|  |  |
| --- | --- |
| Surname & Initials | Student Number |
| Monokoa T.J | 201600428 |
| Mantsi R | 202201932 |
| Mokotoi T.M | 202321234 |
| Kobeli | 202322593 |
| Tjabafu | 202322637 |
| Mapola | 202322602 |
| Mokotjo R.M | 202321189 |

**ISA Definition & Documentation Template**

1. **Overview & Motivation**

**Purpose and Application Context:** The custom ISA is specifically designed for low-cost, AI-enabled mobile phones targeting emerging markets like Lesotho. It directly addresses the application context identified in our Domain Analysis Report, focusing on edge-AI workloads including local voice recognition in Sesotho, biometric security, and intelligent connectivity management.

**Design Goals:** The three primary design goals are:

1. **Power Efficiency:** Maximizing battery life in environments with limited electricity access
2. **Cost-Effectiveness:** Minimizing silicon area and complexity for affordable hardware
3. **Responsive Performance:** Ensuring real-time processing for voice commands and security functions

**Suitability for Target Workload:** This RISC-based ISA with strategic custom extensions directly optimizes for the workload characteristics identified in our analysis. The 16-bit fixed-length format provides excellent code density for memory-constrained devices, while custom instructions like VCMPEQ.B, MAC, and BCNT accelerate the specific pattern matching, neural network inference, and biometric processing operations that dominate our target applications.

## **Architectural Design Choices**

**Instruction Philosophy:** **RISC-style**. This choice directly addresses the domain constraints of cost sensitivity and power efficiency. Simple, fixed-length instructions enable a simpler datapath and control unit, reducing chip size, power consumption, and design complexity compared to CISC alternatives.

**Registers:**

* **Number:** 16 general-purpose registers (x0-x15) - balancing performance needs with hardware cost
* **Size:** 32-bit - sufficient for the data types and address space requirements
* **Roles:**
  + x0: Hardwired zero value
  + x14: Stack Pointer (SP)
  + x15: Link Register (LR) for efficient function calls

**Data Types:** Supported word sizes are **Byte (8-bit)**, **Halfword (16-bit)**, and **Word (32-bit)**. 32-bit operations are the default for arithmetic and addressing. We explicitly exclude hardware floating-point in favor of fixed-point arithmetic, which meets the precision requirements of our target AI workloads while significantly reducing power and area costs.

**Addressing Modes:**

* **Register (R): ADD x1, x2, x3**
* **Immediate (I):** ADDI x1, x2, 100
* **Base-Offset:** LW x1, 40(x2)
* **PC-relative:** BEQ x1, x2, label

**Memory Model:** **Little-endian** byte ordering. Byte-addressable memory with word accesses aligned to 4-byte boundaries for hardware simplicity.

**Instruction Formats:** **Fixed-length 16-bit instructions** to maximize code density in memory-constrained devices. Two primary formats:

* **R-**Type (Register): [ opcode (4) | rd (4) | rs1 (4) | rs2 (4) ]
* **I-**Type (Immediate): [ opcode (4) | rd (4) | rs1 (4) | rs1 (4) | imm (4) ]

## **Instruction Set Summary**

|  |  |  |  |
| --- | --- | --- | --- |
| Group | Mnemonic | Syntax | Description |
| **Arithmetic** | ADD | ADD rd, rs1, rs2 | Add registers |
|  | ADDI | ADDI rd, rs1, imm | Add immediate |
|  | SUB | SUB rd, rs1, rs2 | Subtract registers |
|  | MULI | MULI rd, rs1, imm | Multiply by small immediate |
|  | **MAC** | MAC rd, rs1, rs2 | **Multiply-Accumulate (neural networks)** |
| **Logical** | AND | rd, rs1, rs2 | Bitwise AND |
|  | OR | rd, rs1, rs2 | Bitwise OR |
|  | XORI | rd, rs1, imm | Bitwise XOR immediate |
| **Memory** | LW | rd, imm(rs1) | Load Word |
|  | SW | rs2, imm(rs1) | Store Word |
|  | LHB | rd, imm(rs1) | Load Halfword/Byte (sign-extend) |
| **Control Flow** | JAL | rd, imm | Jump and Link |
|  | BEQ | rs1, rs2, imm | Branch if Equal |
|  | BNE | rs1, rs2, imm | Branch if Not Equal |
|  | BLT | rs1, rs2, imm | Branch if Less Than |
| **Custom AI** | VCMPEQ.B | rd, rs1, rs2 | Vector Compare Equal on Bytes |
|  | BCNT | rd, rs1 | Bit Count (biometric matching) |
|  | SLEEPM | imm | Sleep with Mode (power management) |

## **Instruction Encoding Summary**

**All instructions are 16 bits with two primary formats:**

**R-Type Format (4-4-4-4 bit fields):**

* Bits [15:12]: opcode
* Bits [11:8]: rd (destination register)
* Bits [7:4]: rs1 (source register 1)
* Bits [3:0]: rs2 (source register 2)

**I-Type Format (4-4-4-4 bit fields):**

* Bits [15:12]: opcode
* Bits [11:8]: rd (destination register)
* Bits [7:4]: rs1 (source register)
* Bits [3:0]: imm (4-bit immediate value, sign-extended)

**Encoding Regularity:** The consistent positioning of opcode (always bits 15-12) and register fields (always same positions) enables simple and fast instruction decoding, contributing to power efficiency and hardware simplicity.

1. **Design Rationale & Trade-offs**

**Simplicity vs Capability:** We heavily favored simplicity to meet our primary design constraints. Key exclusions include:

* **Hardware Floating-Point Unit:** Excluded due to high power and area cost; fixed-point arithmetic suffices for our target AI workloads
* **Complex Division Hardware:** Implemented in software for rare cases
* **Large Register File:** Limited to 16 registers to minimize hardware cost

The strategic inclusion of custom instructions (VCMPEQ.B, MAC, BCNT) provides significant performance gains for our specific applications without major hardware overhead.

**Code Density vs Performance:** The 16-bit fixed-length format represents a careful trade-off. While it limits immediate value range and register count, it provides superior code density compared to 32-bit RISC architectures. This is crucial for low-cost devices where memory constitutes a significant portion of the Bill of Materials (BOM) cost.

**Hardware Impact:** The simple 16-bit RISC design with only two instruction formats enables a clean, minimal datapath and control unit. The custom instructions integrate as small, dedicated hardware blocks:

* VCMPEQ.B requires a simple 4-byte SIMD comparison unit
* MAC integrates a multiply-accumulate path into the ALU
* BCNT uses a simple popcount circuit
* SLEEPM leverages existing power gating infrastructure

**Extensibility:** The ISA maintains excellent extensibility:

* 4-bit opcode space allows for 16 instruction classes, with room for future expansion
* The philosophy of adding domain-specific instructions means new application features can be accommodated with new custom opcodes
* The regular instruction format makes decoding new instructions straightforward

**Explicit Workload-to-ISA Mapping:**

**Voice Recognition - Low-latency neural network inference:** MAC instruction accelerates dot products in neural network layers

**Voice Recognition - Real-time keyword matching:** VCMPEQ.B instruction enables parallel byte comparison for audio frame analysis

**Biometric Security - Feature matching:** BCNT instruction enables efficient Hamming distance calculation

**Intelligent Connectivity - Power management:** SLEEPM instruction provides fine-grained power control during scanning intervals

**All Applications - Cost & power constraints:** 16-bit RISC core provides minimal silicon area and power consumption

This ISA represents a holistic solution that begins with domain constraints and application requirements, and delivers a processor architecture optimized specifically for low-cost AI-enabled mobile devices in emerging markets.