DATA TRANSFER INSTRUCTIONS

MOV	Rd, Rr	Move Between Registers	Rd ← Rr
MOVW	Rd, Rr	Copy Register Word	Rd+1:Rd ← Rr+1:Rr
LDI	Rd, K	Load Immediate	Rd ← K
LD	Rd, X	Load Indirect	Rd ← (X)
LD	Rd, X+	Load Indirect and Post-Inc.	Rd ← (X), X ← X + 1

LDS	Rd, k	Load Direct from SRAM	Rd ← (k)

STS	k, Rr	Store Direct to SRAM	(k) ← Rr

Some more instruction

IN	Rd, P	In Port	Rd ← P
OUT	P, Rr	Out Port	P ← Rr
PUSH	Rr	Push Register on Stack	STACK ← Rr
POP	Rd	Pop Register from Stack	Rd ← STACK

BIT AND BIT-TEST INSTRUCTIONS

SBI	P,b	Set Bit in I/O Register	I/O(P,b) ← 1
CBI	P,b	Clear Bit in I/O Register	I/O(P,b) ← 0
LSL	Rd	Logical Shift Left	Rd(n+1) ← Rd(n), Rd(0) ← 0
LSR	Rd	Logical Shift Right	Rd(n) ← Rd(n+1), Rd(7) ← 0
ROL	Rd	Rotate Left Through Carry	Rd(0)←C,Rd(n+1)← Rd(n),C←Rd(7)
ROR	Rd	Rotate Right Through Carry	Rd(7)←C,Rd(n)← Rd(n+1),C←Rd(0)
ASR	Rd	Arithmetic Shift Right	Rd(n) ← Rd(n+1), n=0..6
SWAP	Rd	Swap Nibbles	Rd(3..0)←Rd(7..4),Rd(7..4)←Rd(3..0)

SEI		Global Interrupt Enable
CLI		Global Interrupt Disable

macros

- Macros are a good way to make code more readable, for example if it contains code that is often reused or if a lot of 16-bit calculations are done.
- Macros in AVR assembler can be defined everywhere in the code as long as they're not used at a location before the macro definition

for loops?

```
        ldi r16, 0              ;clear the register
loop:   out PortB, r16          ;write regioster to port b
        inc r16                 ;inc the counter
        cpi r16, 10             ;compare with 10
        brne loop               ;if less than 10 loop

        ldi r16, 0
loop1:  inc r16                 ;this code differs slightly
        out PortB, r16          ;find it
        cpi r16, 10
        brne loop1
```

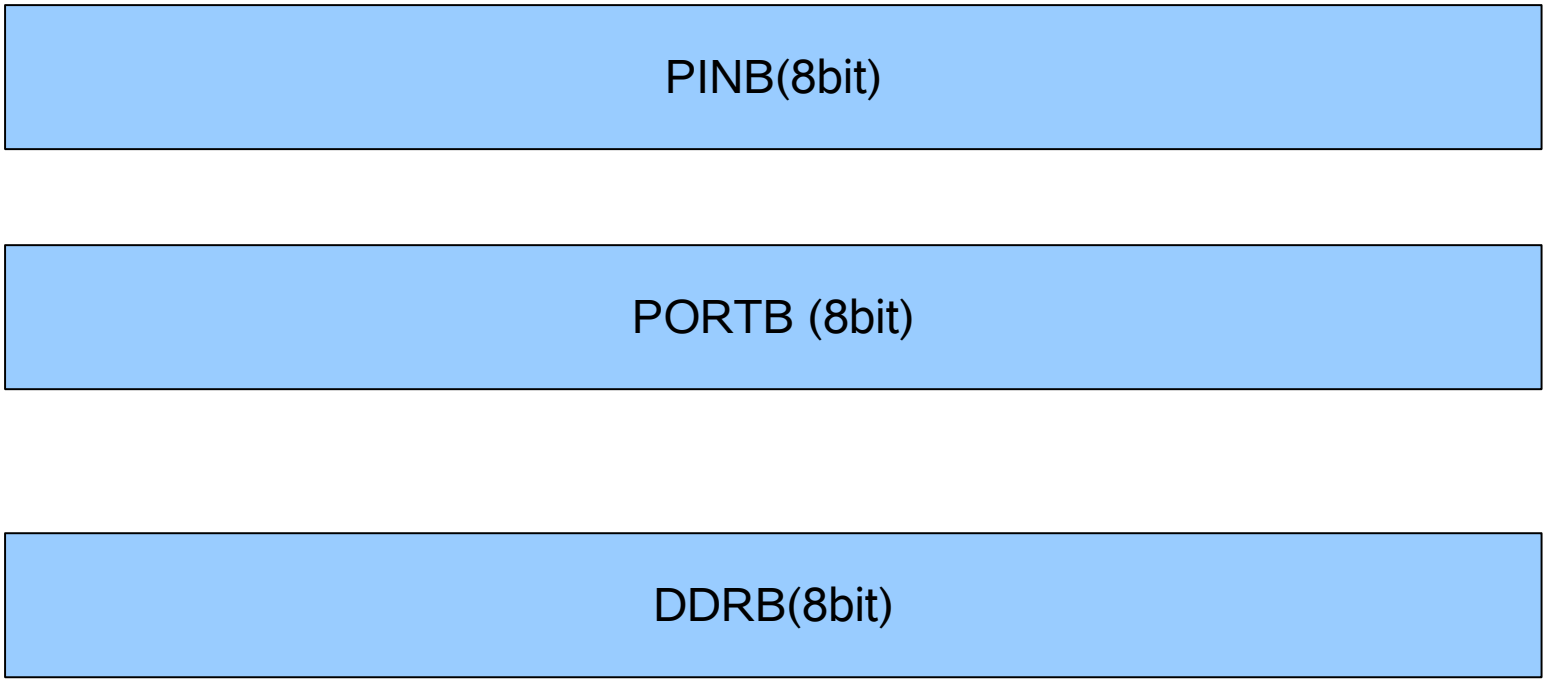
another assignment>>

- make code for WHILE (true) { } ;
- and DO { }WHILE(true)

i/o ports

- Atmega8 is having total 3 i/o ports B-D
- All the ports are bi/directional
- Port B ,PortD are 8bit and portC is 7 bit
- The AVR I/O Ports are pretty simple to understand. They have a Port register, a Data Direction register and a Pin register. These are part of every I/O port in every AVR.

I/o REGISTERS



i/o instructions

- The simplest I/O instructions are in and out.
- in : reads the value of an I/O Port into a register.Example: in r16, PinB
- out :writes the value of a register to an I/O Port
- but before this something is to be done??
- some settings.

note>>>>

sbi and cbi don't operate on all I/O registers. The same is true for sbic/sbis. These can only be used for "classic" I/O Ports and other peripheral registers with addresses from 0 to 31 (0x00 to 0x1F).

.....

thats all .....

Port diagram