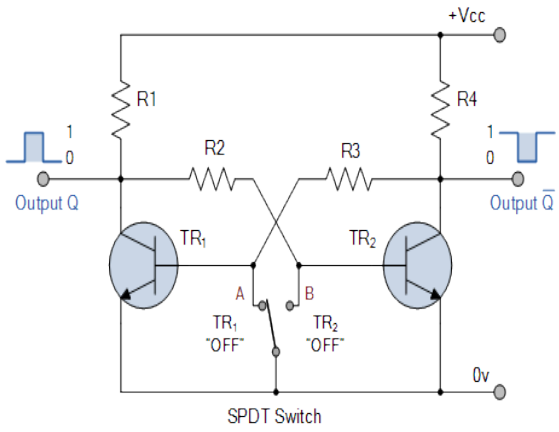
MAR Notes 4 – ATMEGA328P contd….

Memory Structure of ATMega328P - These are the two states for the transistors in the Flip Flop circuit. One transistor is CUT OFF while the other is SATURATED. A transistor that is OFF is said to be in CUT-OFF condition. and one that is fully turned ON is said to be SATURATED. A Flip Flop is the basic memory building block. Types of memory used depends upon the specific purpose of memory and the features of that memory.



1) Memory in a microcontroller is a space where a) Data, b) Program Instructions and c) Control instructions are stored.
2) Memory is always divided in multiple small parts called as memory locations.
3) Every memory location is of same size which will store some data in it.
4) To access the data in the memory, each memory location is allotted an identifying unique binary number called as address.
5) Byte addressable memory – The length of data is 1 Byte only.
6) Word addressable memory – The length of data is a Word of 2 Bytes or 4 Bytes. (Architecture specific)
7) If more than 1 Byte of data is to be stored in B.A.M., then consecutive locations are used to store the data.

- Flash Memory –
- Cache Memory –
- EEPROM –
- SRAM –
- DRAM –
- Virtual Memory –

Types of memories – depends upon the specific purpose of memory.

1) **Main or Primary memory –** contains the program currently being executed by CPU. Moderate speed and Small size.
2) **Associative or Secondary memory –** Programs currently not being used. Slowest but Large size.
3) **Cache memory –** For storage of current and frequently required instructions of the program. Very fast but Smallest in size. Cache is used to reduce the average time to access data from the main memory.

MAR Notes 4 – ATMEGA328P contd….

SRAM (Static RAM)
1) Used as Cache memory
2) Very fast speed
3) Costly
4) Low density
5) Volatile
6) Data is stored in the form of voltage developed between two cross coupled 6 or 4 transistors forming an inverter
7) Periodic refreshing is not required, thus called as Static

DRAM (Dynamic RAM)
1) Used as Main memory
2) Fast
3) Cheaper than SRAM
4) High density
5) Volatile
6) Data is in the form of voltage across a charged Capacitor which gets slowly discharged
7) Thus requires periodic refreshing thus called as Dynamic (few hundred times in a second)

Flash Memory – It is non-volatile advanced type of "simultaneously EEPROM" mostly used in Microcontrollers and other electronics device to store the firmware.

1) Can be quickly erased and programmed, so called as Flash.
2) Performs high speed read, erase and write operations.
3) Has a limited number of erase and write cycles. (10000)
4) Smaller in size. Limited life.
5) Used in Cars, Cameras, Cell phones, Digital diaries etc.
6) Two types – NAND flash and NOR flash

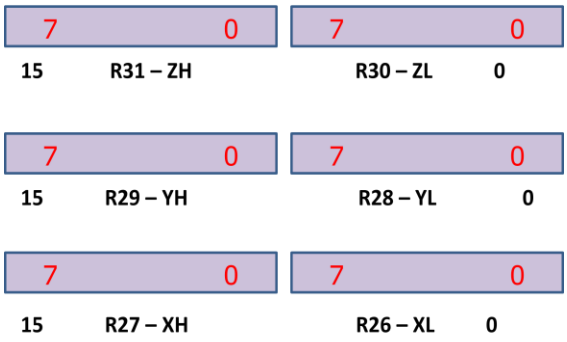
Virtual Memory – Virtual memory is a feature of an operating system (OS) that allows a computer to compensate for shortages of physical memory by temporarily transferring pages of data from random access memory (RAM) to disk storage. Virtual Memory is not included as it is not actual memory.

Register File of General purpose Registers of ATMega328P - 01 to 32 - i.e. 32 locations

Register	Address
R31	0x1F
R30	0x1E
R29	0x1D
R28	0x1C
R27	0x1B
R….	0x….
R….	0x….
R4	0x04
R3	0x03
R2	0x02
R1	0x01
R0	0x00

MAR Notes 4 – ATMEGA328P contd….

- **The X, Y and Z registers of ATMega328P** – can additionally work as 16 bit address pointers for indirect addressing to all the remaining registers in the register file. (R26 to R31)



- **Register File of Special Function Registers**

of ATMega328P – 32 to 95 i.e. next 64 locations

Register	Address
SREG	0x5F
SPH	0x5E
SPL	0x5D
OCRO	0x5C
---	---
---	---
---	---
---	---
TWDR	0x23
TWAR	0x22
TWSR	0x21
TWBR	0x20

Address
0x7FFF
0xFFFE

0x0061
0x0060

- **General Purpose 32 KB Internal RAM of ATMega328P**

MAR Notes 4 – ATMEGA328P contd….

- **Addressing Modes of AVR (ATMega328P) –**

1) Single register direct mode – (Immediate) – deals with the contents of a single register directly.

Format → Instruction Register, (Number)

Register should be between R16 to R31 (R0 to R15 are not allowed)

0 <= Number <= 255 (as 8 bit)

Few examples …..

LDI R18, 0x3C; Load 3C immediately into R18

INC R20; Increment

AND R22, 0b00110101 ANDing of a binary number with contents of R22 and result stored in R22

2) Two register addressing mode – deals with the contents of two registers.

Format → Instruction Register 1, Register 2

Register should be between R16 to R31 (R0 to R15 are not allowed)

0 <= Number <= 255 (as 8 bit)

Few examples …..

ADD R18, R19; Add contents of R19 to R18, result stored in R18

SUB R20, R21; Subtract contents of R21 from R20

MOV R22, R23; Copy contents of R23 to R22

3) Direct addressing mode – deals data between a register and a memory location.

Format → Instruction Register, address

OR Instruction address, Register

Few examples …..

LDS R20, 0x0065 data at memory location 0065H will be loaded into R20

STS 0x0095, R20 Contents of R20 will be stored at memory location 0095H

Question …..

1) LDS 0x0095, 0x0065 → what is meaning of this ?
2) LDI 0x0065, 0x0095 → what is meaning of this ?

Both are illegal, as a *data can not be directly loaded into a memory location. OR data can not be moved from one memory location into another memory location.* It has to be transferred through a register only !!

LDI R22, 0x0065 → what is meaning of this ?

LDS R22, 0x0065 → what is meaning of this ?

LDI R22, 0x0065 → Actual data is the number 0065H which will be moved into R22

LDS R22, 0x0065 → Data is at memory location 0065H which will be moved into R22.

MAR Notes 4 – ATMEGA328P contd….

---

4) Register indirect addressing mode – uses X, Y or Z registers as pointers to indicate the memory location where data is stored.

Example 1 -

LDI XL, 0x40 → 40H is loaded in lower of X

LDI XH, 0x02 → 02H is loaded in higher of X

LD R20, X → (Load indirect) contents of memory location 0240H are copied into R20

Example 1 -

LDI R22, 0x2B	→	??	(Load immediate)
STS 0x0135, R22	→	??	(Store Direct to SRAM)
LDI YL, 0x35	→	??	
LDI YH, 0x01	→	??	
LDS R18, Y	→	??	(Load Direct from SRAM)

Question – What are contents of R18, R22, Y, 0x0135 after execution of all the instructions above ?

What is the actual data ?

Task – How to transport a data of 300 different nos. to 300 consecutive memory locations ?

How to do this ? For this we can use ……

5) A) Register indirect with post-increment –

5) B) Register indirect with pre-decrement –

for e.g.

```
        LDI  R16, 0x20        (counter = 20)
        LDI  R20, 0xFF
        OUT  DDRB, R20        (Port B as output)
        LDI  ZL, 0x90         (Low byte of Z = 90)
        LDI  ZH, 0x00         (Higher byte of Z = 00)
L1:     LD  R20, Z            (Go to location 0090 by Z and copy the data)
        INC  ZL               (increment pointer)
        OUT  PORTB, R20       (contents of R20 sent to Port B)
        DEC  R16              (decrement counter)
        BRNE  L1              (if R16 is not 0, continue the loop – BRNE = Branch if not equal)
```

MAR Notes 4 – ATMEGA328P contd….

---

6) I/O direct addressing mode – This is a Special addressing mode to access the 64 i/o registers in the ATMega328P from 0x0020 to 0x005F. Only IN and OUT instructions are used. To understand the I/O direct addressing mode, we should first know DDR, PORT & PIN registers. There are 3 very important registers used to handle / control the i/o operations of different ports using GPIO pins. They are called as i/o registers.

Data Directions Register – DDRx

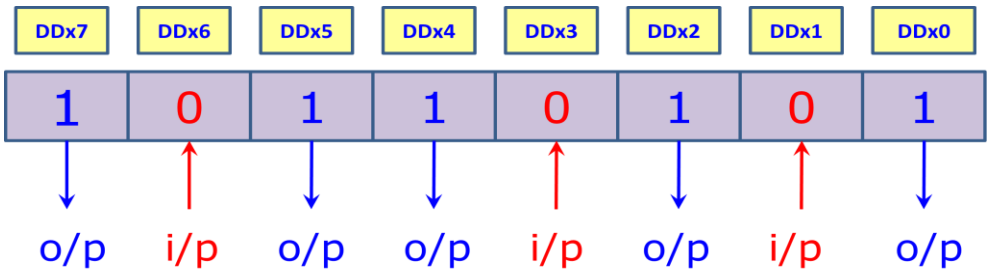
Port Output Register – PORTx

Pin Input Register – PINx …… where x is the name of the port (B, C or D)

Data Directions Register (DDRx) – All the digital pins can work as an input pin or an output pin. DDR will decide whether a pin would be used as i/p pin or o/p pin. If the corresponding bit in the DDR is HIGH (1), the pin will work as OUTPUT pin i.e. eligible to Deliver the data. If the corresponding bit in the DDR is LOW (0), the pin will work as INPUT pin i.e. eligible to Accept the data.

For e.g. Assume DDRD = 0xB5; A Hexa no. B5 is pushed in DDRD OR

DDRD = 0b1011 0101; A binary no. 10110101 is pushed in DDRD



Direction of Data will be decided by DDRD

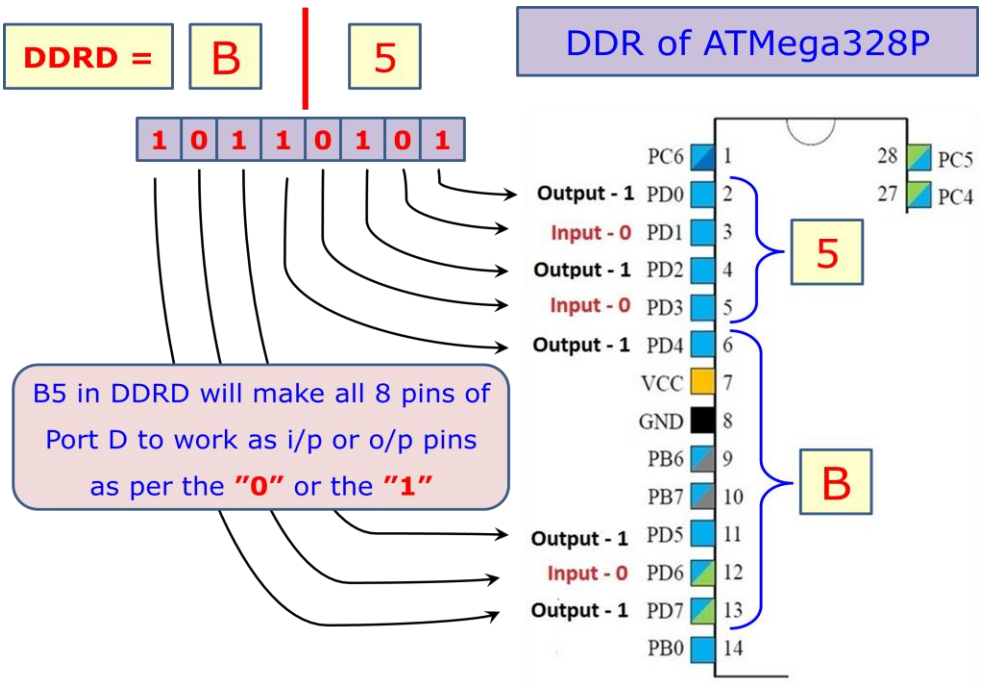
1 = output pin = deliver the data

0 = input pin = accept the data

Question for discussion –

The concepts of output and input are w.r.t. whom ? The Programmer, User, CPU, Pins, MuC or else ?

MAR Notes 4 – ATMEGA328P contd….



PORTx and PINx registers of ATMega328P –

Once the DDR decides whether a pin is configured for outputting the data or inputting the data ……

then the PORTx register decides whether a pin should *deliver* a *HIGH* output or a *LOW* output.

Similarly …….

the PINx register decides whether a pin should *accept* a *HIGH* input or a *LOW* input.

Thus there are total 9 registers as follows ----

DDRB	DDRC	DDRD
PORTB	PORTC	PORTD
PINB	PINC	PIND

For e.g.

DDRD	= 0 b 1 0 1 1 0 0 0 1
PORTD	= 0 b 1 0 0 1 0 0 0 1
OUTPUT =	= 0 b 1 * 0 1 * * * 1

PORTx register **can *not change*** the properties of the pins which are already decided by the DDRx register ….!

MAR Notes 4 – ATMEGA328P contd….

For e.g. 1

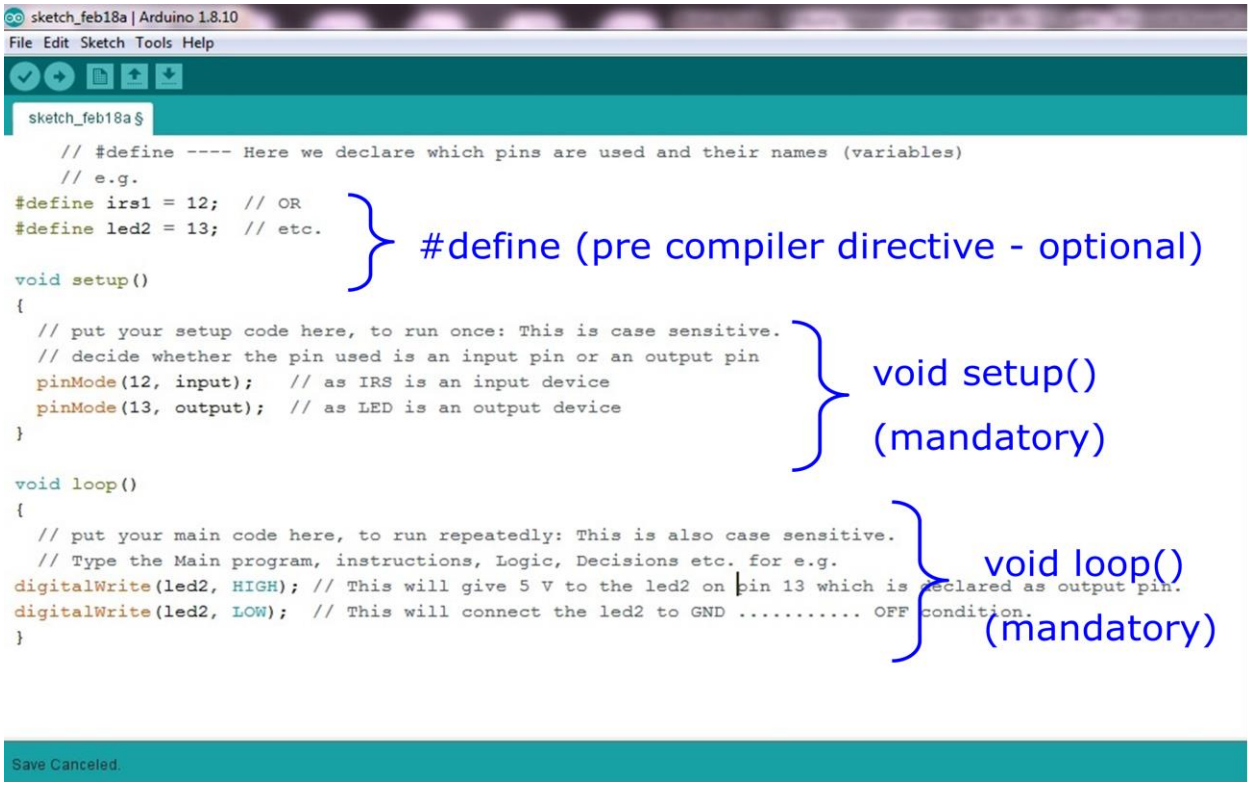
DDRD = 0b 1 0 1 1 0 0 0 1

PIND = 0b 1 0 1 0 1 0 1 0

INPUT = = 0b * 0 * * 1 0 1 *

PINx register **can *not change*** the properties of the pins which are already decided by the DDRx. MuC can read only from those pins which are decided as i/p pins. * pins will enter in Tri state.

Arduino IDE (Integrated Development Environment) –



1) As Arduino is an open platform, many developers all over the world have made and executed innovative programs successfully and are included as examples in public domain by Arduino mentioning developer's name and date with revision.
2) All instructions and syntax are case sensitive.
3) New program is by default named such as sketch_feb15a
4) Sketch (means program) which is executed from top to bottom.
5) void setup() and void loop() are mandatory in any sketch. User made functions are written after void loop()
6) Serial monitor is very important feature to know the real time values.
7) Shields are open source expansion adapter boards that fit on top of Arduino.
8) Writing comments at all important stages is considered as a good programming practice and is strictly followed in industry.

MAR Notes 4 – ATMEGA328P contd….

Important instructions frequently required in Arduino programming –

pinMode(pin no., INPUT or OUTPUT); // a pin is i/p or o/p

e.g. pinMode(12, INPUT); // pin no. 12 will work as input pin

digitalWrite(Pin no., HIGH or LOW); // If the pin is selected as Output pin using pinMode() then, that pin can be set as High (5V) or Low (0V)

digitalRead(pin no.); // reads whether a High or a Low from a digital pin.

e.g. value = digitalRead(irsPin);

analogWrite(pin, value); // the pin will generate a steady square wave of the specified duty cycle (PWM) and will write that analog value to the pin. e.g. to run a motor at various speeds or change brightness etc. Value is between 0 to 255. 0 for 0 V and 255 for 5 V. Frequency is 490 or 980 Hz.

(Pin no. 3, 5, 6, 9, 10 and 11 are the PWM pins)

Duty Cycle = Ton / T total

analogRead(pin no.); // reads not only the voltage but its value as well. This pin reads a value between 0 to 1023 (total 1024 decimal) for a voltage of 5 V or 3.3 V.

---------- End of MAR Notes 4 ----------