Project Requirement Checklist

The following requirements are adapted from the [Project Specification](https://canvas.uw.edu/courses/1331980/pages/project-specification) and the [Supported Instructions & Addressing Modes](https://canvas.uw.edu/courses/1331980/pages/supported-instructions-and-addressing-modes), as found on canvas.

## Code Quality

* Only uses Trap Tasks between 0-15
* Does not use Trap Tasks >= 16
* Uses I/O Subtasks between 50-59
* Does not use any I/O subtasks < 50 or > 59
* Explicitly **DOES NOT** use subtask 60
* Program starts at ***ORG $1000***
* Program & disassembly stays within address **$8999**
* Program DOES NOT reach addresses $9000 - $9800
* These addresses are reserved for Nash
* Uses Nash’s “print” function
* (Prints to console & file, for grading purposes)
* Runs to completion without human interaction

## Output Formatting:

* Disassembler content output is formatted as:

***MemoryLocation***<tab>***OpCode***<tab>***Operand(s)***

* Unrecognized opcodes use the string literal “DATA” as substitute opcode:

***MemoryLocation***<tab>***DATA***<tab>***Operand(s)***

* The program does not crash when unrecognized opcodes are found
* The program continues to print “DATA” lines until a new, recognized opcode is found
* Does NOT use characters outside of:
  + ASCII character set
  + CR
  + LF
* Does:
  + Print a header on startup, including:
    - A double-dash header line to mark file/program start
    - Our Group Name
    - Our individual names
    - The course number
    - The professor’s name
    - A single-dash separator line
  + Print information about Disassembler initiation, including:
    - Start time
    - Target file to open
    - Success/failure of opening the target file
  + Print disassembler content
  + Print a footer on program completion, including:
    - A single-dash separator line
    - The end time
    - The total runtime
    - A statement that the disassembler has completed
    - A double-dash footer line to indicate file/program end

## Functionality:

### Opcodes:

* + Handles all 30 opcodes
    - MOVE
    - MOVEA
    - MOVEM
    - ADD
    - ADDA
    - SUB
    - SUBQ
    - MULS
    - DIVS
    - LEA
    - OR
    - ORI
    - NEG (2)
    - EOR
    - LSR
    - LSL
    - ASR
    - ASL
    - ROL
    - ROR
    - BCLR
    - CMP
    - CMPI
    - BCS
    - BGE
    - BLT
    - BVC
    - BRA
    - JSR (1)
    - RTS (0)
  + Handles all non-supported opcodes as “DATA”
  + Handles at least one invalid opcode (not defined in the motorola spec) as “DATA”

|  |  |  |
| --- | --- | --- |
| Difficulty | Shape | Opcode |
|  | 1110;000;D(1);11;M(3);Xn(3) | ASL  ASR |
| 1110;001;D(1);11;M(3);Xn(3) | LSL  LSR |
| 1110;011;D(1);11;M(3);Xn(3) | ROL  ROR |
|  | 0100;An(3);11;1;M(3);Xn(3) | LEA |

### Effective Addressing:

* + Handles all 6 addressing modes
    - Data Register Direct: **Dn**
    - Address Register Direct: **An**
    - Address Register Indirect: **(An)**
    - (An) w/ Post-increment: **(An)+**
    - (An) w/ Pre-decrement: **-(An)**
    - Immediate Data: **#**
    - Absolute Long Address: **(xxx).L**
    - Absolute Word Address: **(xxx).W**
  + Handles all non-supported addressing modes as “DATA” **(Maybe??)**
  + ***(optional)*** *Handles Displacement* ***for extra credit***

### Error Handling

* + Handles “DATA” as the first opcode with valid opcodes after
  + Handles Every-other lines of valid/data opcodes print successfully
  + Handles an entire document of “DATA” without crashing
  + Handles an empty document without crashing
  + Handles “file not found” without crashing
  + Handles being pointed at itself without crashing
  + Handles non-aligned structures as “DATA” without crashing