This paper covers the approach used to create a master definition file for the Am2900 family RALUs (Am2901, Am2903, and the new Am29203) and related CPU-oriented parts. The created file, AM29CPU.DEF, and its associated source file, AM29203.SRC, were assembled using the beta-site AmSYS29/10, as part of the testing procedure.

Both of the files are included for reference. The files are available from the AMD Customer Education Center.
CREATING A DEFINITION FILE FOR THE Am29203 4-BIT SLICE RALU

by

Donnamaie E. White

An integral part of structured microprogramming is documentation. The largest piece of this is a properly created definition file. For the AmSYS29/10 development system, this is an AMDASM ".DEF" file.

The ".DEF" file is the file that assembles a symbol table for the use of the phase 2 assembly. The phase 2 assembly is the actual assembly of the source code statements. The symbol table is created by a phase 1 assembly of a user-created file of equates and format definitions.

A symbol table entry in microprogramming consists of a mnemonic and its paired bit pattern. The mnemonic, which is user generated, should be selected such that it is a key to what the particular field will cause the hardware to do.

As an example of the process involved, this paper will develop the .DEF file for the Am29203 4-bit slice RALU. A definition file already exists for the Am2903 and the Am2901, predecessors to the newest bit-slice RALU. These existing mnemonics will be taken into consideration as new ones are assigned to the Am29203 instruction set.
The following pages present part of the existing AM29FAML.DEF file, an updated version of the AM2903.DEF and the original AM2900.LIB files. The purpose of these large files is to provide a pre-created set of instructions for all of the Am2900 family parts which might appear in a CPU design. A separate master file exists for Am29116-based controller designs.

A designer or programmer can load in a master file, edit out parts as necessary, add definitions for fields unique to the current design, create definition (format description) statements and size the microword, all with a minimum of time and effort. The master files are all preassembled, working .DEF files, which is of further assistance. The master files serve as reference files for the novice, providing a guide to error-free definitions. And finally, the master files are well-structured and commented, which is useful to the final documentation effort.

Given an existing master file, the task is to add the new RALU, while providing a minimum of impact on the existing mnemonics. The assumption is that several source files (".SRC") exist which reference the existing master .DEF file. The approach used in the AM29FAML.DEF file for conflicting mnemonics is to tag the mnemonic on the less powerful part with a part number.
AMDGOS/29 AMDASM MICRO ASSEMBLER, V1.4
THE AM2900 FAMILY MNEMONICS

;************************************************************
;
;************************************************************
;THE Am2900 FAMILY MNEMONICS
;************************************************************
;
;
WORD 64
;
13 DECEMBER 1976 JRM
;UPDATED SEPTEMBER 28, 1977
;UPDATED APRIL 16, 1981 DEW

Am2901 INSTRUCTION SET

Am2901 SOURCE OPERANDS (R S)

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)

*** TO USE: DELETE THE .01 AND DELETE THE Am2903/29203 INSTRUCTION SET ***

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

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
; Am2903 INSTRUCTION SET
;
; ALU SOURCE OPERANDS (EA, I0, OEB)
; 16 REGISTER - TWO ADDRESS VERSION
;
; 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)
;
SPECL: EQU H#0 ; I0 MUST BE LOW
HIGH: EQU H#0 ; I0 MUST BE HIGH (Q REGISTER SELECT)
SUBR: EQU H#1 ; F = S - R - 1 + Cin
SUBS: EQU H#2 ; F = R - S - 1 + Cin
ADD: EQU H#3 ; F = R + S + Cin
INCRS: EQU H#4 ; F = S + Cin
INCSNON: EQU H#5 ; F = NOT S + Cin
INCRR: EQU H#6 ; F = R + Cin
INCRNON: EQU H#7 ; F = NOT R + Cin
LOW: EQU H#8 ; F = LOW
NOTRS: EQU H#9 ; F = NOT R AND S
EXNOR: EQU H#a ; F = R EXNOR S
EXOR: EQU H#b ; F = R EXOR S
AND: EQU H#c ; F = R AND S
NOR: EQU H#d ; F = R NOR S
NAND: EQU H#e ; F = R NAND S
OR: EQU H#f ; F = R OR S
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; ALU DESTINATION CONTROL ( I8 - I7 - I6 - I5)
; NORMAL FUNCTIONS
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

RAMDA: EQU HB0 ; F TO RAM, ARITHMETIC DOWN SHIFT
RAMDL: EQU HB1 ; F TO RAM, LOGICAL DOWN SHIFT
RAMQDA: EQU HB2 ; DOUBLE PRECISION ARITHMETIC DOWN SHIFT
RAMQDL: EQU HB3 ; DOUBLE PRECISION LOGICAL DOWN SHIFT
RAM: EQU HB4 ; F TO RAM WITH PARITY
QD: EQU HB5 ; F TO Y, DOWN SHIFT Q
LOADQ: EQU HB6 ; F TO Q WITH PARITY
RAMQ: EQU HB7 ; F TO RAM AND Q WITH PARITY
RAMUPA: EQU HB8 ; F TO RAM, ARITHMETIC UP SHIFT
RAMUPL: EQU HB9 ; F TO RAM, LOGICAL UP SHIFT
RAMQUPA:EQU HB10 ; DOUBLE PRECISION ARITHMETIC UP SHIFT
RAMQUPL:EQU HB11 ; DOUBLE PRECISION LOGICAL UP SHIFT
YBUS: EQU HB12 ; F TO Y ONLY
QUP: EQU HB13 ; F TO Y, UP SHIFT Q
SIGNEXT:EQU HB14 ; S100 TO Y1
RAMEXT: EQU HB15 ; F TO Y, SIGN EXTEND LEAST SIG. BYTE
SPECIAL FUNCTIONS (I8-I7-I6-I5)

Am2903 FUNCTIONS ONLY

MULT: EQU H#0 ; UNSIGNED MULTIPLY
TWMULT: EQU H#2 ; TWO'S COMPLEMENT MULTIPLY
TWOLAST: EQU H#6 ; TWO'S COMPLEMENT MULTIPLY LAST STEP
INCMNT: EQU H#4 ; INCREMENT BY 1 + Cin
SGNTWO: EQU H#5 ; SIGN MAGNITUDE-TWO'S COMPL CONVERSION
SLN: EQU H#8 ; SINGLE LENGTH NORMALIZE
DLN: EQU H#A ; DOUBLE LENGTH NORMALIZE
DVFIRST: 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

Am29203 ADDITIONS FORTHCOMING
; ** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; TWO ADDRESS OPERATION - NO EXPANDED MEMORY
; ** * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
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
;
AM2910:  DEF 4VH#E, 3VX, 12V$X, 45X
; DEFAULTS CONT # #
;
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
;
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 CTRL FIRST QUADRANT
;
;
END

TOTAL PHASE 1 ERRORS = 0
The Am2901 RALU uses ADD in its function definitions as does the Am2903. When the AM29FAML.DEF file was created, the Am2901 definition appeared as:

```
ADD.01: EQU Q#0
```

and the Am2903 definition appeared as:

```
ADD: EQU H#3
```

Since the Am29203 also has an add function, and because of certain other factors discussed later, the Am29203 will appear as:

```
ADD: EQU H#3
```

and the Am2903 will be changed to be:

```
ADD.03: EQU H#3
```

The tagging of the less powerful or older parts means that their definitions can be left unchanged when the more powerful, newer parts are the ones used in the design. The tagging approach was also chosen because of the relative ease with which it can be edited out when a tagged part is the one of interest.
By "killing" or by tagging-out the Am29203 definitions, for example, the Am2901 equates can be un-tagged and still remain unique. Removing or adding tags can be done using the normal AmSYS29/10 editor, ED, or with WORDMASTER, a video editor which contains ED commands as a subset. (WORDMASTER is not available from AMC nor is it currently supported by AMC.)

The tagging method was chosen for the three RALUs, Am2901, Am2903, and Am29203. It was also used for the Am2930-Am2932 program control units and the Am2940-Am2942 DMA addressing units to resolve their mnemonic conflicts. The Am2910 and Am29811 sequencer and next-address control units have no conflict. A mere redefinition of the H#F instruction is all that is required to allow the file to service both cases.

A definition file can have several mnemonics for any one bit-pattern. The only overall requirement is uniqueness. AMDASM runs under CP/M (AMDOS 2.1 is a CP/M 2.2 compatible OS) and therefore eight-character names are the limit. More characters can appear in a mnemonic (name) but only the first 8 are deciphered and the first 8 must be unique.
On examining the Am29203 function tables, the differences between the Am2903 and the Am29203 are seen to be slight but significant. There are a number of approaches that can be used to create mnemonic-compatible definitions. The one chosen was selected because of the existing source files which refer to the Am2903 mnemonics, the desire to maintain these and to also allow these same files, with minimal alteration, if any, to refer to the Am29203 mnemonics.

Table 1 in the Am29203 data sheet shows no change in the operand source selection bit patterns. Tables 2A and 2B are different in only three encodings, but these are special. It was decided to not change the operand source equate field from 3 bits (EA, IO, OEB) from the Am2903 (two-address, non-expanded scratchpad memory), but to share the equates.

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

; ALU SOURCE OPERANDS (EA, IO, OEB)
; 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

The function field would not change from four bits (I4, I3, I2, I1) to five bits (I4, I3, I2, I1, I0), but would instead be rearranged to flag the user as to the companion source field requirements. Special-function codes are differentiated and reserved codes are called out for easier insertion when they are eventually defined.

The normal functions then become (using the same mnemonics as used previously for the Am2903, which are now tagged as .03):

; * * 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
  INCRSNAV: 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
  INCRSNAV: 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)
The destination table is the same as before, therefore, the existing destination mnemonics were not tagged and the equates apply to either part.

The last major concern is the special function table, table 4 on the Am29203 data sheet. The Am2903 special functions are a proper subset of those of the Am29203. However, to avoid possible confusion, and since special attention needed to be given to the two special function encodings of I4 through I0 (where one existed for the Am2903), a separate set of equates was created. To anticipate a question, yes, the file could be reduced by deleting the tagged Am2903 special function codes, but it would be less clear as to which special function belonged to which RALU. Clarity was chosen as the objective. The special function table is shown on the following page.
DESTINATION CONTROL - SEE Am2903 DESTINATION (SAME)

Am29203 SPECIAL FUNCTIONS

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 DIVIDS - 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:

MULTBCD: EQU H#1 ; MULTIPRECISION BCD TO BINARY CONVERSION
MULTBIN: EQU H#9 ; MULTIPRECISION BINARY TO BCD CONVERSION
The final step in modifying the master file is to create the definition of the microinstruction format.

AM2903: DEF 19X, 3VQ#0, 4VH#F, 4VH#C, 2VB#00, 4VH#0, 4VH#0, 1VB#0, 1VB#0, 22X
; DEFAULTS X RAMAB OR YBUS NOcin R0 R0 IEN OEY.EN X

already existed. Copying this closely,

AM29203: DEF 19X, 3VQ#0, 4VH#F, 4VH#C, 2VB#00, 4VH#0, 4VH#0, 1VB#0, 1VB#0, 22X
; DEFAULTS X RAMAB OR YBUS NOcin R0 R0 IEN OEY.EN X

The differences between these two is merely the label. For those who breadboarded an Am29203 design using the Am2903, this is not the final physical layout. This is the conceptual microword. AMSCRNM can be used to "fix" the proper physical layout. AMSCRNM translates the human-readable conceptual microword into a user-defined physical microword.

The following pages present the new master CPU file, AM29CPU.DEF, which is 20k bytes unassembled, 27k bytes assembled.

To prove that an existing Am2903 source file would assemble against the new Am29203 symbol table, AM2903.SRC, a sample code file, was assembled against AM29CPU.DEF as it is listed. The results follow the definition file listing.
The following changes were all that were required:
   1. Using substitute (S)
      *25S2903Zc29203Zc
   2. Rename the file AM29203.DEF

Several additions were made:
   1. The Single-length-normalize example was completed
   2. All code after label 28. was added
AMDM/29 AMDASM MICRO ASSEMBLER, V1.4
THE AM2900 FAMILY MNEMONICS

; * * * * * * * * * * * * * *
;
; AM2914 INSTRUCTION SET
;
; * * * * * * * * * * * * * *
;
MCLR: EQU H#0 ; MASTER CLEAR
CLRIN: EQU H#1 ; CLEAR ALL INTERRUPTS
CLRMB: EQU H#2 ; CLEAR INTERRUPTS FROM M-BUS
CLRM: 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
BCLR: EQU H#A ; BIT CLEAR MASK REGISTER
BSETM: EQU H#B ; BIT SET MASK REGISTER
CLR: 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

; 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
AMOS/29 AMDASM MICRO ASSEMBLER, V1.4
THE AM2900 FAMILY MNEMONICS

; Am2932 PROGRAM CONTROL UNIT

; TO USE: DELETE THE .32 FROM AM2932 NMEMONICS 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

; 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
WC0I: 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, IO, OEB)
; 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, IO ALL NOT 0)
; Am2903 ONLY ! ** TO USE, DELETE THE .03 AND TAG OR
; DELETE THE Am29203 FUNCTIONS ** *
;
;
; 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
INCR.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
Am29203 ALU FUNCTIONS - NORMAL MODE (I4, I3, I2, I1, I0 ALL NOT 0)

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
INCRSNNON: 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
INCR: EQU H#6 ; Fi = Ri + Cin
INCRN: 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)
AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4
THE AM2900 FAMILY MNEMONICS

; * * * * * * * * * * * * * * * * * * * * *
;
; ALU DESTINATION CONTROL (I8 - I7 - I6 - I5)
; 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 ; S100 TO Yi
RAMNEXT: EQU H#F ; F TO Y, SIGN EXTEND LEAST SIG. BYTE
; AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4
; THE AM 2900 FAMILY MNEMONICS

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
; SPECIAL FUNCTIONS (I8-I7-I6-I5)
;
;
; 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
;
AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4
THE AM2900 FAMILY MNEMONICS

******************************************************************************
NEW! NEW! 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

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4
THE AM 2900 FAMILY MNEMONICS

; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; DESTINATION CONTROL - SEE Am2903 DESTINATION (SAME)
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; Am29203 SPECIAL FUNCTIONS
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; 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
DECRMT: EQU H#3 ; DECREMENT BY 1 OR 2
INCRMT: 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 DIVIDS - 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.SUBL: EQU H#F ; BCD SUBTRACT Fi = Si - Ri - 1 + Cin [BCD]

; THE FOLLOWING ARE USED WITH SPECIL.2:

MULTBCD: EQU H#1 ; MULTIPRECISION BCD TO BINARY CONVERSION
MULTBIN: 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 Am20705

ONLY THE SOURCE FIELDS CHANGE – EXPANDED
A THIRD ADDRESS FIELD WAS ALSO ADDED (ADDRESS C)

SOURCE OPERANDS
ADDED A ADDRESS BITS (A6-A5-A4)

AINALU: EQU Q#0
AIS7051: EQU Q#1
AIS7052: EQU Q#2
ACONST: EQU Q#3
ABUS: EQU Q#4

ADDED B ADDRESS BITS (B5-B4)

BINALU: EQU B#00
BIS7051: EQU B#01
BIS7052: EQU B#10
BBUS: EQU B#11
AMDS/29 AMDASM MICRO ASSEMBLER, V1.4
THE AM2900 FAMILY Mnemonics

; * * * * * * * * * * * * * * * * *
; THREE ADDRESS OPERATION - THIRD ADDRESS FIELD
; ADDED C ADDRESS BITS (C5-C4)
; * * * * * * * * * * * * * * * * *
;
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
;
; * * * * * * * * * * * * * * * * *
; I0 SOURCE SELECT FIELD REPLACES EA-I0-OEB THREE-BIT FIELD
; * * * * * * * * * * * * * * * * *
;
QREGSEL:EQU B#1 ; SOURCE IS Q REGISTER
NONQREG:EQU B#0 ; SOURCE IS RAMB OR B.BUS
;
MISC.

** MISCELLANEOUS FIELDS **

** REGISTERS **

- 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) **

- CARRY: EQU B#01
- NOCARRY: EQU B#00

** C. I. C. **

- Z: EQU B#11

** IMAGINATIVE! **

- C. I. C. with C is C. I. N.
AMOS/29 AMDASM MICRO ASSEMBLER, V1.4
THE AM2900 FAMILY MNEMONICS

; ; * * * * * * * * * * * * * * * * * * * * * * *
; ; 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
SDMRQ:     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; QO->Mc
SDROTMC:   EQU    H#8    ; ROT.R; R0->Mc; ROT.Q
SDROT:     EQU    H#9    ; ROT.R WITH Mc; ROT.Q
SDIC:      EQU    H#A    ; ROT.R; ROT.Q
SDIC:      EQU    H#B    ; R0->RN; R0->QN
DDROTC:    EQU    H#C    ; Mc->RN; R0->QN; QO->Mc
DDR0TMC:   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)
;
SURZQ:     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

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
AMOS/29 AMDASM MICRO ASSEMBLER, V1.4
THE AM2900 FAMILY MNEMONICS

; 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
TESTMV: EQU 12Q#4677 ; NO STATUS OPERATION
TESTMC: EQU 12Q#5277 ; NO STATUS OPERATION
TESTMN: EQU 12Q#5677 ;
TEST.IOV: EQU 12Q#6677 ;
TEST.IC: EQU 12Q#7277 ;
;
TEST ENABLE
;
OECTEN: EQU B#0
OECTDIS: EQU B#1
;
OUTPUT ENABLE
;
OYEN: EQU B#0
OYDIS: EQU B#1
;
INSTRUCTION ENABLE
;
IEN: EQU B#0
IENDIS: EQU B#1
;
CONDITIONAL CODE MULTIPLEXER (DATA MONITOR)
;
NOACK: EQU Q#0
COUR: EQU Q#1
PASS: EQU Q#7 ;
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 OEEY.EN
;
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 OEEY.EN
;
AM2910: DEF 4VH#E, 3VX, 12V$X, 45X
; DEFAULTS CONT # #
;
AM2904: DEF 42X, 12VQ#2001, 1VB#1, 1VB#0, 4VX, 1VB#1, 3X
; DEFAULTS LOAD.MSR OECTDIS OEEYEN 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 OEEYEN SE.DIS
; ADDED STATEMENTS FOR DATA MONITOR PROBLEM - USES Am2903 sans .03 **
; ************************************************************************************

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
B\>amdasim P2 AM29203 D=AM29CPU W=100

AMDS/29 AMDSM MICRO ASSEMBLER, V1.4

TOTAL PHASE 2 ERRORS = 0

TYPE B: AM29203.P2L

AMDS/29 AMDSM MICRO ASSEMBLER, V1.4

TEST FILE FOR AM29CPU.DEF FILE
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)
PASS: EQU Q#7 ; ALREADY DEFINED
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 --> Y1

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, INCRR, RAM, CARRY, R15, R15

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

19. HARDWARE-ASSISTED BYTE SWAP

0005 AM2910 & AM29203 DARAMB, INCRR, RAM, NOCARRY, R15

OR...

0006 AM2910 & AM29203 RAMAB, , RAM, , , R15, OEYDIS
; 19. DA --> Q

; 20. OUTPUT R2 AND PERFORM 2*(R2+1) --> R2 IN ONE MICROCYCLE

; OR...

; 21. UNSIGNED 16 BIT MULTIPLY (R1*R2)

; 22. TWO'S COMPLEMENT 16 BIT MULTIPLY (R1*R2)

; 23. PERFORM A DOUBLE PRECISION DOWN SHIFT USING R2 AND Q

; 24. PERFORM 4*R2 --> Q IN ONE MICROCYCLE

; ************************** REQUIRES TWO MICROCYCLES! **************************
27. SINGLE LENGTH NORMALIZE OF R6 (16 BIT ALU) - AT END, R4 HAS EXPONENT

16-BIT ALU

0012 NORMR0: AM2910 & AM29203 , INCRR, LOADQ, NOCARRY, R6
0013 AM2910 CJP, ZERO, ABORT & AM29203 , SPECL, SLN, , , IENDIS
0014 AM2910 CJP, COUT, LBLA & AM29203 , 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 Z = 1 => Q = 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)
AMDOS/29 AMDASM MICRO ASSEMBLER, V1.4
TEST FILE FOR AM29CP0.DEF FILE

28. DOUBLE LENGTH NORMALIZE OF R6.R7 (16 BIT ALU)
ADD R1 + R2 → R2

AM2910 & AM29203, ADD, RAM, R1, R2

NAND DA, DB → YBUS

AM2910 & AM29203 DADB, NAND, YBUS; YBUS REPEATED FOR CLARITY

SIGN EXTEND R2

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

AM2910 & AM29203, SPECL, SGNTWO, Z, R2

INCREMENT R2 BY 2 ( R2 ← R2 + 2 )

AM2910 & AM29203, SPECL, INCRMNT, CARRY, R2

BCD ADD R3 + R4 → R3

AM2910 & AM29203, SPECL, BCD.ADD, NOCARRY, R4, R3

DECREMENT R2 BY 2 ( R2 ← R2 - 2 )

AM2910 & AM29203, SPECL, DECRMNT, NOCARRY, R2

END
# Partial Symbol Table

<table>
<thead>
<tr>
<th>SYMBOLS</th>
<th>AB</th>
<th>0001</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABORT</td>
<td>0004</td>
<td></td>
</tr>
<tr>
<td>ACONST</td>
<td>0003</td>
<td></td>
</tr>
<tr>
<td>ADD</td>
<td>0002</td>
<td></td>
</tr>
<tr>
<td>ADCI</td>
<td>0002</td>
<td></td>
</tr>
<tr>
<td>ADD</td>
<td>0003</td>
<td></td>
</tr>
<tr>
<td>ADD:01</td>
<td>0003</td>
<td></td>
</tr>
<tr>
<td>AINALU</td>
<td>0000</td>
<td></td>
</tr>
<tr>
<td>AIS7051</td>
<td>0001</td>
<td></td>
</tr>
<tr>
<td>AIS7052</td>
<td>0002</td>
<td></td>
</tr>
<tr>
<td>AND</td>
<td>0004</td>
<td></td>
</tr>
<tr>
<td>AND:01</td>
<td>0004</td>
<td></td>
</tr>
<tr>
<td>AND:03</td>
<td>0004</td>
<td></td>
</tr>
<tr>
<td>AQ</td>
<td>0003</td>
<td></td>
</tr>
<tr>
<td>BCD ADD</td>
<td>0003</td>
<td></td>
</tr>
<tr>
<td>BCD BIN</td>
<td>0003</td>
<td></td>
</tr>
<tr>
<td>BCD DIV</td>
<td>0007</td>
<td></td>
</tr>
<tr>
<td>BCD SUBB</td>
<td>0002</td>
<td></td>
</tr>
<tr>
<td>BCD SUBS</td>
<td>0002</td>
<td></td>
</tr>
<tr>
<td>BCD CLR M</td>
<td>0009</td>
<td></td>
</tr>
<tr>
<td>BIN BCD</td>
<td>0000</td>
<td></td>
</tr>
<tr>
<td>BIN CLR</td>
<td>0000</td>
<td></td>
</tr>
<tr>
<td>BUS</td>
<td>0003</td>
<td></td>
</tr>
<tr>
<td>CARRY</td>
<td>0001</td>
<td></td>
</tr>
<tr>
<td>CBEU</td>
<td>0003</td>
<td></td>
</tr>
<tr>
<td>CLH</td>
<td>0014</td>
<td></td>
</tr>
<tr>
<td>CIN2903</td>
<td>0000</td>
<td></td>
</tr>
<tr>
<td>CIS17051</td>
<td>0001</td>
<td></td>
</tr>
<tr>
<td>CJPP</td>
<td>0003</td>
<td></td>
</tr>
<tr>
<td>CJSC</td>
<td>0001</td>
<td></td>
</tr>
<tr>
<td>CJV</td>
<td>0006</td>
<td></td>
</tr>
<tr>
<td>CLRIN</td>
<td>0001</td>
<td></td>
</tr>
</tbody>
</table>