To output to a Port

PORTy = Bxxxxxxxx , where y = D|B|C

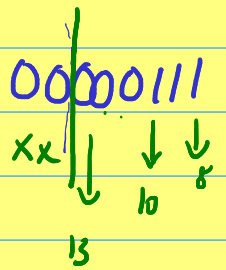
1 => High

0 => Low

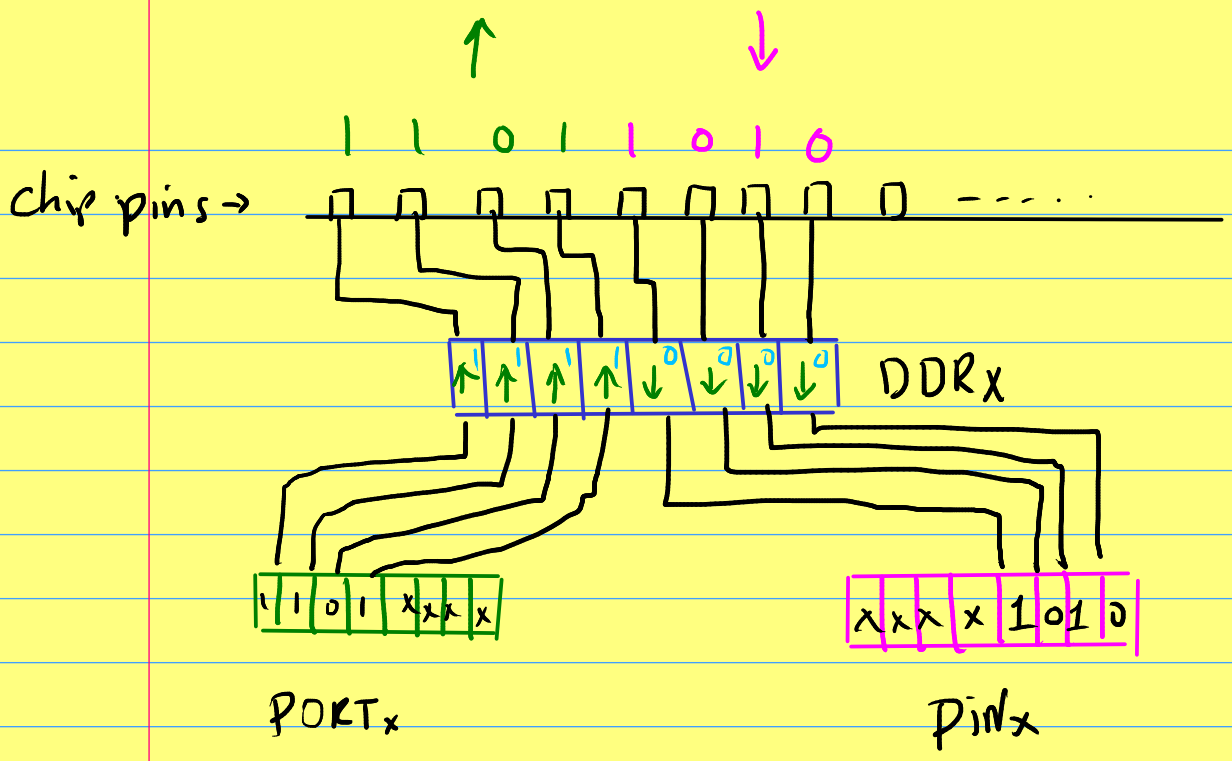
So PORTD = B10101010 will set pin 7 High, pin 6 Low, pin 5 High, etc

Input : uses the PINy , where y = D|B|C

Example: if Pins 8-10 were attached to a logic 1 and Pins 11-13 were attached to logic 0, then the PinB expression would result in 00000111



Char C = PINB;



DDRx = B11110000 ; Where x : D|B|C

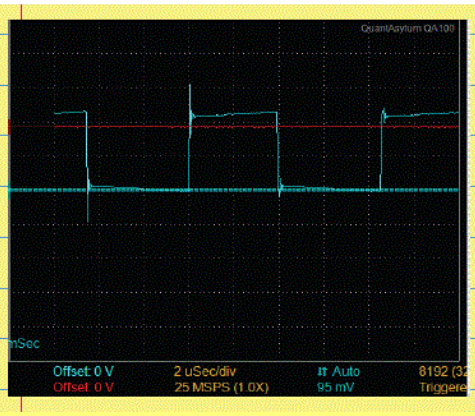
PORTx = 1101xxxx ; where x = don't care

Pinx;

illustrating speed difference when manipulating ports directly:

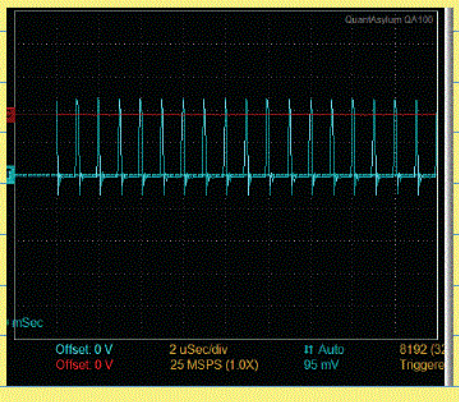
```
void setup()
{
  DDRD = B11111111; // set PORTD (digital 7~0) to outputs
}

void loop()
{
  digitalWrite(7, HIGH);
  digitalWrite(7, LOW);
}
```




```
void setup()
{
  DDRD = B11111111; // set PORTD (digital 7~0) to outputs
}

void loop()
{
  PORTD = B11110000; // digital 4~7 HIGH, digital 3~0 LOW
  PORTD = B00001111; // digital 4~7 LOW, digital 3~0 HIGH
}
```



Bit manipulation Operators

Table 5: Bitwise Operators

Operator	Name	Example	Defined
~	Bitwise complement NOT	~x	Changes 1 bits to 0 and 0 bits to 1
&	Bitwise AND	x&y	Bitwise AND of x and y
|	Bitwise OR	x|y	Bitwise OR of x and y
^	Bitwise exclusive OR	x^y	Bitwise XOR of x and y
<<	Left shift	x<<2	Bits in x shifted left 2 bit positions
>>	Right shift	x>>3	Bits in x shifted right 3 bit positions

$x << y \equiv x * 2^y$

$x >> y \equiv x / 2^y$

$32 = 100000$

$32 >> 1 = 10000$

$= 16$

When a positive value is shifted to the left or to the right, the vacant bits are filled with 0s

When a negative value is shifted to the left or right, vacant bits can be filled with 1s or 0s depending on the implementation

Shifting in 0s is known as a logical shift

" " 1s " " " " arithmetic shift

i.e, it preserves the sign of the number.

Something like 10 >> 50 is unpredictable and implementation dependent

* Bit manipulation can be used to implement masking — allowing you to access a specific bit or group of bit and hiding other

* We can use bit manipulation to reduce the amount of memory needed to store data.