EE-222: Microprocessor Systems

AVR Microcontroller:
Status Register and Directives

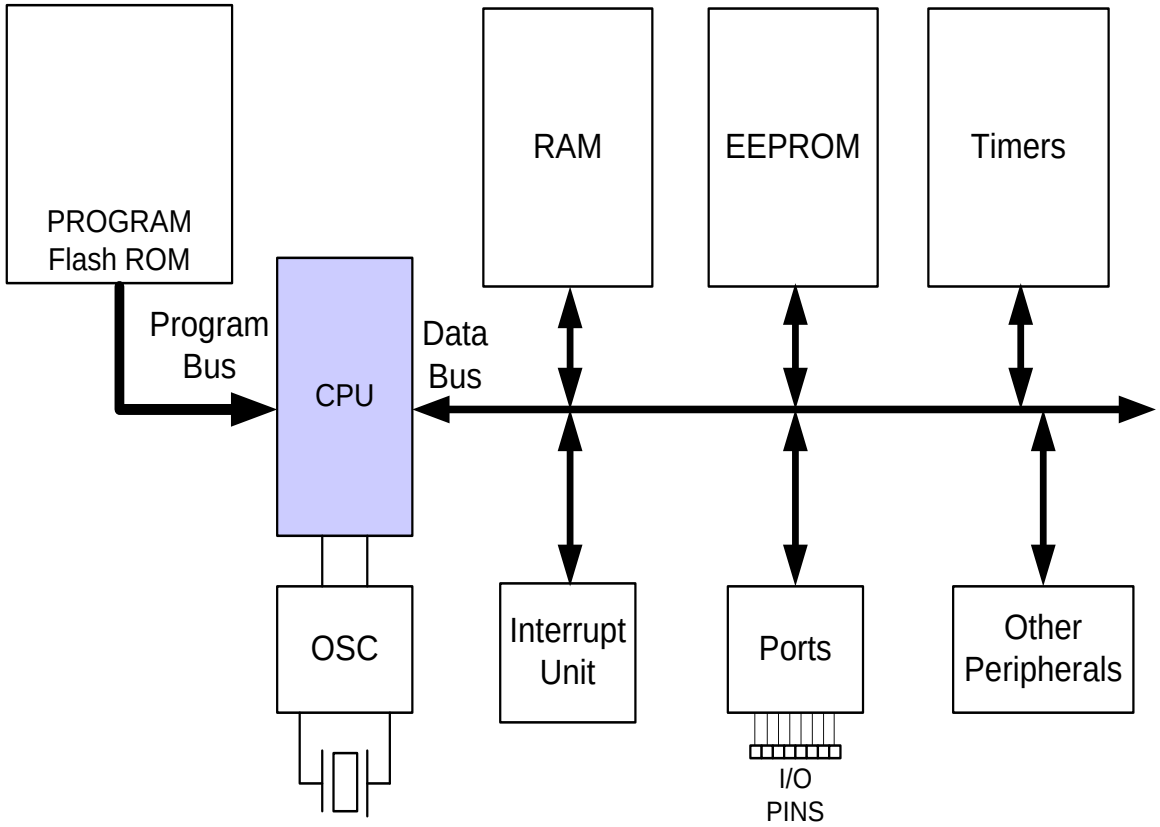
Instructor: Dr. Arbab Latif

NUST
SCHOOL OF ELECTRICAL ENGINEERING &
COMPUTER SCIENCE (SEECS)

Review

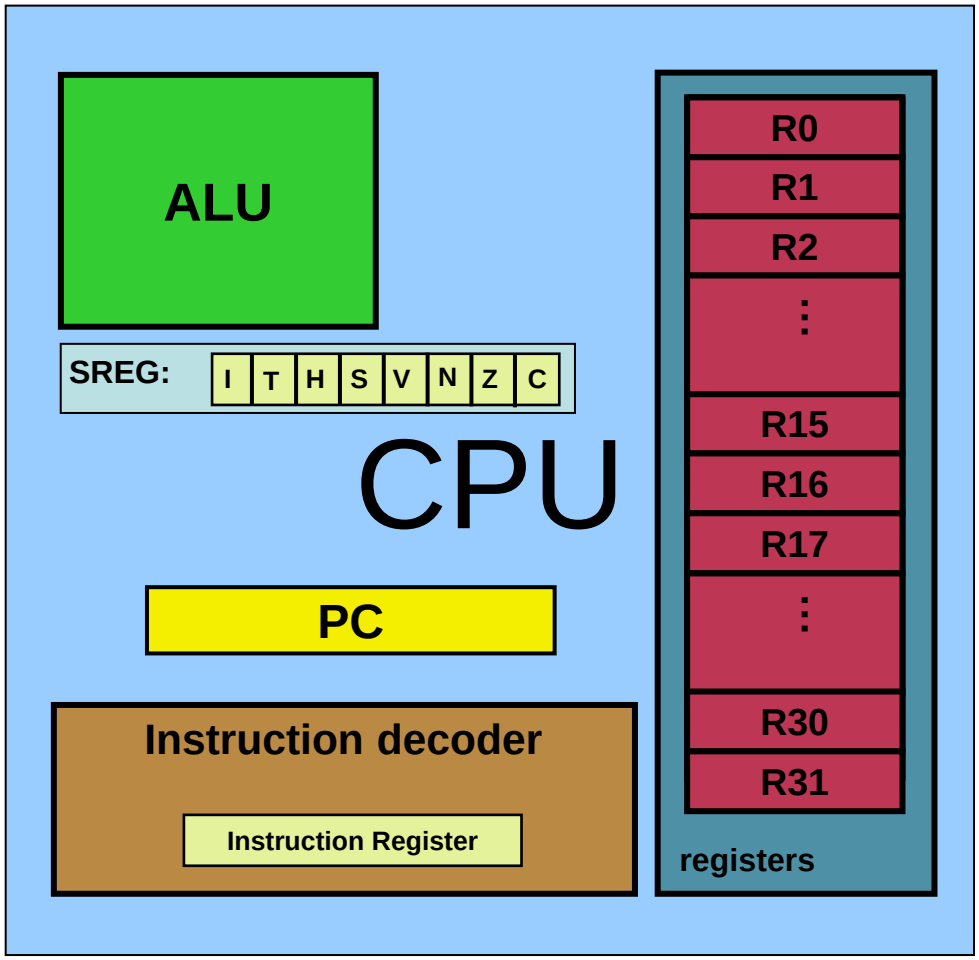
Programming AVR

- To program in Assembly language, we must understand the registers and architecture of a given CPU and the role they play in processing data.



AVR Registers

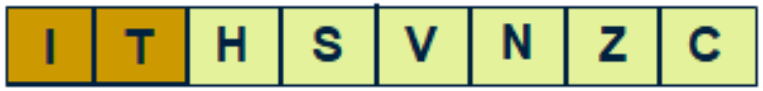
- AVR Registers:
  - Two Types:
    - General Purpose:
    - Special Purpose:
      - Three Registers:
        - » Program counter
        - » Stack Pointer
        - » Status Register



AVR Status Register

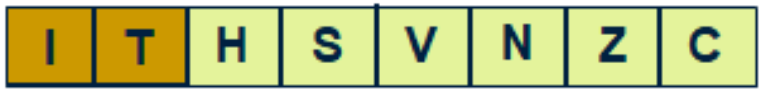
AVR Status Register: SREG

- Status Register (SREG) aka Flag Register:
  - To indicate arithmetic conditions like ***carry bit***
  - is an 8-bit register



I/O Register $3F : SREG

- All Bits are R/W:
  - I – Global Interrupt Enable
  - T – Bit Copy Storage
  - H – Half Carry Flag
    - set when there is a carry from D3 to D4 during an ADD or SUB
  - S – Sign Bit
    - XOR of N and V flags
  - V – Two's Complement Overflow Flag
    - set whenever the signed number result is too large
  - N – Negative Flag
    - N = 0; arithmetic operation result is positive
    - N = 1; arithmetic operation result is negative
  - Z – Zero Flag
    - If an arithmetic or logic operation result is ZERO
  - C – Carry Flag
    - set whenever there is a carry from the D7 bit



Status Register (SREG)

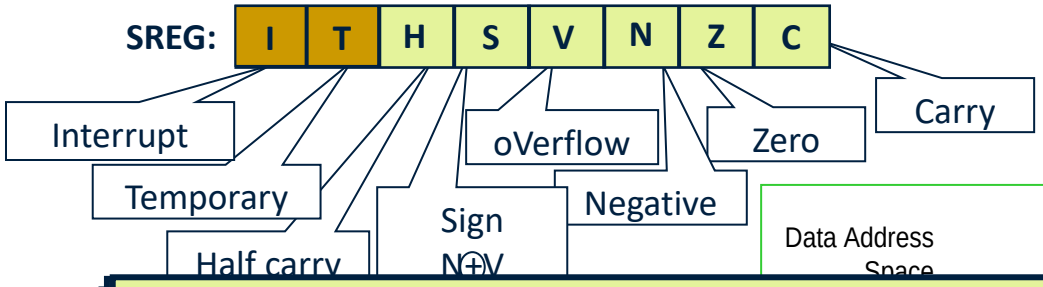


Table 2-5: AVR Branch (Jump) Instructions Using Flag Bits	
Instruction	**Action**
BRLO	Branch if C = 1
BRSH	Branch if C = 0
BREQ	Branch if Z = 1
BRNE	Branch if Z = 0
BRMI	Branch if N = 1
BRPL	Branch if N = 0
BRVS	Branch if V = 1
BRVC	Branch if V = 0

Example: Show the status of the C, H, and Z flags after the su… from 0x9C in the following instructions:

```
LDI     R20, 0x9C
LDI     R21, 0x9C
SUB     R20, R21        ;subtract R21 from R20
```

Solution:

	$9C 1001 1100	
-	$9C 1001 1100	
	$00 0000 0000	R20 = $00

C = 0 because R21 is not bigger than R20 and there is no borrow from D8 bit.
Z = 1 because the R20 is zero after the subtraction.
H = 0 because there is no borrow from D4 to D3.

Example: ADD Instruction and Z Flag

- Show the status of Z flag during the execution of the following program:

```
LDI R20,4
DEC R20
DEC R20
DEC R20
DEC R20
```

After	Value of R20	The Z flag
LDI R20, 4	4	0
DEC R20	3	0
DEC R20	2	0
DEC R20	1	0
DEC R20	0	1

NOT all instructions affect the flags

Instruction	C	Z	N	V	S	H
ADD	X	X	X	X	X	X
ADC	X	X	X	X	X	X
ADIW	X	X	X	X	X	
AND		X	X	X	X	
ANDI		X	X	X	X	
CBR		X	X	X	X	
CLR		X	X	X	X	
COM	X	X	X	X	X	
DEC		X	X	X	X	
EOR		X	X	X	X	
FMUL	X	X				
INC		X	X	X	X	
LSL	X	X	X	X		X
LSR	X	X	X	X		
OR		X	X	X	X	
ORI		X	X	X	X	
ROL	X	X	X	X		X
ROR	X	X	X	X		
SEN			1			
SEZ		1				
SUB	X	X	X	X	X	X
SUBI	X	X	X	X	X	X
TST		X	X	X	X	

AVR Data Format and Directives

AVR Data Type

- Only ONE data type:
  - It is 8 bits
  - Therefore size of each register is also 8 bits

- Four ways to represent a byte in AVR:
  - HEX numbers:
    - Put 0x in front like this: LDI R16, 0x99
    - Put $ in front of the number this: LDI R22, $99
  - Binary numbers:
    - LDI R16, 0b10011001
  - Decimal numbers:
    - Simply use decimal and nothing before or after it
      - LDI R17, 12
  - ASCII characters:
    - Use single quotes
      - KDI R23, '2' ; R23 = 00110010 or 32 in hex

Assembler Directives

- As instructions are (directions) for the CPU,
  - *Assembler directives are directions to the assembler.*
    - Not translated to machine language during the assembly process

Some AVR Assembler Directives

Directives	Description
BYTE	Reserve a Byte for a Variable
CSEG	Define the Code Segment Section
DB	Define Constant Byte(s)
DEF	Define a Symbolic Name
DEVICE	Define which Microcontroller to Assemble for
DSEG	Define the Data Segment Section
DW	Define Constant Word(s)
ENDMACRO	Signifies the End of a Macro
EQU	Set a Symbol Equal to an Expression
ESEG	Define the EEPROM Section
Exit	Exit from a File
INCLUDE	Read Source Code from a File
LIST	Turn List Generation ON
LISTMAC	Turn Macro Expression ON
MACRO	Signifies the Beginning of a Macro
NOLIST	Turn List Generation OFF
ORG	Set Program Origin
SET	Set Symbol to an Expression

.EQU and .SET

- .EQU *name* = *value*
  - is used to define a constant value or a fixed address
  - A label assigned to a value by the EQU directive is a constant and can not be changed or redefined
  - Example:

```
.EQU     COUNT = 0x25
         LDI     R21, COUNT              ;R21 = 0x25
         LDI     R22, COUNT + 3          ;R22 = 0x28
```

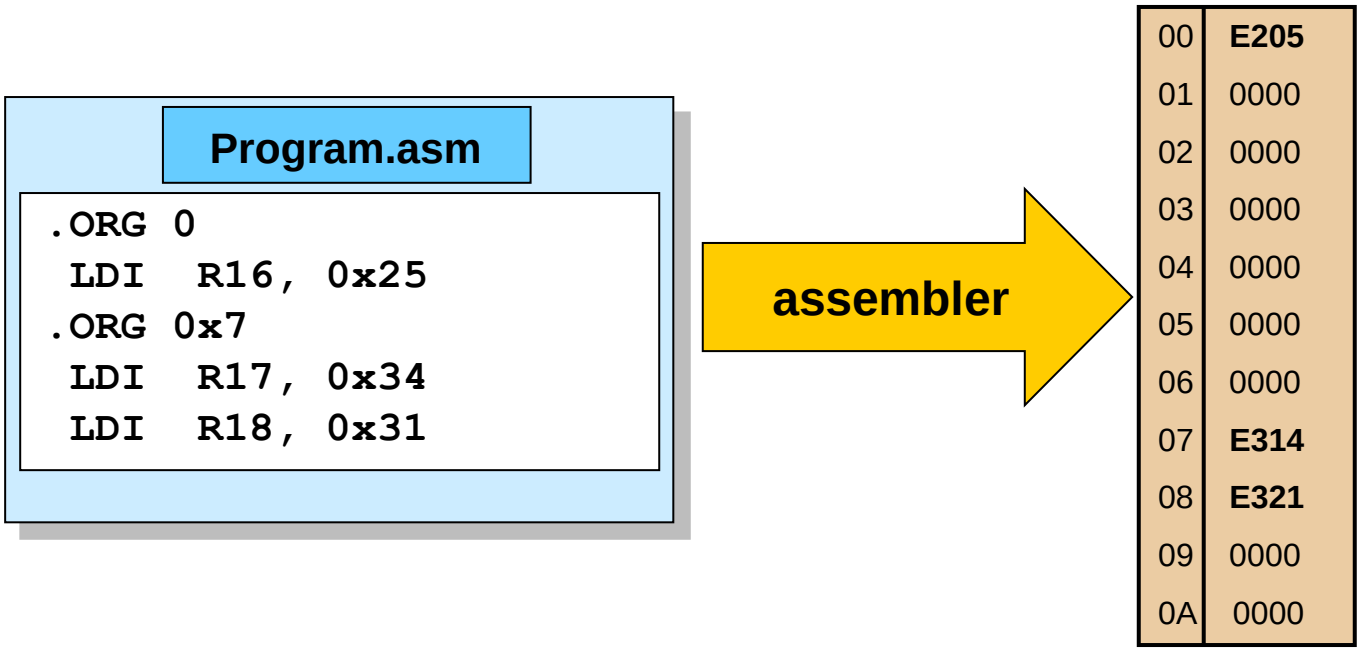
- .SET *name* = *value*
  - *Example:*

```
.SET     COUNT = 0x25
 LDI     R21, COUNT              ;R21 = 0x25
 LDI     R22, COUNT + 3          ;R22 = 0x28
.SET     COUNT = 0x19
 LDI     R21, COUNT              ;R21 = 0x19
```

  - Note that a label assigned to a value by the SET may be reassigned later.

.ORG (origin)

- The ORG directive is used to indicate the beginning of the address
  - It can be used for both code and data

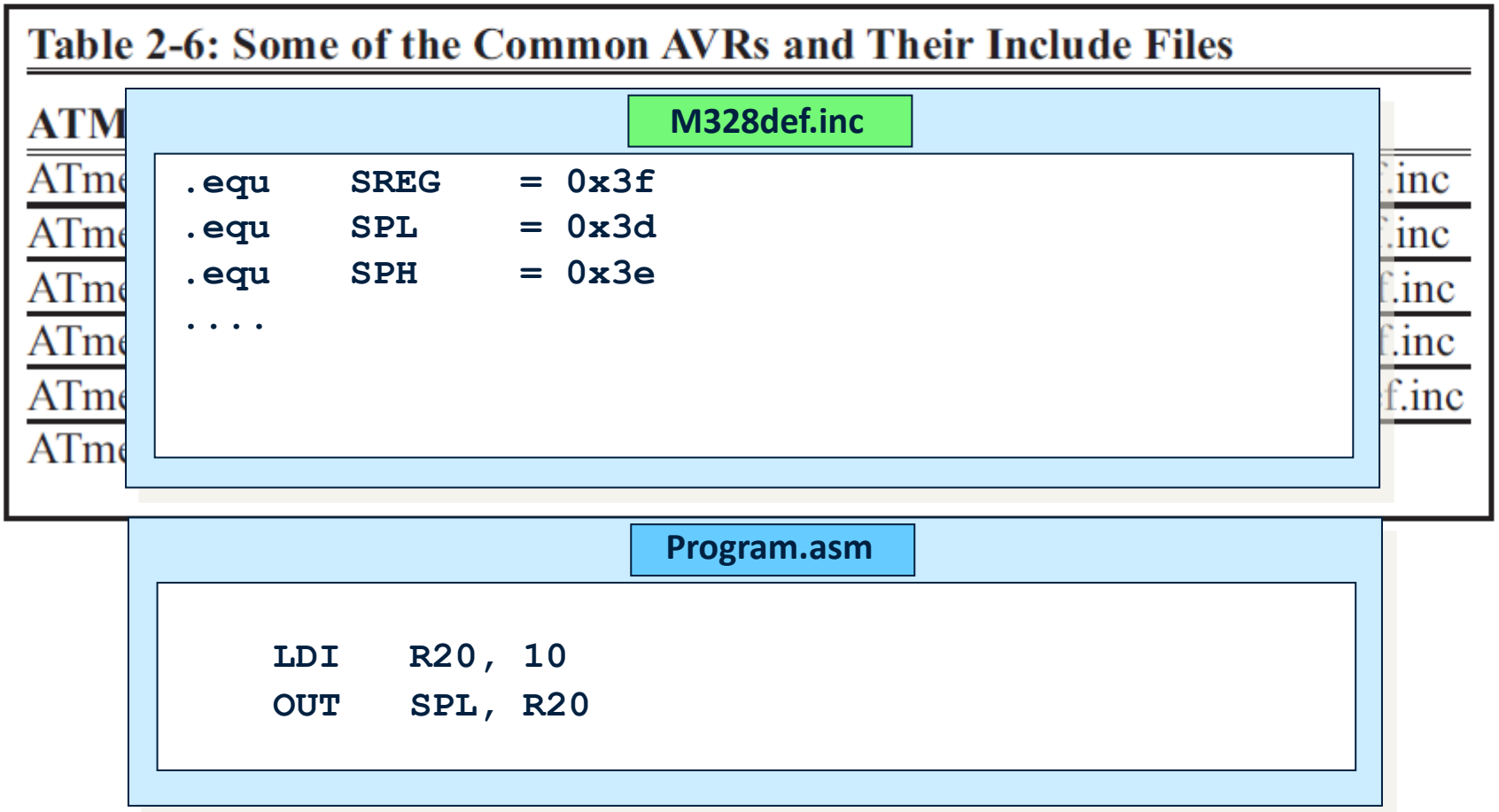


.ORG (origin) – Watch Out

- If an ORG directive is given within a Data Segment, then it is the SRAM location counter which is set.
- If the directive is given within a Code Segment, then it is the Program memory counter which is set.
- If the directive is given within an EEPROM Segment, it is the EEPROM location counter which is set.

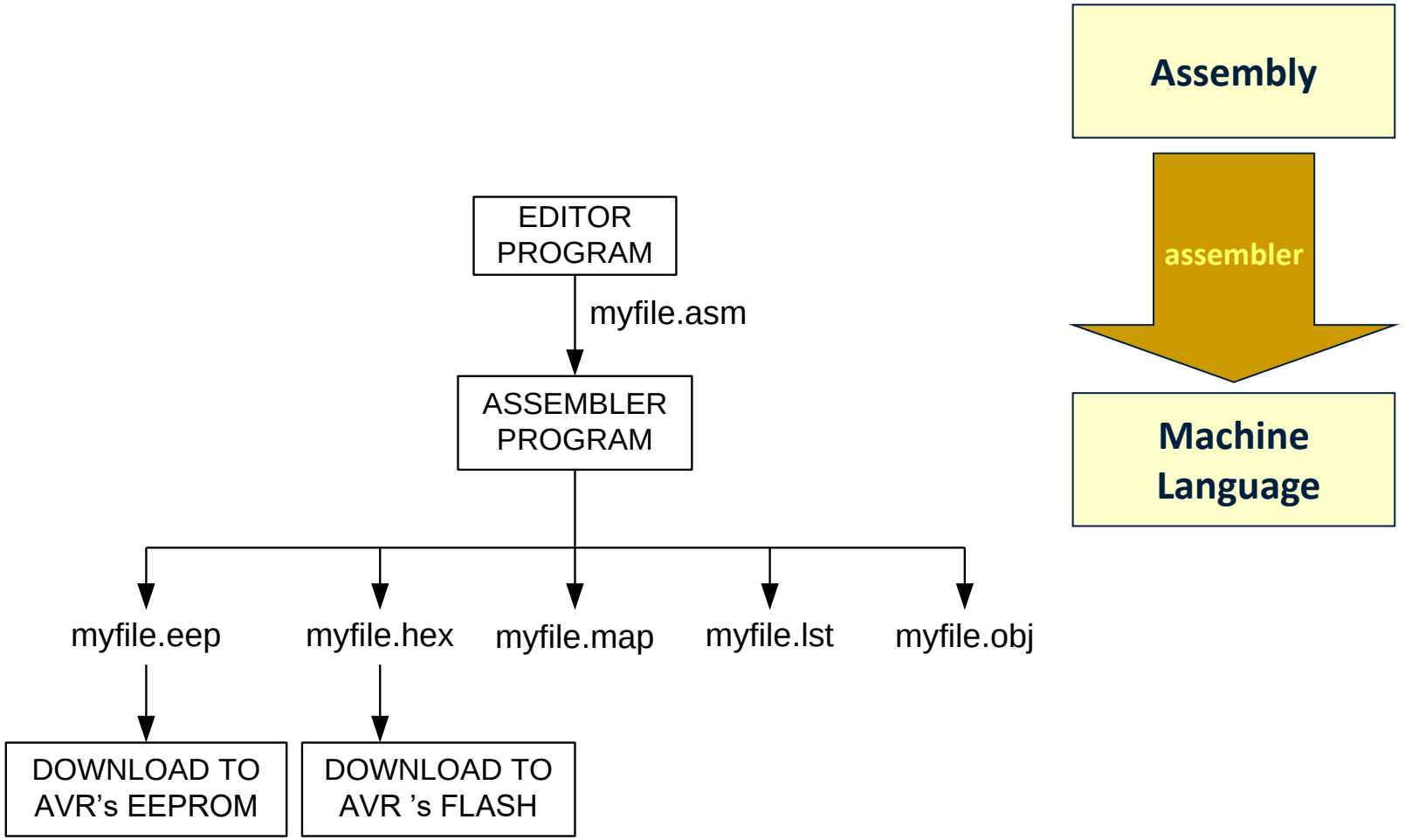
.INCLUDE

- .INCLUDE "*filename.ext*"



Assembling an AVR Program

Assembling an AVR Program



List File

```
                ;store SUM in SRAM location 0x300.
                   .DEVICE ATMega32
                   .EQU  SUM   = 0x300      ;SRAM loc $300 for SUM

                   .ORG 00                  ;start at address 0
000000 e205        LDI R16, 0x25            ;R16 = 0x25
000001 e314        LDI R17, $34             ;R17 = 0x34
000002 e321        LDI R18, 0b00110001      ;R18 = 0x31
000003 0f01        ADD R16, R17             ;add R17 to R16
000004 0f02        ADD R16, R18             ;add R18 to R16
000005 e01b        LDI R17, 11              ;R17 = 0x0B
000006 0f01        ADD R16, R17             ;add R17 to R16
000007 9300 0300   STS SUM, R16             ;save the SUM in loc $300
000009 940c 0009 HERE: JMP HERE             ;stay here forever

RESOURCE USE INFORMATION
------------------------
...
Memory use summary [bytes]:
Segment    Begin     End       Code    Data    Used     Size    Use%
---------------------------------------------------------------------
[.cseg]  0x000000 0x000016      22       0      22 unknown       -
[.dseg]  0x000060 0x000060       0       0       0 unknown       -
[.eseg]  0x000000 0x000000       0       0       0 unknown       -

Assembly complete, 0 errors, 0 warnings
```

Reading

- The AVR Microcontroller and Embedded Systems: Using Assembly and C by Mazidi et al., Prentice Hall
  - Chapter-2: 2.4 – 2.7

THANK YOU