**Documentation file:**

**Assembler.c:**

typedef struct{

char labelName[MAX\_LINE\_LENGTH];

MAX\_LINE\_LENGTH

int lineNumber;

} LabelInfo;

* This struct will help us store labels on the first loop of the file, we will
* Use it to represent a legal label by its name, and line (as in the imemin.txt file)

int checkLABEL(char\* word)

* Check if we are at a Label Line

char\* createBinaryLinePkoda(char words[MAX\_WORDS\_PER\_LINE][MAX\_LINE\_LENGTH])

* This function is called when we refer to a readed line as command line(not a label line or empty line), it will break the line into serveal analysis for each component and assign its desierd hex value.

void opcode\_to\_hex(const char \*opcode\_name, char \*addr)

* Function to convert the opcode to its right hex value

void processLine(char \*line, char words[MAX\_WORDS\_PER\_LINE][MAX\_LINE\_LENGTH])

* Process line make the line suitable for work : tokenize and remove commas.

void writeLineToFile(const char\* line)

* This function takes a ready hexa line, and copy it to the imemin.txt .

void reg\_to\_hex(const char\* regName, char\* hexValue)

* Convert the register name to its desierd hex value as given in the project.

void int\_to\_hex3(const char\* intString, char\* hexString)

* Function to help converting a immediate value representing a decimal int value to a 3 length 2’s complement hexa value.

void storeLabel(const char\* labelName, int lineNumber)

* Used when found a label line, appending the label and its code line to the label struct.

void int\_to\_hex3\_jal(const char\* Label, char\* hexString)

* Finding the desierd label to jump to and use covert to its hexa value.

**Simulator.c:**

typedef struct LinkedList\_int{

struct LinkedList\_int \*next;

int cycleVal;

}LinkedList\_int;

* Used for saving the irq2 list.

int init(int argc, char const \*argv[])

* Initiallize some files, memory and some global variables

int simClockCycle(void)

* The heard of the code, this function will represent a single clock cycle iteration, will call to other fucntion for most of the functionaltiy.

int byebye(int argc, char const \*argv[])

* the function called prior to the programm ending, will close file, handle memory if needed etc.

int writeToMonitor(void)

* the function that will desgin the monitor file.

int writeToLeds(FILE \*ledsFileName)

* the function that will desgin the led file.

int incrementTimer(void)

* function to icrement the timer and alert if needed.

int readimemin(const char \*meminFileName)

int readdmemin(const char \*dmeminFileName)

int readdiskin(const char \*diskinFileName)

int readirq2in(const char \*irq2inFileName)

* functions to read from files.

uint64\_t hex\_to\_bin64(char hex\_char)

uint32\_t hex\_to\_bin32(char hex\_char)

* converting functions

int update\_traceFile(void)

int update\_hwRegTraceFile(const char\* action, int regNum, uint32\_t data)

int writeDmemout(FILE\* dmemoutFile)

int writeRegout(FILE\* regoutFile)

int writeCycles(FILE\* cyclesFile)

int writeLeds(FILE\* ledFile)

int write7Seg(FILE\* sevenSegmentFile)

int writeDiskOut(FILE\* diskoutFile)

int writeMonitorTxt(FILE\* monitorFile)

int writeMonitorFiles(FILE\* txtMonitorFile, FILE\* binMonitorFile)

* functions design to update/ write the requierd output files .

int\* parseInstruction(uint64\_t inst)

* parsing instruction to 2s complment hexa, used often for imm1 and imm2 loading before the instruction run.

int handleDisk(void)

int hexCharToValue(char c)

uint8\_t hexStringToByte(const char\* hexString)

* other helper functions.

int addCmd(int rd, int rs, int rt, int rm)

int subCmd(int rd, int rs, int rt, int rm)

int macCmd(int rd, int rs, int rt, int rm)

int andCmd(int rd, int rs, int rt, int rm)

int orCmd(int rd, int rs, int rt, int rm)

int xorCmd(int rd, int rs, int rt, int rm)

int sllCmd(int rd, int rs, int rt)

int sraCmd(int rd, int rs, int rt)

int srlCmd(int rd, int rs, int rt)

int beqCmd(int rs, int rt, int rm)

int bneCmd(int rs, int rt, int rm)

int bltCmd(int rs, int rt, int rm)

int bgtCmd(int rs, int rt, int rm)

int bleCmd(int rs, int rt, int rm)

int bgeCmd(int rs, int rt, int rm)

int jalCmd(int rd, int rm)

int lwCmd(int rd, int rs, int rt, int rm)

int swCmd(int rd, int rs, int rt, int rm

int retiCmd(void)

int inCmd(int rd, int rs, int rt)

int outCmd(int rs, int rt, int rm)

* all the opcode operation function based on the project discription, each function is desgin to follow and complete a single type opcode instruction.