**Instruction Set Summary**



Opcodes 00000, 11000 to 11011, 11110 are UNDEFINED

Bit encoding has been optimized so that subsections can be identified by first 2/3 bits to aid controller simplicity.

**General Instruction Formatting**

LS: 1 = Load Data, 0 = Store Data No. = 1 or 2 or 3 L = Link Register

**Example Coding**

Data Manipulation

These operations are performed by the Arithmetic Logic Unit and examples are shown below.



The value ‘c’ corresponds to the carry bit flag in the ALU from the previous calculation.

CMP is a comparison instruction for performing a subtraction without saving the result. The updated status flags can then be used for a conditional branch.

Byte Immediate

These instructions ADD/SUB an 8-bit immediate value from the given register, replacing the result back in that register. Alternatively, the same formatting is used for loading the upper/lower byte of a register with an 8-bit immediate value.



Data Transfer

When loading data, the value at the memory location held in Ra, adds an offset held in Ro, and replaces the returned value in register Rd. When storing data, the same functionality is used, only with data transferring in opposite direction.



Control Transfer

This set of instructions adjust the value of the program counter by a relative amount determined by the location of the given label. Conditions are as follows:

* BR – Branch Always – Unconditionally branch to the stated location
* BNE – Branch if not equal – Conditionally branch if zero status flag (z) equals zero
* BE – Branch if equal – Conditionally branch if zero status flag (z) equals one
* BLT – Branch if < – Conditionally branch if negative status flag (n) equals one
* BGE – Branch if ≥ – Conditionally branch if negative status flag (n) equals zero
* BWL – Branch with link – Unconditionally branch to stated location, saving PC to link register (LR)
* RET – Return – Unconditionally jump to the value stored in the link register (LR)
* ABR – Absolute Branch – Unconditionally branch to the location held in register Ra

Stack Operations

These operations are for popping or pushing either a general purpose register or the link register onto the system stack, useful for context saving when an interrupt occurs. PUSH increments stack pointer (SP) and POP decrements stack pointer (SP) for a top-down growing stack. If the ‘L’ bit is set, the link register value will be used instead of the value in register Ra.