```
TITLE    ASSEMBLY LANGUAGE DEFINITION FILE FOR FPGA-CPU
WORD     24                          ; 24 || 16 || 8 bits
WIDTH    72
LINES    50

;*************************************************************************
; INSTRUCTION OPCODE LABELS - 1-Bit prefix + 1-Hex opcode
;*************************************************************************
LL:          EQU    B#0              ; prefix 0 (for all commands except Rd = Ra * Rb)
LH:          EQU    B#1              ; prefix 1 (for Rd = Ra * Rb)
LSTOP:       EQU    4H#F
LJORG:       EQU    4H#0             ; jump to 0
LJZ:         EQU    4H#1             ; jump to A if Rd == 0
LJMP:        EQU    4H#2
LMOVR:       EQU    4H#3             ; Rd = Ra
LADD:        EQU    4H#4
LSUB:        EQU    4H#5
LOR:         EQU    4H#6
LAND:        EQU    4H#7
LLWR:        EQU    4H#8             ; Rd = RAM(Ra.Rb)
LSWR:        EQU    4H#9             ; RAM(Ra.Rb) = Rd
LLWI:        EQU    4H#A             ; Rd = RAM(A)
LSWI:        EQU    4H#B             ; RAM(A) = Rd
LIN:         EQU    4H#C
LOUT:        EQU    4H#D
LMOVI:       EQU    4H#E             ; Rd = DI
LMUL:        EQU    4H#E             ; Rd = Ra * Rb

;*************************************************************************
; INSTRUCTION FIELDS DEFINITIONS
;*************************************************************************
Rd:          SUB    3VB#000          ; destination register
Ra:          SUB    3VB#000          ; first source register
Rb:          SUB    3VB#000          ; second source register
DI8:         SUB    8VH#00; 8-bit immediate
A8:          SUB    8VH#00; 8-bit address
A16:         SUB    16VH#0000        ; 16-bit address
Z1:          EQU    B#0              ; const 0
Z3:          EQU    B#000            ; const 000
Z4:          EQU    4H#0             ; const 0000

;*************************************************************************
; DATA PSEUDO OPS
;*************************************************************************
DB:          DEF    8VH#00; 8-BIT DATA DIRECTIVE
;*************************************************************************
; ASSEMBLY LANGUAGE INSTRUCTIONS
;*************************************************************************
STOP:        DEF    LL,LSTOP,        Z3                  ; STOP
JORG:        DEF    LL,LJORG,        Z3                  ; jump to 0
JZ:          DEF    LL,LJZ, Rd,A8                ; jump to A[7:0] if Rd == 0
JMP:         DEF    LL,LJMP,         Z3,A8               ; jump to A[7:0]
MOVR:        DEF    LL,LMOVR,        Rd,Z1,Ra,Z4         ; Rd = Ra               move register
ADDA:        DEF    LL,LADD,         Rd,Z1,Ra,Z1,Rb      ; Rd = Ra + Rb
SUBA:        DEF    LL,LSUB,         Rd,Z1,Ra,Z1,Rb      ; Rd = Ra - Rb          cannot be 'SUB' due to keyword
OR:          DEF    LL,LOR,          Rd,Z1,Ra,Z1,Rb      ; Rd = Ra # Rb
AND:         DEF    LL,LAND,         Rd,Z1,Ra,Z1,Rb      ; Rd = Ra & Rb
LWR:         DEF    LL,LLWR,         Rd,Z1,Ra,Z1,Rb      ; Rd = RAM(Ra.Rb)       load word register addressing
SWR:         DEF    LL,LSWR,         Rd,Z1,Ra,Z1,Rb      ; RAM(Ra.Rb) = Rd       store word register addressing
LWI:         DEF    LL,LLWI,         Rd,A16              ; Rd = RAM(A[15:0])     load word immediate addressing
SWI:         DEF    LL,LSWI,         Rd,A16              ; RAM(A[15:0]) = Rd   store word immediate addressing
IN:          DEF    LL,LIN,          Rd,A8               ; Rd = INP(A[7:0])      read from IO
OUT:         DEF    LL,LOUT,         Rd,A8               ; OUT(A[7:0]) = Rd      write to IO
MOVI:        DEF    LL,LMOVI,        Rd,DI8              ; Rd = DI               move immediate
MUL:         DEF    LH,LMUL,         Rd,Z1,Ra,Z1,Rb      ; Rd = Ra * Rb
```

```
;**********************************************************************
; CPU's REGISTERS CODES DEFINITION
;**********************************************************************
R0:                 EQU     B#000
R1:                 EQU     B#001
R2:                 EQU     B#010
R3:                 EQU     B#011
R4:                 EQU     B#100
R5:                 EQU     B#101
R6:                 EQU     B#110
R7:                 EQU     B#111

END
```