AVR ASM INTRODUCTION

Search this site

AVR ASSEMBLER TUTOR >

2a. BASIC ARITHMETIC

BASIC AVR ARITHMETIC v1.1
ADDITION and SUBTRACTION

by RetroDan@GMail.Com

CONTENTS:

A single register in the 8-Bit AVRs are 8 bits wide so they can hold values from zero to 255 (zero to $FF in Hex). To work with larger numbers we combine the use of several registers. First we will examine commands that work with single registers and later we will combine these commands to handle larger numbers.

1. ADDING AND SUBTRACTING ONE FROM A REGISTER:

Adding and subtracting one to a register is done so often, such as modifying counters in loops, that there are specific commands that do just that. The increment (INC) and decrement (DEC) will add or subtract a one from a register.

```
.DEF N = R0          ;Define “N” as Register Zero (R0)
     INC N           ;Increment N (R0) by adding one
     DEC N           ;Decrement N (R0) by subtracting
```

Another advantage of using the increment (INC) and decrement (DEC) commands over standard addition and subtraction is that they require less operands and work with all registers R0 to R31. Some commands only work on registers R16 to R31.

For example to create a loop that goes from zero to 255:

```
.DEF  N = R0          ;Define “N” as Register Zero

      CLR N           ;CLear Register N to all zeros
LOOP:                 ;A label for our loop
      INC N           ;INCrement N by one
       BRNE LOOP      ;BRanch if Not Equal to zero
                      ;Keep looping until N wraps arou
                      ;255 ($FF)to zero again
DONE:                 ;N (R0)=ZERO so we are done our
```

To create a loop that counts backward from 255 ($FF) to one:

```
.DEF  N = R0
      SER N           ;SEt Register N to all ones = 25
LOOP: DEC N           ;DECrement N by one
       BRNE LOOP      ;Keep looping until N equals zer
DONE:                 ;N (R0)=ZERO so we are done our
```

2. LOADING A REGISTER WITH A CONSTANT:

Instructions that work with constants are referred to as “immediate” mode instructions. To load a constant value into a register we use the LoaD Immediate (LDI) command.

```
.DEF  A = R16
      LDI A,10        ;Set A (R16) to value of 10
```

Only registers in the range R16 to R31 can be loaded immediate. We cannot load a constant into the registers R0 to R15 directly. It would have to be loaded into a valid register first then copied. To load the value of 10 into register zero (R0):

```
.DEF  N = R0
.DEF  A = R16
      LDI A,10       ;Set A (R16) to value of 10
      MOV N,A        ;Copy contents of A(R16) to N(R0
```

Constants can be expressed in a number of ways. Below are some examples.

```
.DEF  A = R16

      LDI A,10       ;Set A (R16) to value of 10 (Dec

      LDI A,010      ;Set A to value of 9 (Octal Numb
                     ;The leading Zero indicates an O

      LDI A,0x10     ;Set A to 16 (Hexadecimal Number
                     ;The leading “0x” indicates Hex

      LDI A,$10      ;Set A to 16, alternate format o
                     ;The leading “$” indicates Hex N

      LDI A,0b0000_0010 ;Set A to 2 (Binary Number)
                     ;The leading “0b” indicates a Bi

      LDI A,'1'      ;Set A to 49, the ASCII value of
                     ;The '' indicates an ASCII Chara
```

3. ADDING AND SUBTRACTING A CONSTANT FROM A SINGLE REGISTER:

Constants are values that do not change during the execution of a program. Since they don't change in value, there is no need to use an additional registers to store them.

To subtract a constant from a register we will use the SUBtract Immediate instruction (SUBI). Please note that the SUBI command does not work with Registers Zero to Fifteen (R0-R15), so our working register must be in the range of (R16 to R31). In the example below we use Register 16 (R16).

To subtract two from register A:

```
.DEF  A = R16        ;Set “A” to Register 16 (R16)
      SUBI A,2       ;Subtract two from A (decrement
```

Adding a constant to a register is a little tricky using AVR ASM because it does not have an add immediate instruction. The way we can do it is to negate the constant we wish to add and subtract that negative constant from the register. This takes advantage of the fact that subtracting a negative number results in a double negative that is the same as addition: X-(-2)=X+2.

```
.DEF  A = R16        ;Set “A” to Register 16 (R16)
      SUBI A,-2      ;Subtract negative two from A, A
                     ;The net result is adding two to
```

4. ADDING TWO SINGLE-BYTE REGISTERS:

To add two registers together we simply load the registers with the values we want and use the ADD instruction which works with all registers R0 to R31.

```
.DEF  A = R16        ;Set “A” to Register 16 (R16)
.DEF  B = R18        ;Set “B” to Register 18 (R18)

      LDI A,1        ;Store value of one in R16
      LDI B,48       ;Store value of 48 in R18
      ADD A,B        ;Add the value stored in B
                     ;to the value stored in A (A=A+B
                     ;The result is that A=49 and
                     ;the value stored in B remains u
```

Since a single register can only hold 8 bits with a maximum value of 255. To be complete we should check if our total is more than 255. When we ADD two registers together that total more than 255 the “Carry Flag” will be set automatically.

```
      ADD A,B        ;Add the value stored in B to A
       BRCS OVERFLOW ;BRanch if Carry Set to overflow
```

5. SUBTRACTING TWO SINGLE-BYTE REGISTERS:

For us to subtract two registers we use the subtract (SUB) command. For example to convert a numerical character to its numerical value we subtract 48.

```
      LDI A,'1'      ;Store value of 49 in R16
                     ;49 is the ASCII value of the ch
      LDI B,48       ;Store value of 48 in R18
      SUB A,B        ;Subtract the value stored in B
                     ;The value left in A is one.
```

If we need to check if the result of our subtraction is negative, the carry flag is automatically set when our result goes below zero. We can check that flag with the Branch if Carry Set (BRCS) command.

```
      SUB A,B         ;Subtract the value stored in B
       BRCS UNDERFLOW ;BRanch if Carry Set to underfl
```

6. ADDING MULTI-BYTE CONSTANTS:

If we wish to work with numbers greater than 255 we can use two registers together to give us a range from zero to 65,535 ($FFFF). To automatically break a constant like 420 into two bytes we can use the HIGH and low functions.

```
.DEF  AL = R16         ;To hold low byte of value
.DEF  AH = R17         ;To hold HIGH byte of value

      LDI AL,LOW(420)  ;Copy low byte of 420
      LDI AH,HIGH(420) ;Copy HIGH byte of 420
```

We can expand this method for numbers up to four bytes long with the BYTE2(), BYTE3() and BYTE4() functions. This gives us a range from zero to 4,294,967,295 approximately 4.3 Billion.

```
.DEF  A1 = R16         ;To hold low byte of value
.DEF  A2 = R17         ;To hold second byte of value
.DEF  A3 = R18         ;To hold third byte of value
.DEF  A4 = R19         ;To hold fourth byte of value

      LDI A1,LOW(420420)   ;Copy low byte of 420420
      LDI A2,BYTE2(420420) ;Copy second byte of 42042
      LDI A3,BYTE3(420420) ;Copy third byte of 420420
```

```
LDI A4,BYTE4(420420) ;Copy fourth byte of 42042
```

7. SUBTRACTING MULTI-BYTE CONSTANTS:

To subtract a two byte constant from a two byte value stored in the register pair AL/AH we first subtract the low byte of the constant from the low byte register (AL). If the result of the subtraction is negative the carry flag gets set. This carry flag is used by the SuBtract with Carry Immediate command (SBCI) below to indicate that the result of subtracting the two lower bytes resulted in a negative number so a one needs to be “borrowed” from the result of subtracting the two high bytes.

```
SUBI AL,LOW(420) ;Subtract two low bytes: AL=AL
SBCI AH,HIGH(420);Subtract high bytes and inclu
                 ;AH=AH-HIGH(420)-CarryFlag
 BRCS UNDERFLOW  ;If the result of the complete
                 ;is negative then the Carry Fl
```

A similar method can be used to subtract four-byte numbers.

```
SUBI A1,LOW(420420)  ;Subtract two low bytes: A
SBCI A2,BYTE2(420420);Subtract second bytes and
SBCI A3,BYTE3(420420);Subtract third bytes and
SBCI A4,BYTE4(420420);Subtract fourth bytes and
```

Since the AVRs do not have a Add Immediate instruction we have to subtract the negative of the constant we wish to add. Once again we use the low() and high() functions to break our constant into two bytes.

```
SUBI AL,LOW(-420) ;Add two low bytes: AL=AL+LOW
SBCI AH,HIGH(-420);Add high bytes and include C
                  ;AH=AH+HIGH(420)+CarryFlag
 BRCS OVERFLOW    ;If the result of the complet
                  ;is greater than 65,535 then
                  ;Carry Flag is set
```

We can use the same method to add four byte numbers.

```
SUBI A1,LOW(-420420)  ;Add two low bytes
SBCI A2,BYTE2(-420420);Add second bytes and inc
```

```
      SBCI A3,BYTE3(-420420);Add third bytes and incl
      SBCI A4,BYTE4(-420420);Add fourth bytes includi
```

8. ADDING MULTI-BYTE REGISTERS:

To add two 16 bit registers we first add the two lower bytes together with the ADD command, then we add the two high bytes together and include the carry bit which contains any overflow from the addition of the lower bits. This is accomplished with the Add with Carry command (ADC).

```
.DEF  AL = R16          ;To hold low byte of value1
.DEF  AH = R17          ;To hold high byte of value1
.DEF  BL = R18          ;To hold low byte of value2
.DEF  BH = R19          ;To hold high byte of value2

     ADD AL,BL          ;Add the lower bytes together
     ADC AH,BH          ;Add the higher bytes and incl
      BRCS OVERFLOW
```

We can expand this method to add together two four-byte registers.

```
.DEF  A1 = R16          ;To hold low byte of value1
.DEF  A2 = R17          ;To hold second byte of value1
.DEF  A3 = R18          ;To hold third byte of value1
.DEF  A4 = R19          ;To hold fourth byte of value1
.DEF  B1 = R20          ;To hold low byte of value2
.DEF  B2 = R21          ;To hold second byte of value2
.DEF  B3 = R22          ;To hold third byte of value2
.DEF  B4 = R23          ;To hold fourth byte of value2

     ADD A1,B1          ;Add the lower bytes together
     ADC A2,B2          ;Add the second bytes and incl
     ADC A3,B3          ;Add the third bytes and inclu
     ADC A4,B4          ;Add the fourth bytes and incl
```

9. SUBTRACTING MULTI-BYTE REGISTERS:

For subtraction we use a similar procedure to addition, we first subtract the lower bits with the subtract command (SUB) then we subtract the two high bytes including the carry bit with the subtract with

carry command (SBC). The carry flag will indicate that the addition of the lower bytes resulted in a negative number and that a one needs to be “borrowed” from the high bytes. If the carry flag is one, by subtracting its value from the high bytes we accomplish the needed borrowing of one.

```
SUB AL,BL          ;Subtract two low bytes
SBC AH,BH          ;Subtract two high bytes and i
 BRCS UNDERFLOW
```

The same procedure can be used to subtract larger multi-byte numbers.

```
SUB A1,B1          ;Subtract the lower bytes
SBC A2,B2          ;Subtract the second bytes and
SBC A3,B3          ;Subtract the third bytes and
SBC A4,B4          ;Subtract the fourth bytes and
```

Comments

You do not have permission to add comments.