ATmega1284P Assembly III

Outline:

- The ATmega1284P Status Register
- Pointer Registers
- Defining Tables in the Program memory
- Subroutines
- Elementary example program

Pointer Registers I

- **The pairs of registers:**

  r31:r30 (Z), r29:r28 (Y), r27:r26 (X)

  are special pointer registers and can be used to access memory via :

```
LD Rd, X  or  LD Rd, Y   or  LD Rd, Z
```

Meaning: Load the contents of memory address X or Y or Z on register Rd

Pointer Registers II

- **We can chose to increment memory address after the operation is finished:**

```
LD Rd, X+    LD Rd, Y+    LD Rd, Z+
```

or decrement before the operation :

```
LD Rd, -X    LD Rd, -Y    LD Rd, -Z
```

Pointer Registers III

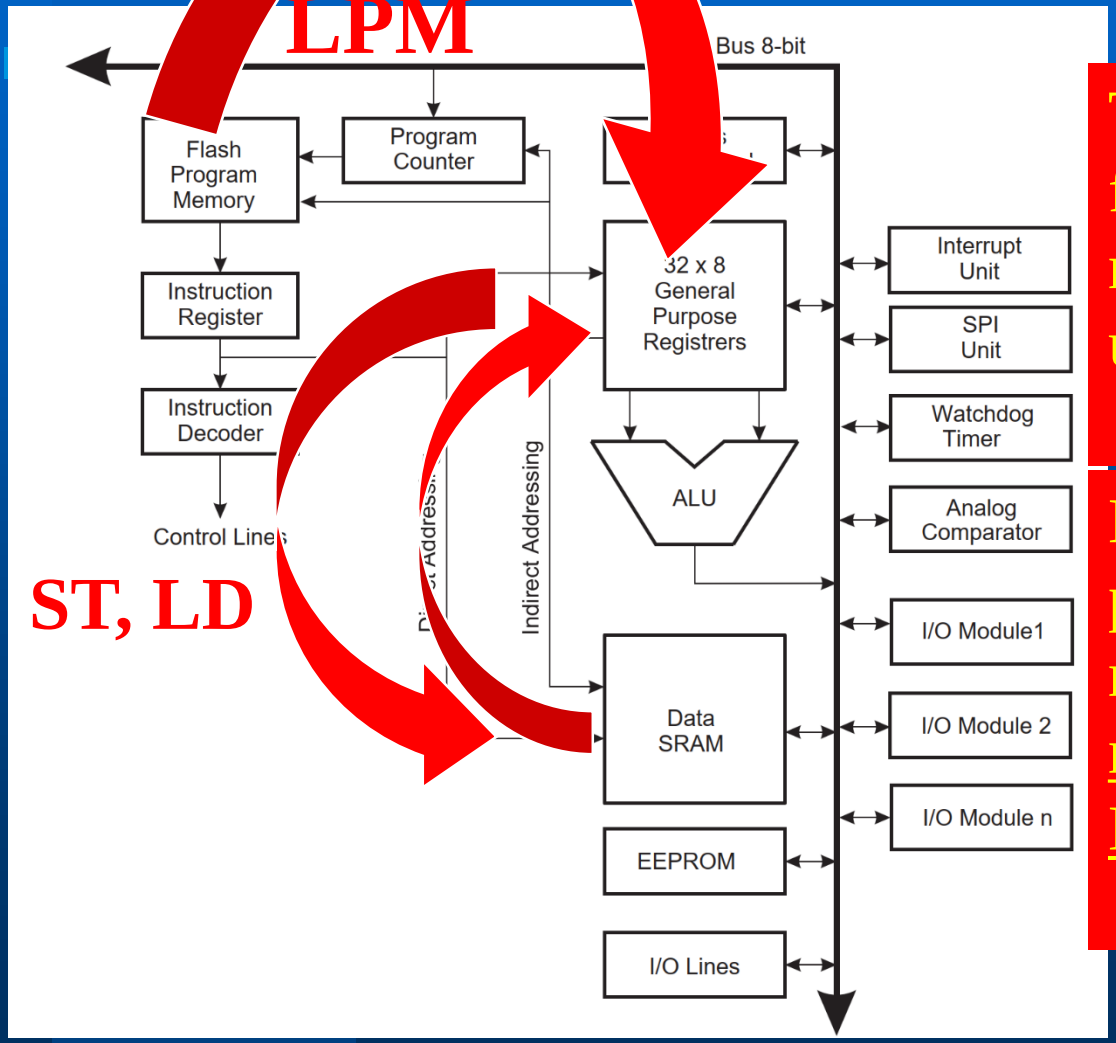
- **To store the register contents to the memory use :**

```
ST  X,  Rx     ST Y, Rx        ST Z, Rx

ST  X+,  Rx    ST Y+, Rx       ST Z+, Rx

ST  -X,  Rx    ST -Y, Rx       ST -Z, Rx
```

Pointer Registers IV



This way data can be moved from the data SRAM to the register file and the reverse using LD and ST commands

Data can be moved from the program flash memory to the register file ONLY via the Z-register and by using the LPM command.

Pointer Registers V

- **Special command which moves data from the program memory at address Z to register r0 :**

```
LPM
```

SRAM testing program:

Homework 2: **Write a program that:**

1) First, writes incrementing numbers between 0 – FF to the SRAM.

2) Next it should read them back, one by one and compare with what you expect to read. If it is not correct it should send an $55 pattern to the PORTB LEDs and if it is correct it should send an $AA pattern in the PORTB LEDs. Each time you should 'clear' the pattern by sending $FF to PORTB after each test and wait for 1 second.

Hint: Before you do this find out from the documentation at what address does the SRAM data space starts.

Tables in the Program Memory

- **Convenient way to store a data pattern in the program memory:**

```
MyTable:
.DW $0100, $0302, $0504, $0706, $0908      ;Table loading with 16 bit words
.DW $0B0A, $0D0C, $0F0E, $1110, $1312  ;Next 5 16-bit words in the table
.DW $1514, $1617, $1918, $1B1A, $1D1C  ;Next 5 16-bit words in the table
.DW $1F1E                                  ;Last word in the table
```

Accessing Program Memory I

- **Access the table :**

```
MyTable:
.DW $0100, $0302, $0504, $0706, $0908      ;Table loading with 16 bit words
.DW $0B0A, $0D0C, $0F0E, $1110, $1312  ;Next 5 16-bit words in the table
.DW $1514, $1617, $1918, $1B1A, $1D1C  ;Next 5 16-bit words in the table
.DW $1F1E                                  ;Last word in the table
```

- **Need to find out the address of this table in the microprocessors program memory !**

Accessing Program Memory II

- **To calculate the address use:**

```
Start:
        ldi  ZH, HIGH(MyTable*2)      ; load the Table address to Z (High Byte)
        ldi  ZL, LOW(MyTable*2)       ; load the Table address to Z (Low Byte)
        SUBI ZL, $01
        rjmp EndStart
EndStart:     ret
```

Accessing Program Memory III

- **Use then LPM to bring the contains of Z to r0 :**

```
ADIW ZL, 01          ; point to the table address
LPM                  ; bring the low byte to r0
MOV r24, r0          ; move the result to register 24
ADIW ZL, 01          ; now increment the address by one
                     ;to get the high byte
LPM                  ;bring high byte to r0
MOV r25, r0          ; put it in register 25
```

Exercise

- Write a program that reads the contents of a table in the memory and sends the output to PORTB.
- Use subroutines to organize your program.
- Use delays to see the different patterns flashing as they light the LEDS of PORTB.

Homework 3

Write a program that stores the sequence 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F to the program flash memory as a table and then access the table and sends them to PORTB LEDs every 2 sec.