V810 Architecture: Summary

Register set:

General purpose 32 bit program registers
R0 and R26-31 reserved for explicit hardware operations
R1-R5 are used by assembler and compiler
R6-25 (20 registers) available for general program use
Multiple registers permit much improved program execution speed

Data types:

Integer/Unsigned; Byte (8 bits), Halfword (16 bits) and Word (32 bits)
Floating Point: 23 bit mantissa, signed 9 bit exponent
Bit String: any length bit string from or to anywhere in memory

Alignment:

Little Endian (All bytes, half words and words must line up to bit 0)
Data aligns to byte, half word and word boundaries
Instructions align to halfword and word boundaries
Truncation is performed if boundaries are misligned

Instruction Set:

Data transfer:	Register to register and register to or from memory
I/O:	Input and Output
Arithmetic:	Signed / unsigned add subtract, multiply and divide
Logical:	Compare, and, or, not, exclusive or
Shift:	Logical shift and arithmetic shift
System:	System register load and store
Bit string:	Move, and, or, exclusive or, search
Floating Point:	Add, subtract, multiply, divide, compare, convert
Branch:	Jump, conditional branch and jump and link
Miscellaneous:	Trap, RETI, Nop, Halt, and Compare and exchange

Instruction Execution Clock Cycles:

Most instructions:	1
Load / store:	1 to 3
Integer / logical:	1 except multiply = 13 and divide = 38
Shift:	1
Floating point:	24 to 44
Branch / jump:	1 if branch not taken, 3 if branch taken
Bit string:	3 to 12

Instruction Format:

Internal format of instruction word aligned on halfword boundaries
Instructions are may be half word (16 bits) or word (32 bits)

Addressing modes:

Load / Store instructions affect a + or - 32kb offset from base register
Immediate Load: Moves 16 bit value to upper or lower halfword of register
Function Call: Changes program execution to + or - 32mb offset from PC

Flags:

Move, load, store in, out, jump and branch instructions do not affect flags
All other operations affect flags appropriately to their operation
Floating point flags are set to signal an invalid result or error condition

Load/In:

Load instructions are sign extended into the upper bits
In instructions are zero extended into the upper bits

Function Call (jal): (synthesized through discrete instructions in V810)

When a jal op is executed the return address is saved to register r31
The return address may then be pushed onto a software stack
The function is then executed
The return address is then popped from the stack back to r31
A jump to the return address stored in r31 is then performed

Bitstring Operation:

Bit strings may be any length and may affect any area of memory
Search: find first 1 or zero bit up or down from specified bit
Move: moves specified bits from source address to destination address
Logical (BitBlt): ands, ors or xors specified range of bits

Floating point operations: (add, subtract, multiply and divide)

.9 million floating point operations per second at 25 MHz
Add = 24, subtract = 26, multiply = 27 and divide = 44 clock cycles
Conversion from 32 bit float to/from 32 bit integer instructions
Floating point data is handled in normal 32 bit general registers

Interrupts and Exceptions:

Virtual Game Boy has 5 levels of hardware maskable interrupts
Software exceptions: are produced by TRAP codes and illegal operations
Interrupts and exceptions branch to fixed addresses to specified routines
Return from Interrupt or exception (RETI) restores PC and branches to it
Lower level interrupts may be interrupted by higher level interrupts
Higher level interrupts complete before responding to lower ones
Interrupts will (nest) properly

Important Hardware Issues

Some pipeline architectures allow succesive instruction faults to exist
In the V81 Hardware interlock prevents faults and reduces instructions
Load/Store interlock stops base register change before load/store hazard
Flag interlock prevents conditional branch after flag modification hazard
Extra clock cycles are automatically inserted to “flush the pipe” if needed

Cache:

ROM = 3 clock cycles, RAM = 2 clock cycles, Cache = 1 clock cycle
Efficient use of Cache is mandatory for fast applications
Cache = 1 k bytes, direct mapping with 8 byte block and 4 byte sub-block
Cache is best used for often executed loops and routines
Cache can be enabled/disabled, cleared and saved to or from memory
The cache control word (CHCW) register controls the use of cache