CSS 422: Hardware and Computer Organization

Disassembler – Program Specifications

Team – Single Precision REEE

# Manual

|  |  |
| --- | --- |
| **Constants** | **Description** |
| CUR\_OP\_CODE | Refers to the current 16-bit instruction set |
| OP\_SIZE | Refers to the current size of the current opcode which can be used for immediate data as well |
| G\_BUFFER | A “good” buffer that is used to print valid instructions |
| B\_BUFFER | A “bad” buffer that is used to print invalid instructions |
| IS\_VALID | Flag set to determine if it’s a valid code. Contains 1 if valid, 0 if is not valid |

|  |  |
| --- | --- |
| **Data Registers** | **Description** |
| A4 | Contains the G\_BUFFER where the valid instructions are stored |
| A5 | Stores the starting address that is given in the Config.cfg file |
| A6 | Stores the ending address that is given in the Config.cfg file |

|  |  |  |
| --- | --- | --- |
| **Printing Format** | **Sample Output** | **Example** |
| Valid Instruction | <ADDRESS> <OPERAND> <EA> | 00009000 MOVE.B #15, D0 |
| Invalid Instruction | <ADDRESS> DATA $<HEX VALUE> | 00009800 DATA $FFFF |

# Specification

This disassembler converts a memory set of instructions and data back to the 68000 Assembly Language. It is also called an inverse assembler which is the opposite of an assembler where the program takes basic computer instructions and converts them into a pattern of bits that the computer’s processor can use to perform its basic operations. After loading a starting and ending address, an optional test program can be loaded into memory outside the address range as well in order to guarantee the correctness of translating the instructions.

This project’s disassembler is intended to run on the EASy68K which is an open-source 68000 Structured Assembly Language Integrated Development Environment. The disassembler will keep printing supported instructions or data between starting from the provided start address until the end address is reached.

**The following table displays the specified Supported OP codes by the disassembler:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| MOVE | MOVEA | MOVEM | ADD | ADDA | SUB | SUBQ | MULS |
| DIVS | LEA | OR | ORI | NEG | EOR | LSR | LSL |
| ASR | ASL | ROL | ROR | BCLR | CMP | CMPI | BCS |
| BGE | BLT | BVC | BRA | JSR | RTS | NOP | DATA |

DATA notes that it’s an INVALID OP code. NOP while not mentioned in the supported specifications has been included as a special note from Professor Nash doing it in class already and expecting it to be supported.

**The following table displays the unsupported OP codes recognized by the disassembler:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ORI to CCR | ORI to SR | ANDI to CCR | ANDI to SR | ANDI | SUBI |
| ADDI | EORI to CCR | EORI to SR | EORI | BTST | BCHG |
| BSET | BTST | BCHG | BCLR | BSET | MOVEP |
| MOVE from SR | MOVE to CCR | MOVE to SR | NEGX | CLR | NOT |
| EXT | NBCD | SWAP | PEA | ILLEGAL | TAS |
| TST | TRAP | LINK | UNLK | MOVE USP | RESET |
| STOP | RTE | TRAPV | RTR | JMP | CHK |
| ADDQ | Scc | DBcc | BSR | BCC | BEQ |
| BGT | BHI | BLE | BLS | BMI | BNE |
| BPL | BVS | MOVEQ | DIVU | SBCD | SUBX |
| SUBA | CMPM | CMPA | MULU | ABCD | EXG |
| AND | ADDX | ROXR | ROXL | ROXR (rot) | ROXL (rot) |

For each of the recognized unsupported OP codes, IS\_VALID will be set to 0, and the proper DATA output will be loaded into B\_BUFFER. Scc and DBcc’s ‘main’ bits are recognized, none of the explicit conditionals for them are added in. ROXR (rot) and ROXL (rot) specifies rotational ROXR and ROXL.

**The following table displays the supported Effective Addressing modes:**

|  |  |
| --- | --- |
| Data Register Direct “Dn” | Address Register Direct “An” |
| Address Register Indirect “(An)” | Address Register Indirect with Post incrementing “(A0)+” |
| Address Register Indirect with Pre decrementing “-(SP)” | Immediate Data “#” |
| Absolute Long Address “(xxx).L” | Absolute Word Address “(xxx).W” |

**The table below highlights the Effective Address modes that support displacement:**

|  |  |  |
| --- | --- | --- |
| **BCS** | **BGE** | **BLT** |
| **BVC** | **BRA** | **BNE** |