Rhea Mae Edwards
ECE 375

LAB 5 PRELAB

1. ADDITION:

ADC – Add with Carry

Adds two registers and the contents of the C Flag and places the result in the destination register Rd.

ADD – Add without Carry

Adds two registers without the C Flag and places the result in the destination register Rd.

ADIW – Add Immediate to Word

Adds an immediate value (0 - 63) to a register pair and places the result in the register pair. This instruction operates on the upper four register pairs, and is well suited for operations on the pointer registers.

SUBTRACTION:

SBC – Subtract with Carry

Subtracts two registers and subtracts with the C Flag, and places the result in the destination register Rd.

SBCI – Subtract Immediate with Carry SBI – Set Bit in I/O Register

Subtracts a constant from a register and subtracts with the C Flag, and places the result in the destination register Rd.

SBIW – Subtract Immediate from Word

Subtracts an immediate value (0-63) from a register pair and places the result in the register pair. This instruction operates on the upper four register pairs, and is well suited for operations on the Pointer Registers.

SUB – Subtract Without Carry

Subtracts two registers and places the result in the destination register Rd.

SUBI – Subtract Immediate

Subtracts a register and a constant, and places the result in the destination register Rd. This instruction is working on Register R16 to R31 and is very well suited for operations on the X, Y, and Z-pointers.

MULTIPLICATION:

FMUL – Fractional Multiply Unsigned

Performs 8-bit × 8-bit → 16-bit unsigned multiplication and shifts the result one bit left.

FMULS – Fractional Multiply Signed

Performs 8-bit × 8-bit → 16-bit signed multiplication and shifts the result one bit left.

FMULSU – Fractional Multiply Signed with Unsigned

Performs 8-bit × 8-bit → 16-bit signed multiplication and shifts the result one bit left.

MUL – Multiply Unsigned

Performs 8-bit × 8-bit → 16-bit unsigned multiplication.

MULS – Multiply Signed

Performs 8-bit × 8-bit → 16-bit signed multiplication.

MULSU – Multiply Signed with Unsigned

Performs 8-bit × 8-bit → 16-bit multiplication of a signed and an unsigned number.

2. [Pseudocode]

- $0111:$0110 and $0121:$0120 (16-bit numbers)
- Add them together
- Store 16-bit result (in data memory addresses $0101:$0100)

Note: The syntax “$0111:$0110” is meant to specify that the function will expect little-endian data, where the highest byte of a multi-byte value is stored in the highest address of its range of addresses.

3. [Pseudocode]

- $0111:$0110 (16-bit number)
- Subtract from $0121:$0120 (16-bit number)
- Store 16-bit result (into $0101:$0100)