS CHANGE - EXPANDED
; A THIRD ADDRESS FIELD WAS ALSO ADDED (ADDRESS C)
;
;
;
; SOURCE OPERANDS                      [15]
; ADDED A ADDRESS BITS (A6-A5-A4)
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
;
AINALU: EQU     Q#0               ; A ADDRESSES 2903 REGISTERS
AIS7051:EQU     Q#1               ; A ADDRESSES FIRST 29705 ADDITION
AIS7052:EQU     Q#2               ; A ADDRESSES SECOND 29705 ADDITION
ACONST: EQU     Q#3               ; A ADDRESSES CONSTANT PROM
ABUS:   EQU     Q#4               ; A FROM BUS
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; ADDED B ADDRESS BITS (B5-B4)         [16]
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
;
BINALU: EQU     B#00              ; B ADDRESSES 2903 REGISTERS
BIS7051:EQU     B#01              ; B ADDRESSES FIRST 29705 ADDTION
BIS7052:EQU     B#10              ; B ADDRESSES SECOND 29705 ADDITION
BBUS:   EQU     B#11              ; B FROM BUS
;
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * *
;
; THREE ADDRESS OPERATION - THIRD ADDRESS FIELD
; ADDED C ADDRESS BITS (C5-C4)          [17]
; * * * * * * * * * * * * * * * * * * * * * * * * *
;
;
CIN2903:EQU     B#00              ; C ADDRESSES 2903 REGISTERS
CIS7051:EQU     B#01              ; C ADDRESSES FIRST 29705 ADDITION
CIS7052:EQU     B#10              ; C ADDRESSES SECOND 29705 ADDITION
CBUS:   EQU     B#11              ; C TO B BUS OUT
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; IO SOURCE SELECT FIELD REPLACES EA-IO-OEB THREE-BIT FIELD
;                                       [18]
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
QREGSEL:EQU     B#1               ; SOURCE IS Q REGISTER
NONQREG:EQU     B#0               ; SOURCE IS RAMB OR B.BUS
;
```

```
 ; * * * * * * * * * * * * * * * * * * * * * *
 ;
 ; MISCELLANEOUS FIELDS
 ;
 ; REGISTERS                           [R]
 ; * * * * * * * * * * * * * * * * * * * * * *
 ;
 R0:            EQU     H#0
 R1:            EQU     H#1
 R2:            EQU     H#2
 R3:            EQU     H#3
 R4:            EQU     H#4
 R5:            EQU     H#5
 R6:            EQU     H#6
 R7:            EQU     H#7
 R8:            EQU     H#8
 R9:            EQU     H#9
 R10:           EQU     H#A
 R11:           EQU     H#B
 R12:           EQU     H#C
 R13:           EQU     H#D
 R14:           EQU     H#E
 R15:           EQU     H#F
 ;
 ; * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
 ; CARRY BIT (2 BITS FOR NOW)          [19]
 ;
 ; * * * * * * * * * * * * * * * * * * * * * * * * * *
 ;
 ;
 CARRY:         EQU     B#01
 NOCARRY:       EQU     B#00          ;IMAGINATIVE!
 IC:            EQU     B#10          ; Cin is Cout
 Z:             EQU     B#11          ; Z is Cin
 ;
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; Am2910 MICROPROGRAM CONTROLLER INSTRUCTION SET [20]
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
JZ:             EQU     H#0             ; RESET STACK, MICROPC, ADDRESS
CJS:            EQU     H#1             ; COND JUMP SUBROUTINE, PUSH STACK
JMAP:           EQU     H#2             ; UNCOND JUMP TO MEMORY MAP (Di)
CJP:            EQU     H#3             ; COND JUMP PIPELINE
PUSH:           EQU     H#4             ; PUSH STACK, LOAD REG MAYBE, CONT
JSRP:           EQU     H#5             ; JUMP SUB FROM REG (F) OR PIPE(T)
CJV:            EQU     H#6             ; COND JUMP TO VECTOR INTER (Di)
JRP:            EQU     H#7             ; JUMP TO REG (F) OR PIPE (T)
RFCT:           EQU     H#8             ; DO LOOP REPEAT UNTIL CTR=0 - STACK
RPCT:           EQU     H#9             ; DO LOOP UNTIL CTR=0 - PIPE
CRTN:           EQU     H#A             ; COND RETURN, POP STACK (T)
CJPP:           EQU     H#B             ; COND JUMP PIPELINE, POP STACK
LDCT:           EQU     H#C             ; LOAD REGISTER, CONTINUE
LOOP:           EQU     H#D             ; DO LOOP UNTIL TEST=T - STACK
CONT:           EQU     H#E             ; CONTINUE
TWB:            EQU     H#F             ; THREE WAY (DEAD MAN TIMER!)
;
; * * * * * * * * * * * * * * * * *
;
; Am29811 INSTRUCTION SET
;
; * * * * * * * * * * * * * * * * *
;
; NOTE: THE SAME AS THE Am2910 EXCEPT TWB IS REPLACED BY JP
; AND ALL TESTS ARE ACTIVE HIGH RATHER THAN ACTIVE LOW
;
JP:             EQU     H#F             ; UNCONDITIONAL JUMP PIPELINE
;
;
; EXAMPLE SEQUENCER SUB STATEMENT
;
SEQR:    SUB    4VH#E, 3VQ#0,  12V$X    ;
;               INSTR  MUX     ADDR
;               CONT    ?       #
;
;
;
```

```
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; Am2925 CYCLE LENGTH SELECT                        [21]
; System Clock Generator and Driver
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; THE FOLLOWING ARE THE CYCLE LENGTH CODES  (PRELIM)
;                                                           EXAMPLE CYCLE (1 OF 4)
CLA:    EQU     Q#0     ; 3 CLOCK PERIODS                   100ns AT 30MHz
CLB:    EQU     Q#1     ; 4                                 160ns AT 25MHz
CLC:    EQU     Q#5     ; 5                                 200ns AT 25MHz
CLD:    EQU     Q#7     ; 6                                 200ns AT 30MHz
CLE:    EQU     Q#3     ; 7                                 280ns AT 25MHz
CLF:    EQU     Q#2     ; 8                                 320ns AT 25MHz
CLG:    EQU     Q#6     ; 9                                 300ns AT 30MHz
CLH:    EQU     Q#4     ; 10 CLOCK PERIODS                  322ns AT 31MHz
;                                              (max crystal frequency is 311MHz)
;
; OTHER CONTROL LINES FOR THE Am2925
; INCOMPLETELY DEFINED AT PRESENT (IN THIS FILE)
;
FIRST.25:       EQU     B#1             ;
LAST.25:        EQU     B#0             ;
;
HALT:           EQU     B#00            ;
NOHALT:         EQU     B#00            ;
;
SINGLSTP:       EQU     B#00            ;
RUN:            EQU     B#00            ;
;
WAITREQ:        EQU     B#0             ;
NOWAITRQ:       EQU     B#1             ;
;
READY:          EQU     B#0             ;
NOTREADY:       EQU     B#1             ;
;
INITIALIZE:     EQU     B#0             ;
NO.INIT:        EQU     B#1             ;
;
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; Am2904 SHIFT INSTRUCTIONS (I9-I8-I7-I6 AND SE)
;    I10 IS TIED TO I8 OF Am2903/29203
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
;
; DOWN SHIFTING
;
SDZRZQ:         EQU     H#0             ; Z->RN; Z->QN
SDOROQ:         EQU     H#1             ; 1->RN; 1->QN
SLN.RECOVER:    EQU     H#2             ; 0->RN; R0->Mc; MN->QN
DDOR:           EQU     H#3             ; 1->RN; R0->QN
DDMCR:          EQU     H#4             ; Mc->RN; R0->QN
DLN.RECOVER:    EQU     H#5             ; MN->RN; R0->QN
DDZR:           EQU     H#6             ; 0->RN; R0->QN
DDZRQMC:        EQU     H#7             ; 0->RN; R0->QN; Q0->Mc
SDROTMC:        EQU     H#8             ; ROT.R; R0->Mc; ROT.Q
SDROTC:         EQU     H#9             ; ROT.R WITH Mc; ROT.Q
SDROT:          EQU     H#A             ; ROT.R; ROT.Q
SDIC:           EQU     H#B             ; Ic->RN; R0->QN
DDROTC:         EQU     H#C             ; Mc->RN; R0->QN; Q0->Mc
DDROTMC:        EQU     H#D             ; Q0->RN; R0->QN; Q0->Mc
DDINIOVR:       EQU     H#E             ; IN EXOR IOVR -> RN; R0->QN
DDROT:          EQU     H#F             ; DOUBLE PRECISION ROTATE DOWN
;
; UP SHIFTING (INCOMPLETE)
;
SURZQZ:         EQU     H#2             ; R0<-0; Q0<-0
;
; SHIFT ENABLES
;
SE.EN:          EQU     B#0             ; ENABLE SHIFTING
SE.DIS:         EQU     B#1             ; DISABLE SHIFTING
```

```
;
;
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; Am2904 STATUS REGISTER INSTRUCTION CODES
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; MACHINE STATUS REGISTER INSTRUCTION CODES
;    I5-I4-I3-I2-I1-I0 AND EZ-EC-EN-EOVR-CEM ENABLES
; MICRO STATUS REGISTER INSTRUCTION CODES
;    I5-I4-I3-I2-I1-I0 AND CEu ENABLE
;
; THE FOLLOWING TAKES THESE ALL TOGETHER - YOU MAY WISH TO DO THIS ANOTHER WAY
;
; ORDER: 543 210 ZCNOVR CEM CEu
;        Q#  Q#  H#     B#  B#
;
ONELEVEL:       EQU     12Q#0000        ; Y -> MSR; MSR -> USR
SET.MSR:        EQU     12Q#0101        ; SET MACRO STATUS ONLY
SET.USR:        EQU     12Q#0176        ; SET MICRO STATUS ONLY
SWAP.REG:       EQU     12Q#0200        ; MSR <--> USR
;
LOAD.MSR:       EQU     12Q#2001        ; ALU STATUS -> MSR
; THE ABOVE IS ONE OF SEVERAL CODES - YOU DON'T NEED THEM ALL!
;
LOAD.USR:       EQU     12Q#2076        ; ALU STATUS -> USR
; DITTO!
;
LOAD.BOTH:      EQU     12Q#2000        ; ALU -> MSR, USR
; AGAIN DITTO!
;
LDINVRTM:       EQU     12Q#3001        ; ALU -> MSR; Ic INVERTED
LDINVRTU:       EQU     12Q#3076        ; ALU -> USR; Ic INVERTED
LOAD.INVERT:    EQU     12Q#3000        ; ALU -> MSR, USR; Ic INVERTED
;
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; Am2904 CONDITION CODE OUTPUT INSTRUCTION CODES
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; caution!  I5-I4-I3-I2-I1-I0 ARE ALSO USED FOR TESTING!!!!
; ENABLE TESTING VIA OEct ENABLE
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
TESTMZ:         EQU     12Q#4477        ; NO STATUS OPERATION
TESTMOVR:       EQU     12Q#4677        ; NO STATUS OPERATION
TESTMC:         EQU     12Q#5277        ; NO STATUS OPERATION
TESTMN:         EQU     12Q#5677        ;
TEST.IOVR:      EQU     12Q#6677        ;
TEST.IC:        EQU     12Q#7277        ;
;
;
; TEST ENABLE
;
OECTEN:         EQU     B#0
OECTDIS:        EQU     B#1
;
;
; OUTPUT ENABLE
;
OEYEN:          EQU     B#0
OEYDIS:         EQU     B#1
;
; INSTRUCTION ENABLE
;
IEN:            EQU     B#0
IENDIS:         EQU     B#1
;
;
; CONDITIONAL CODE MULTIPLEXER (DATA MONITOR)
;
NOACK:          EQU     Q#0
COUT:           EQU     Q#1
PASS:           EQU     Q#7
;
```

```
;
; Certain of the EQU groupings are labeled by [i].  The following DEF statements
; are documented by referring to the group of mnemonics which may be substituted
; in the various variable fields.  The reference is via [i].  The DEF statements
; are also documented by providing a comment on the default value given.
;
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; Am2901-BASED CPU
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
AM2901:  DEF 19X, 3VQ#1, 3VQ#0, 3VQ#1, 2VB#00, 4VH#0, 4VH#0, 26X
; DEFAULTS         AB     ADD.01 NOP.01  NOCin  R0     R0
;                  [1]    [2]    [3]     [19]   [R]    [R]
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; Am29203/2903 TWO ADDRESS OPERATION - NO EXPANDED MEMORY
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
AM29203: DEF 19X, 3VQ#0, 4VH#F, 4VH#C, 2VB#00, 4VH#0, 4VH#0, 1VB#0, 1VB#0, 22X
; DEFAULTS         RAMAB  OR     YBUS   NOCin  R0     R0     IEN    OEY.EN
;                  [9]    [13]   [11]   [19]   [R]    [R]
;                                [14]
;
AM2903:  DEF 19X, 3VQ#0, 4VH#F, 4VH#C, 2VB#00, 4VH#0, 4VH#0, 1VB#0, 1VB#0, 22X
;  DEFAULTS        RAMAB  OR     YBUS   NOCin  R0     R0     IEN    OEY.EN
;                  [9]    [10]   [11]   [19]   [R]    [R]
;                                [12]
;
AM2910:   DEF   4VH#E, 3VX, 12V$X, 45X
; DEFAULTS      CONT    #    #
;               [20]
;
;
; note: This file is written such that the AM2910, AM29203, and AM2904 DEF
; statements are overlayed in the .SRC file.  The user would add additional DEF
; definitions for his/her additional fields in the microword.  Or, the user can
; delete all DEF statements and begin anew.  The intent of this file and all
; other MASTER files is to help reduce the effort required in using AMDASM by
; reducing the effort required to create a customized .DEF file.
;
```

```
;
;
; ADDITIONAL EXAMPLE DEF STATEMENTS
;
;
AM2904:    DEF   42X,     12VQ#2001, 1VB#1,  1VB#0, 4VX,  1VB#1, 3X
; DEFAULTS                LOAD.MSR   OECTDIS OEYEN  X     SE.DIS
;
SHIFT:     DEF   56X,  4VX, 1B#0, 3X
;                      SHIFT SE.EN
TEST:      DEF   42X, 12VQ#7777, 1VB#0, 9X
;                     DISABLED   OECTEN
STATUS:    DEF   42X, 12VQ#2001, B#1, 1VB#0, 4X, B#1, 3X
;                     LOAD.MSR  NO CT OEYEN     SE.DIS
;
; *******************************************************************
; ADDED STATEMENTS FOR DATA MONITOR PROBLEM - USES Am2903 sans .03 **
; ( This is for the EDSYS29 class - other users should delete )
; *******************************************************************
;
NOP2903:   DEF   19X, Q#0, H#F, H#C, B#00, H#0, H#0, B#0, B#1, 22X
;
CTRL:      DEF   61X, 1VB#0, 1VB#0, 1VB#0
;                     DATAin DATAout MEMORY MAP SELECT
;                     CTRL   CTRL    FIRST QUADRANT
END


TOTAL PHASE 1 ERRORS =   0
```

```
;
;
; SAMPLE CODE FOR THE Am29203 - FOR USE IN ED2900A AND ED2900B SEMINARS
; =====================================================================
;
; This file was originally created for the Am2903.  There are a few
; subtle changes necessary for proper operation of the Am29203 due
; to the bidirectional DA port.  RAMAB CANNOT be used as the source for
; INCRR - RAMAQ or DAQ is required, for example.  The original draft
; of the file did not catch these changes - although the DEF file was
; properly flagged.  For users who are switching from one CPU to the
; other - be sure to proof your code for the changes shown in the
; Am29203 function table (normal functions, see the data book).
; The major problems will be encountered in selecting the proper source
; field to work with a given function field.  The don't care source
; equates must also be given careful consideration.
;
;
; This file was created for study.
; It does not contain a complete microword.
;
; Anyone finding errors in this file is requested to send a marked listing
; or portion thereof to: AMD CUSTOMER EDUCATION CENTER
;                        490-A LAKESIDE DRIVE
;                        PO BOX #453 MS #71
;                        SUNNYVALE, CA 94086
;
;
; Advanced Micro Devices reserves the right to make changes in its product
; without notice in order to improve design or performance characteristics.
; The company assumes no responsibility for the use of any circuits or
; programs described herein.
;
;
```

```
     ;
     ;
     ;
     ;
     ; ------------------------------------------------------------------
     ; EDITED JAN 12, 1982
     ;
     ; ADDED EQUATES - CONDITIONAL CODE MULTIPLEXER
     ;
     ZERO:          EQU     Q#0      ; Z STATUS LINE TO MUX
     ;COUT:         EQU     Q#1      ; ALREADY DEFINED (note that this is a comment)
0000 ;PASS:         EQU     Q#7      ; ALREADY DEFINED
0002 OVR:           EQU     Q#2      ; OVERFLOW STATUS LINE TO MUX
     NOVR:          EQU     Q#5      ; INVERTED OVERFLOW STATUS LINE TO MUX
     ;
     ;
     ; * * * * *
     ; SAMPLE Am29203 OPERATIONS FROM THE ED2900A CLASS NOTES
     ;                               THE 2900 FAMILY STUDY GUIDE
     ;                               THE ED2900B CLASS NOTES
     ; * * * * *
     ;----------------------------------------------------------------------
     ; 15. DA + DB --> Yi
0005 ;----------------------------------------------------------------------
0000        AM2910 & AM29203 DADB, ADD, YBUS, NOCARRY
     ;
     ;----------------------------------------------------------------------
     ; 16. RA + RB --> RC (ANY THREE REGISTERS)
     ;----------------------------------------------------------------------
0001        AM2910 & AM29203 , ADD, RAM, NOCARRY, R0, R1    ; ADD THIRD REG FIELD
     ;
     ;----------------------------------------------------------------------
     ; 18.  INCREMENT R15 AND OUTPUT ITS ORIGINAL VALUE
     ;----------------------------------------------------------------------
0002        AM2910 & AM29203 RAMAQ, INCRR, RAM, CARRY, R15, R15
     ;
     ; THE AM2903 WOULD USE RAMAB (000) WHILE THE AM29203 REQUIRES
     ; RAMAQ (010) BE USED WITH THE INCRR FUNCTION.  SEE THE DATA SHEET.
     ;
     ;
     ;
     ;---------------------------------------------------------------------------
     ; 17.  FIRMWARE BYTE SWAP
     ;---------------------------------------------------------------------------
0003        AM2910 LDCT,, H#2 & AM29203 , ADD, RAMUPL, IC, R15, R15 & SHIFT SDROT
0004 LA:    AM2910 RPCT,, LA  & AM29203 , ADD, RAMUPL, IC, R15, R15 & SHIFT SDROT
     ;
     ;----------------------------------------------------------------------------
     ; HARDWARE-ASSISTED BYTE SWAP
     ;---------------------------------------------------------------------------
0005        AM2910 & AM29203 DAQ, INCRR, RAM, NOCARRY, , R15
     ; OR...
0006        AM2910 & AM29203 RAMAB,     , RAM,       , , R15, , OEYDIS
     ;
     ;     **DAQ MUST BE CODED AS 101**
     ;---------------------------------------------------------------------------
     ; 19. DA --> Q
     ;---------------------------------------------------------------------------
0007        AM2910 & AM29203 DAQ, INCRR, LOADQ, NOCARRY
     ;
     ;---------------------------------------------------------------------------
     ; 20. OUTPUT R2 AND PERFORM 2*(R2+1) --> R2 IN ONE MICROCYCLE
     ;---------------------------------------------------------------------------
0008        AM2910 & AM29203 RAMAQ, INCRR, RAMUPL, CARRY, R2, R2 & SHIFT SURZQZ
     ; OR...
0009        AM2910 & AM29203 RAMAQ, INCRR, RAMUPA, CARRY, R2, R2 & SHIFT SURZQZ
     ;
     ;---------------------------------------------------------------------------
     ; 21. UNSIGNED 16 BIT MULTIPLY (R1*R2)
     ;---------------------------------------------------------------------------
000A        AM2910 LDCT ,, H#F & AM29203 RAMAQ, INCRR, LOADQ, NOCARRY, R2
000B LB:    AM2910 RPCT ,, LB  & AM29203      , SPECL, MULT,  NOCARRY, R1, R0
     ;
     ;---------------------------------------------------------------------------
     ; 22. TWO'S COMPLEMENT 16 BIT MULTIPLY (R1*R2)
     ;---------------------------------------------------------------------------
000C        AM2910 LDCT ,, H#E & AM29203 RAMAQ, INCRR, LOADQ,   NOCARRY, R2
000D LC:    AM2910 RPCT ,, LC  & AM29203      , SPECL, TWOMULT, NOCARRY, R1, R0
     /&             SHIFT DDOR
000E        AM2910             & AM29203      , SPECL, TWOLAST, Z,       R1, R0
     /&             SHIFT DDOR
```

```
      ;
      ;----------------------------------------------------------------------------
      ; 23. PERFORM A DOUBLE PRECISION DOWN SHIFT USING R2 AND Q
      ;----------------------------------------------------------------------------
000F        AM2910 & AM29203 , INCRS, RAMQDL, , , R2 & SHIFT DDZR
      ;
      ;----------------------------------------------------------------------------
      ; 24. PERFORM 4*R2 --> Q IN ONE MICROCYCLE
      ;----------------------------------------------------------------------------
0010        AM2910 & AM29203 , ADD, RAMQUPA, NOCARRY, R2, R2 & SHIFT SURZQZ
0011        AM2910 & AM29203 , INCRS, LOADQ, NOCARRY,    , R2
      ;
      ; *************** REQUIRES TWO MICROCYCLES! ********************************
```

```
          ;
          ;
          ;-----------------------------------------------------------------------------
          ; 27. SINGLE LENGTH NORMALIZE OF R6 (16 BIT ALU) - AT END, R4 HAS EXPONENT
          ;     16-BIT ALU
          ;-----------------------------------------------------------------------------
0012 NORMR0: AM2910                  & AM29203 RAMAQ, INCRR, LOADQ, NOCARRY, R6
0013         AM2910 CJP, ZERO, ABORT & AM29203      , SPECL, SLN, , , IENDIS
0014         AM2910 CJP, COUT, LBLA  & AM29203 RAMAQ, INCRR, RAM,   NOCARRY, R7, R4
0015         AM2910 CJP, OVR,  LBLA  & AM29203      , SPECL, SLN,   CARRY,   R4, R4
0016 LBL4:   AM2910 CJP, NOVR, LBL4  & AM29203      , SPECL, SLN,   CARRY,   R4, R4
0017         AM2910                  & AM29203      , INCRS, QD,    NOCARRY
0018         AM2910                  & AM29203      , SPECL, DECRMNT, CARRY,   , R4
0019         AM2910                  & AM29203 RAMAQ, INCRS, RAM,   NOCARRY,     R6
001A LBLA:   FF 64X  ; PLACEHOLDER
001B ABORT:  FF 64X  ; PLACEHOLDER
          ;
          ; load R6 into Q register
          ; test for Zi = 1 <==> Qi = 0, all i; execute SLN with Ien disabled to set up
          ;    special status pins
          ; test Cout = 1 <==> Q3 exor Q2 on MSS = 1; already normalized; set-up exponent
          ;    register R4 with existing exponent of number in R6 (or zero it out)
          ; test Ovr = 1 <==> Q2 exor Q1 on MSS = 1; normalized after this step; execute
          ;    SLN with Ien enabled; SLN shifts Q AND increments the exponent register
          ; test for no Ovr; loop until overflow occurs (number is normalized); SLN
          ; recover Q since status lags behind true state (ED2900A/ED2900B/C seminars)
          ;    by down shifting Q register
          ; decrement the exponent register by -1 (NOTE: unique to Am29203)
          ; put <Qreg> -> <R6>
          ;
          ; REFERENCE: Am2900 FAMILY STUDY GUIDE AND TEACHER'S MANUAL; CEC.PUB-29-3 ($18.00)
```

```
;------------------------------------------------------------------------
; 28. DOUBLE LENGTH NORMALIZE OF R6.R7 (16 BIT ALU)
;------------------------------------------------------------------------
;- NOT CODED YET --------  1/20/82
```

```
        ;
        ;----------------------------------------------------------------------
        ; ADD R1 + R2 -> R2
        ;----------------------------------------------------------------------
001C            AM2910 & AM29203 , ADD, RAM, , R1, R2
        ;
        ;----------------------------------------------------------------------
        ; NAND DA, DB -> YBUS
        ;----------------------------------------------------------------------
001D            AM2910 & AM29203 DADB, NAND, YBUS               ; YBUS REPEATED FOR CLARITY
        ;
        ;----------------------------------------------------------------------
        ; SIGN EXTEND R2
        ;----------------------------------------------------------------------
001E            AM2910 & AM29203 , SPECL, SIGNEXT, NOCARRY, , R2
        ;
        ; NOTE: ADD 1-BIT FIELD FOR I5 FOR RAMEXT
        ;       FIRST TWO SLICES RECEIVE RAMEXT, LAST TWO SLICES RECEIVE SIGNEXT
        ;
        ;----------------------------------------------------------------------
        ; SIGN MAGNITUDE CONVERSION OF R8 INTO TWO'S COMPLEMENT
        ;----------------------------------------------------------------------
001F            AM2910 & AM29203 , SPECL, SGNTWO, Z, , R2
        ;
        ;----------------------------------------------------------------------
        ; INCREMENT R2 BY 2 ( R2 <- R2 + 2 )
        ;----------------------------------------------------------------------
0020            AM2910 & AM29203  , SPECL, INCRMNT, CARRY, , R2
        ;
        ;----------------------------------------------------------------------
        ; BCD ADD R3 + R4 -> R3
        ;----------------------------------------------------------------------
0021            AM2910 & AM29203 , SPECL, BCD.ADD, NOCARRY, R4, R3
        ;
        ;----------------------------------------------------------------------
        ; DECREMENT R2 BY 2 ( R2 <- R2 - 2 )
        ;----------------------------------------------------------------------
0022            AM2910 & AM29203 , SPECL, DECRMNT, NOCARRY, , R2
        ;
        ;----------------------------------------------------------------------
        ;
        END
```

```
0000 1110XXXXXXXXXXXX XXX1010011110000 0000000000XXXXXX XXXXXXXXXXXXXXXX
0001 1110XXXXXXXXXXXX XXX0000011010000 0000000100XXXXXX XXXXXXXXXXXXXXXX
0002 1110XXXXXXXXXXXX XXX0100110010001 1111111100XXXXXX XXXXXXXXXXXXXXXX
0003 1100XXX000000000 0100000011100110 1111111100XXXXXX XXXXXXXX10100XXX
0004 1001XXX000000000 1000000011100110 1111111100XXXXXX XXXXXXXX10100XXX
0005 1110XXXXXXXXXXXX XXX1100110010000 0000111100XXXXXX XXXXXXXXXXXXXXXX
0006 1110XXXXXXXXXXXX XXX0001111010000 0000111101XXXXXX XXXXXXXXXXXXXXXX
0007 1110XXXXXXXXXXXX XXX1100110011000 0000000000XXXXXX XXXXXXXXXXXXXXXX
0008 1110XXXXXXXXXXXX XXX0100110100101 0010001000XXXXXX XXXXXXXX00100XXX
0009 1110XXXXXXXXXXXX XXX0100110100001 0010001000XXXXXX XXXXXXXX00100XXX
000A 1100XXX000000001 1110100110011000 0010000000XXXXXX XXXXXXXXXXXXXXXX
000B 1001XXX000000001 0110000000000000 0001000000XXXXXX XXXXXXXXXXXXXXXX
000C 1100XXX000000001 1100100110011000 0010000000XXXXXX XXXXXXXXXXXXXXXX
000D 1001XXX000000001 1010000000001000 0001000000XXXXXX XXXXXXXX00110XXX
000E 1110XXXXXXXXXXXX XXX0000000011011 0001000000XXXXXX XXXXXXXX00110XXX
000F 1110XXXXXXXXXXXX XXX0000100001100 0000001000XXXXXX XXXXXXXX01100XXX
0010 1110XXXXXXXXXXXX XXX0000011101000 0010001000XXXXXX XXXXXXXX00100XXX
0011 1110XXXXXXXXXXXX XXX0000100011000 0000001000XXXXXX XXXXXXXXXXXXXXXX
0012 1110XXXXXXXXXXXX XXX0100110011000 0110000000XXXXXX XXXXXXXXXXXXXXXX
0013 0011000000000011 0110000000100000 0000000010XXXXXX XXXXXXXXXXXXXXXX
0014 0011001000000011 0100100110010000 0111010000XXXXXX XXXXXXXXXXXXXXXX
0015 0011010000000011 0100000000100001 0100010000XXXXXX XXXXXXXXXXXXXXXX
0016 0011101000000010 1100000000100001 0100010000XXXXXX XXXXXXXXXXXXXXXX
0017 1110XXXXXXXXXXXX XXX0000100010100 0000000000XXXXXX XXXXXXXXXXXXXXXX
0018 1110XXXXXXXXXXXX XXX0000000001101 0000010000XXXXXX XXXXXXXXXXXXXXXX
0019 1110XXXXXXXXXXXX XXX0100100010000 0110000000XXXXXX XXXXXXXXXXXXXXXX
001A XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
001B XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
001C 1110XXXXXXXXXXXX XXX0000011010000 0001001000XXXXXX XXXXXXXXXXXXXXXX
001D 1110XXXXXXXXXXXX XXX1011110110000 0000000000XXXXXX XXXXXXXXXXXXXXXX
001E 1110XXXXXXXXXXXX XXX0000000111000 0000001000XXXXXX XXXXXXXXXXXXXXXX
001F 1110XXXXXXXXXXXX XXX0000000010111 0000001000XXXXXX XXXXXXXXXXXXXXXX
0020 1110XXXXXXXXXXXX XXX0000000010001 0000001000XXXXXX XXXXXXXXXXXXXXXX
0021 1110XXXXXXXXXXXX XXX0000000101100 0100001100XXXXXX XXXXXXXXXXXXXXXX
0022 1110XXXXXXXXXXXX XXX0000000001100 0000001000XXXXXX XXXXXXXXXXXXXXXX
```

```
 SYMBOLS

  AB        0001
  ABORT     001B
  ABUS      0004
  ACONST    0003
  ADCD      0006
  ADCI      0002
  ADD       0003
  ADD.01    0000
  ADD.03    0003
  AINALU    0000
  AIS7051   0001
  AIS7052   0002
  AND       000C
  AND.01    0004
  AND.03    000C
  AQ        0000
  BBUS      0003
  BCD.ADD   000B
  BCD.BIN   0001
  BCD.DIV2  0007
  BCD.SUBR  000F
  BCD.SUBS  000D
  BCLRM     000A
  BIN.BCD   0009
  BINALU    0000
  BIS7051   0001
  BIS7052   0002
  BSETM     000B
  CARRY     0001
  CBUS      0003
  CHLD      001E
  CIN2903   0000
  CIS7051   0001
  CIS7052   0002
  CJP       0003
  CJPP      000B
  CJS       0001
  CJV       0006
  CLA       0000
  CLB       0001
  CLC       0005
  CLD       0007
  CLE       0003
  CLF       0002
  CLG       0006
  CLH       0004
  CLRIN     0001
  CLRM      000C
  CLRMB     0002
  CLRMR     0003
  CLRVC     0004
  CONT      000E
  COUT      0001
  CRTN      000A
  DA        0005
  DADB      0005
 DAQ       0006
 DARAMB    0004
 DDINIOVR  000E
 DDMCR     0004
 DDOR      0003
 DDROT     000F
 DDROTC    000C
 DDROTMC   000D
 DDZR      0006
 DDZRQMC   0007
 DECRMNT   0003
 DISIN     000D
 DIVFRST   000A
 DIVFRST.  000A
 DIVIDE    000C
 DIVIDE.0  000C
 DIVLAST   000E
 DIVLAST.  000E
 DLN       000A
 DLN.03    000A
 DLN.RECO  0005
 DQ        0006
 DZ        0007
 ENCT      0007
 ENIN      000F
 EXNOR     000A
 EXNOR.01  0007
 EXNOR.03  000A
 EXOR      000B
 EXOR.01   0006
 EXOR.03   000B
 FD        0003
 FIRST.25  0001
 FPC       0001
 FPC.32    0004
 FPD       0005
 FPLR      0008
 FPLR.32   000A
 FPR       0006
 FPR.32    0009
 FR        0002
 FR.32     0008
 FRD       0004
 FRDR      0009
 FSD       0007
 HALT      0000
 HIGH      0000
 HIGH.03   0000
 IC        0002
 IEN       0000
 IENDIS    0001
 INCRMNT   0004
 INCRMNT.  0004
 INCRNON   0007
 INCRNON.  0007
 INCRR     0006
 INCRR.03  0006
 INCRS     0004
```

```
INCRS.03 0004
INCRSNON 0005
INCSNON. 0005
INITIALI 0000
JMAP     0002
JMPD     0011
JMPD.32  0005
JMPR     0010
JMPR.32  000B
JMPZ     0012
JP       000F
JPPD     0014
JPPR     0015
JPPR.32  000C
JPRD     0013
JRP      0007
JSBD     0017
JSBR     0016
JSBR.32  000D
JSBZ     0018
JSPD     001A
JSPR     001B
JSPR.32  000E
JSRD     0019
JSRP     0005
JZ       0000
LA       0004
LAST.25  0000
LB       000B
LBL4     0016
LBLA     001A
LC       000D
LDAD     0005
LDCT     000C
LDINVRTM 0601
LDINVRTU 063E
LDM      000E
LDSTA    0009
LDWC     0006
LOAD.BOT 0400
LOAD.INV 0600
LOAD.MSR 0401
LOAD.USR 043E
LOADQ    0006
LOOP     000D
LOW      0008
LOW.03   0008
MCLR     0000
MULT     0000
MULT.03  0000
MULTIBCD 0001
MULTIBIN 0009
NAND     000E
NAND.03  000E
NO.INIT  0001
NOACK    0000
NOCARRY  0000
NOHALT   0000
```

```
 NONQREG   0000
 NOP.01    0001
 NOR       000D
 NOR.03    000D
 NORMR0    0012
 NOTREADY  0001
 NOTRS     0009
 NOTRS.01  0005
 NOTRS.03  0009
 NOVR      0005
 NOWAITRQ  0001
 OECTDIS   0001
 OECTEN    0000
 OEYDIS    0001
 OEYEN     0000
 ONELEVEL  0000
 OR        000F
 OR.01     0003
 OR.03     000F
 OVR       0002
 PASS      0007
 PHLD      000F
 PLDR      000A
 PLDR.32   000F
 POPP      000E
 POPS      000D
 POPS.32   0003
 PRST      0000
 PRST.32   0000
 PSHD      000C
 PSHD.32   0002
 PSHP      000B
 PSHP.32   0006
 PSUS      001F
 PSUS.32   0001
 PUSH      0004
 QD        0005
 QREG.01   0000
 QREGSEL   0001
 QUP       000D
 R0        0000
 R1        0001
 R10       000A
 R11       000B
 R12       000C
 R13       000D
 R14       000E
 R15       000F
 R2        0002
 R3        0003
 R4        0004
 R5        0005
 R6        0006
 R7        0007
 R8        0008
 R9        0009
 RAM       0004
 RAMA.01   0002
```

```
  RAMAB     0000
  RAMADB    0001
  RAMAQ     0002
  RAMD.01   0005
  RAMDA     0000
  RAMDL     0001
  RAMEXT    000F
  RAMF.01   0003
  RAMQ      0007
  RAMQD.01  0004
  RAMQDA    0002
  RAMQDL    0003
  RAMQU.01  0006
  RAMQUPA   000A
  RAMQUPL   000B
  RAMU.01   0007
  RAMUPA    0008
  RAMUPL    0009
  RDAC      0003
  RDCR      0001
  RDM       0007
  RDSTA     0006
  RDVC      0005
  RDWC      0002
  READY     0000
  REIN      0004
  RESRVD.1  0006
  RESRVD.2  0007
  RFCT      0008
  RPCT      0009
  RTS       001C
  RTS.32    0007
  RTSD      001D
  RUN       0000
  SDIC      000B
  SDOROQ    0001
  SDROT     000A
  SDROTC    0009
  SDROTMC   0008
  SDZRZQ    0000
  SE.DIS    0001
  SE.EN     0000
  SET.MSR   0041
  SET.USR   007E
  SETM      0008
  SGNTWO    0005
  SGNTWO.0  0005
  SIGNEXT   000E
  SINGLSTP  0000
  SLN       0008
  SLN.03    0008
  SLN.RECO  0002
  SPECIL.2  0008
  SPECL     0000
  SPECL.03  0000
  SUBR      0001
  SUBR.01   0001
  SUBR.03   0001
  SUBS      0002
  SUBS.01   0002
  SUBS.03   0002
  SURZQZ    0002
  SWAP.REG  0080
  TEST.IC   0EBF
  TEST.IOV  0DBF
  TESTMC    0ABF
  TESTMN    0BBF
  TESTMOVR  09BF
  TESTMZ    093F
  TWB       000F
  TWOLAST   0006
  TWOLAST.  0006
  TWOMULT   0002
  TWOMULT.  0002
  WAITREQ   0000
  WC1D      0004
  WC1I      0000
  WCCD      0005
  WCCI      0001
  WCOD      0007
  WCOI      0003
  WRCR      0000
  YBUS      000C
  Z         0003
  ZA        0004
  ZB        0003
  ZERO      0000
  ZQ        0002

TOTAL PHASE 2 ERRORS =    0
```