**INSTRUCTION SET 24 bit**

**Total 29 Instructions RISC Based**

|  |  |  |
| --- | --- | --- |
| **Opcode** | **Mnemonic** | **Instruction** |
| **00000** | **ADD** | **Add 2 reg and store into 3rd reg** |
| **00001** | **ADC** | **ADD with added carry** |
| **00010** | **SUB** | **Subtract 2 reg and store in 3rd reg** |
| **00011** | **SBB** | **SUB with subtracted borrow** |
| **00100** | **INC** | **Increment Reg** |
| **00101** | **DEC** | **Decrement Reg** |
| **00110** | **NEG** | **Complement Reg** |
| **01000** | **AND** | **Logical And 2 reg and store in 3rd reg** |
| **01001** | **OR** | **Logical Or 2 reg and store in 3rd reg** |
| **01010** | **ROR** | **Rotate right reg upto 8 times** |
| **01011** | **ROL** | **Rotate left reg upto 8 times** |
| **01100** | **SHR** | **Shift right reg upto 8 times** |
| **01101** | **SHL** | **Shift left reg upto 8 times** |
| **01110** | **SETF** | **Set the flag reg** |
| **01111** | **PUSHF** | **Push carry flag into reg** |
| **10000** | **JMP** | **Unconditional jump** |
| **10001** | **JC** | **Jump if carry** |
| **10010** | **JNC** | **Jump if no carry** |
| **10011** | **JZ** | **Jump if zero** |
| **10100** | **JNZ** | **Jump if not zero** |
| **10101** | **JPE** | **Jump if parity even** |
| **10110** | **JPO** | **Jump if parity odd** |
| **11000** | **RES** | **Reset** |
| **11001** | **NOP** | **No-Operation** |
| **11010** | **SPC** | **Store Program Counter** |
| **11011** | **RSPC** | **Restore Program Counter** |
| **11100** | **LOAD** | **Load memory into reg** |
| **11101** | **STOR** | **Store reg to memory** |
| **11110** | **MOV** | **Move 1 reg to another** |

**Arithmetic Group 00**

**7 instructions**

Sel:

0=Reg

1=Immediate value(IV)

#: Register select

@: Address bits

1. Add R1🡨 R2 + (R3/IV) Update Flags **Mnemonic: ADD**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 00 | 000 | ##### | ##### | 0/1 | xxx##### or 8bit IV |

1. Add with Carry R1🡨 R2 + (R3/IV) + C Update Flags **Mnemonic: ADC**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 00 | 001 | ##### | ##### | 0/1 | xxx##### or 8bit IV |

1. Sub R1🡨 R2 - (R3/IV) Update Flags **Mnemonic: SUB**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 00 | 010 | ##### | ##### | 0/1 | xxx##### or 8bit IV |

1. Sub with borrow R1🡨 R2 – (R3/IV) -B Update Flags **Mnemonic: SBB**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 00 | 011 | ##### | ##### | 0/1 | xxx##### or 8bit IV |

1. INC Reg **Mnemonic: INC**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 00 | 100 | ##### | xxxxx | x | xxxx xxxx |

1. DEC Reg **Mnemonic: DEC**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 00 | 101 | ##### | xxxxx | x | xxxx xxxx |

1. COMPLEMENT Reg **Mnemonic: NEG**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 00 | 110 | ##### | xxxxx | x | xxxx xxxx |

**Logical & Misc Group 01**

**8 instructions**

Sel:

0=Reg

1=Immediate value(IV)

#: Register select

@: Address bits

F: Flag Bits

1. AND R1🡨 R2 & (R3/IV)  **Mnemonic: AND**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 01 | 000 | ##### | ##### | 0/1 | xxx##### or 8bit IV |

1. OR R1🡨 R2 | (R3/IV)  **Mnemonic: OR**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 01 | 001 | ##### | ##### | 0/1 | xxx##### or 8bit IV |

1. ROTATE RIGHT Reg>>bit val **Mnemonic: ROR**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 01 | 010 | ##### | xxxxx | x | xxxxxxxx |

1. ROTATE LEFT Reg<<bit val **Mnemonic: ROL**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 01 | 011 | ##### | xxxxx | x | xxxxxxxx |

1. SHIFT RIGHT Reg>>bit **Mnemonic: SHR**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 01 | 100 | ##### | xxxxx | x | xxxxxxxx |

1. SHIFT LEFT Reg<<bit **Mnemonic: SHL**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 01 | 101 | ##### | xxxxx | x | xxxxxxxx |

1. SET FLAGS **Mnemonic: SETF**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 01 | 110 | xxxxx | xxxxx | x | xxxxFFFF(4 flag bits) |

1. PUSH FLAG **Mnemonic: PUSHF**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 01 | 111 | ##### | xxxxx | x | xxxxxxxx |

**Branch Group 10**

**7 instructions**

#: Register select

@: Address bits INS MEMORY

1. JUMP Instruction Memory Address **Mnemonic: JMP**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | IV |
| 10 | 000 | xxxxx | xxxxx | x | @@@@@@@@ |

1. Jump If Carry **Mnemonic: JC**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | IV |
| 10 | 001 | xxxxx | xxxxx | x | @@@@@@@@ |

1. Jump If No Carry **Mnemonic: JNC**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | IV |
| 10 | 010 | xxxxx | xxxxx | x | @@@@@@@@ |

1. Jump If Zero **Mnemonic: JZ**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | IV |
| 10 | 011 | xxxxx | xxxxx | x | @@@@@@@@ |

1. Jump If Non Zero **Mnemonic: JNZ**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | IV |
| 10 | 100 | xxxxx | xxxxx | x | @@@@@@@@ |

1. Jump If Even Parity **Mnemonic: JPE**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | IV |
| 10 | 101 | xxxxx | xxxxx | x | @@@@@@@@ |

1. Jump If Odd Parity **Mnemonic: JPO**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | IV |
| 10 | 110 | xxxxx | xxxxx | x | @@@@@@@@ |

**Machine Control / Load Store Group 11**

**7 instructions**

#: Register select

@: Address bits DATA MEMORY

1. RESET **Mnemonic: RES**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 11 | 000 | xxxxx | xxxxx | x | xxxx xxxx |

1. NOP **Mnemonic: NOP**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 11 | 001 | xxxxx | xxxxx | x | xxxx xxxx |

1. Save PC **Mnemonic: SPC**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 11 | 010 | xxxxx | xxxxx | x | xxxx xxxx |

1. Restore PC **Mnemonic: RSPC**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 11 | 011 | xxxxx | xxxxx | x | xxxx xxxx |

1. Load Reg 🡨 Memory **Mnemonic: LOAD**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 11 | 100 | ##### | @@@@@ | @ | @@@@@@@@ |

1. Store Reg 🡪 Memory **Mnemonic: STOR**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 11 | 101 | ##### | @@@@@ | @ | @@@@@@@@ |

1. Move Reg 🡨 Reg **Mnemonic: MOV**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Group | Opcode | Reg | Reg | Sel | Reg/IV |
| 11 | 110 | ##### | ##### | x | xxxxxxxx |

**FLAG REGISTER**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parity(0=EP/1=OP)** | **Zero** | **Borrow** | **Carry** |
| **F** | **F** | **F** | **F** |