HOLONOMIC ROBOT DRIVE SYSTEM

Sponsor: Maharashtra Institute of Technology, Pune

Academic Year: 2013-14

Project Guide: Prof S. G. Kulkarni

Abstract :

Wheeled mobile robots have good maneuverability that makes them be applied widely in industrial production and people's daily life. Application of such drives in Robotics has also evolved in recent years.

Differential driving is the most common movement but with the special mechanism of omnidirectional wheels, omnidirectional mobile robot performs 3 degree-of-freedom (CDOF) motion on the Two-dimensional plane. It can achieve translation and rotation simultaneously along arbitrary direction. Any kind of motion can be implemented while keeping the pose invariable which employs it a zero turning radius. This project includes an automated navigation system of a four wheel omnidirectional drive system using mecanum wheels. It also implements an algorithm which concentrates on reducing the errors in navigation due the factors internal as well as external factors to the robot.

Drive systems:

Drive System is a mechanism to move a robot autonomously.

A mechanism has general requirements such as:

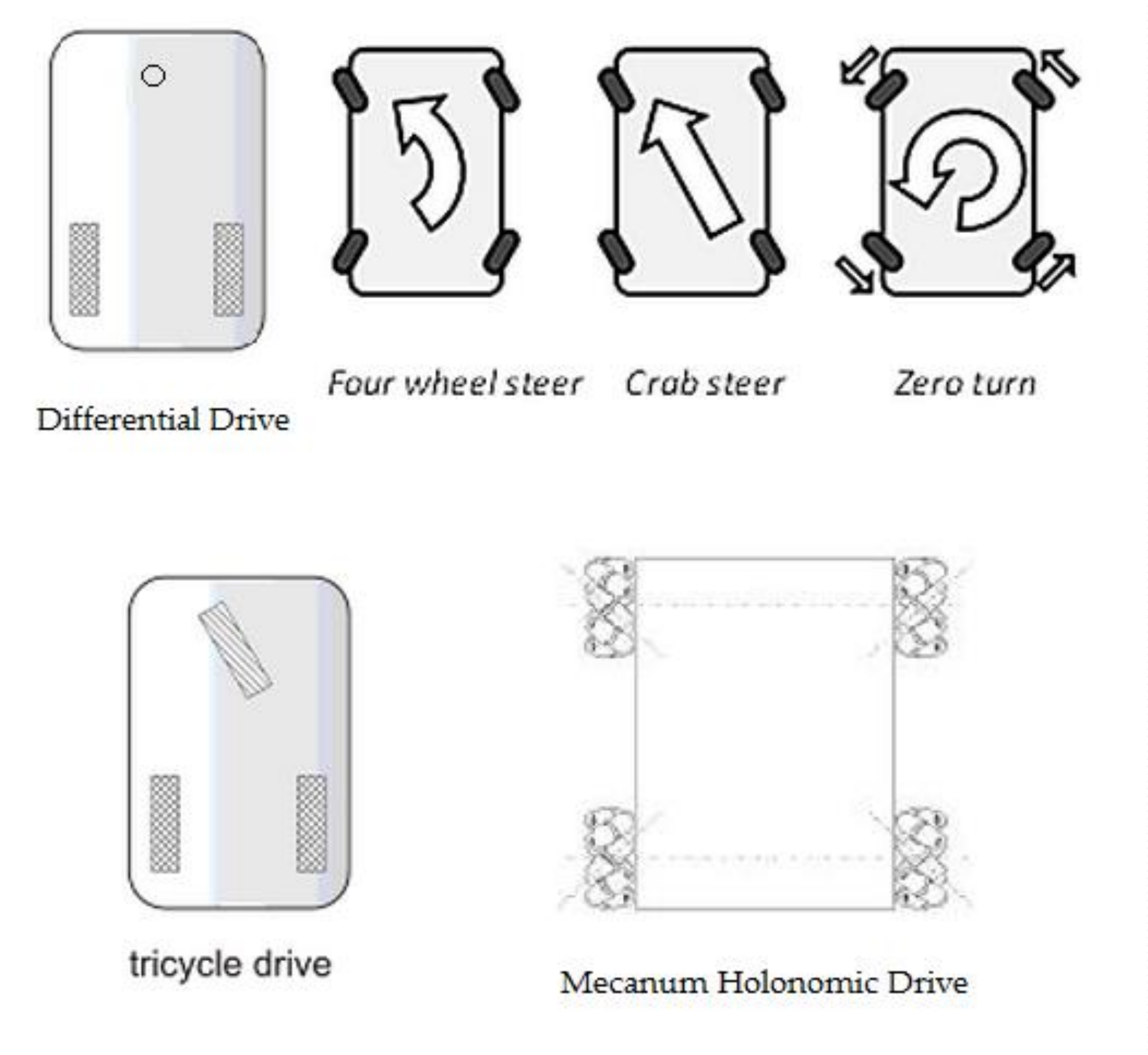
- Good maneuverability
- Faster speed i.e. least time of operation
- Controllable Degrees of Freedom (CDOF)
- Good traction
- Good control over navigation
- Ability to align and position precisely

Types of Driving Mechanisms:

Some standard and widespread driving mechanisms used in robotics are:

- Two wheel drive system (Differential driving)
- Three wheel drive system (Tricycle)
- Steering mechanism
- Holonomic drive system (Omni-directional).

Various Drives



Specification	Differential (2 wheel)	Tricycle (3 wheel)	Steering	Holonomic (omni-directional)
Controllable DOF	2	2	3	3
No. of actuator(Motors)	2	3	5 to 8	4
Maneuverability	Good	Good	Better	Best
Accuracy in positioning	Lesser	Least	Higher	Most accurate
Turning radius	Non zero	non zero	Zero	Zero
Pushing Power	Least	Least	Highest	Higher
Navigational Delays	Yes (parallel parking problem)	Yes (parallel parking problem)	Yes (delay due to wheel turning)	Zero (immediate turning)

What is Holonomic Robot?

A robot is holonomic if:

- Controllable Degrees Of Freedom (CDOF) = Total Degrees of Freedom (DOF) possible in the navigational space (1D, 2D or 3D).
- It can achieve *translation* and *rotation* simultaneously.
- Robot is able to move instantaneously in any direction in the space of its degrees of freedom.

For Holonomic Robot using mecanum :

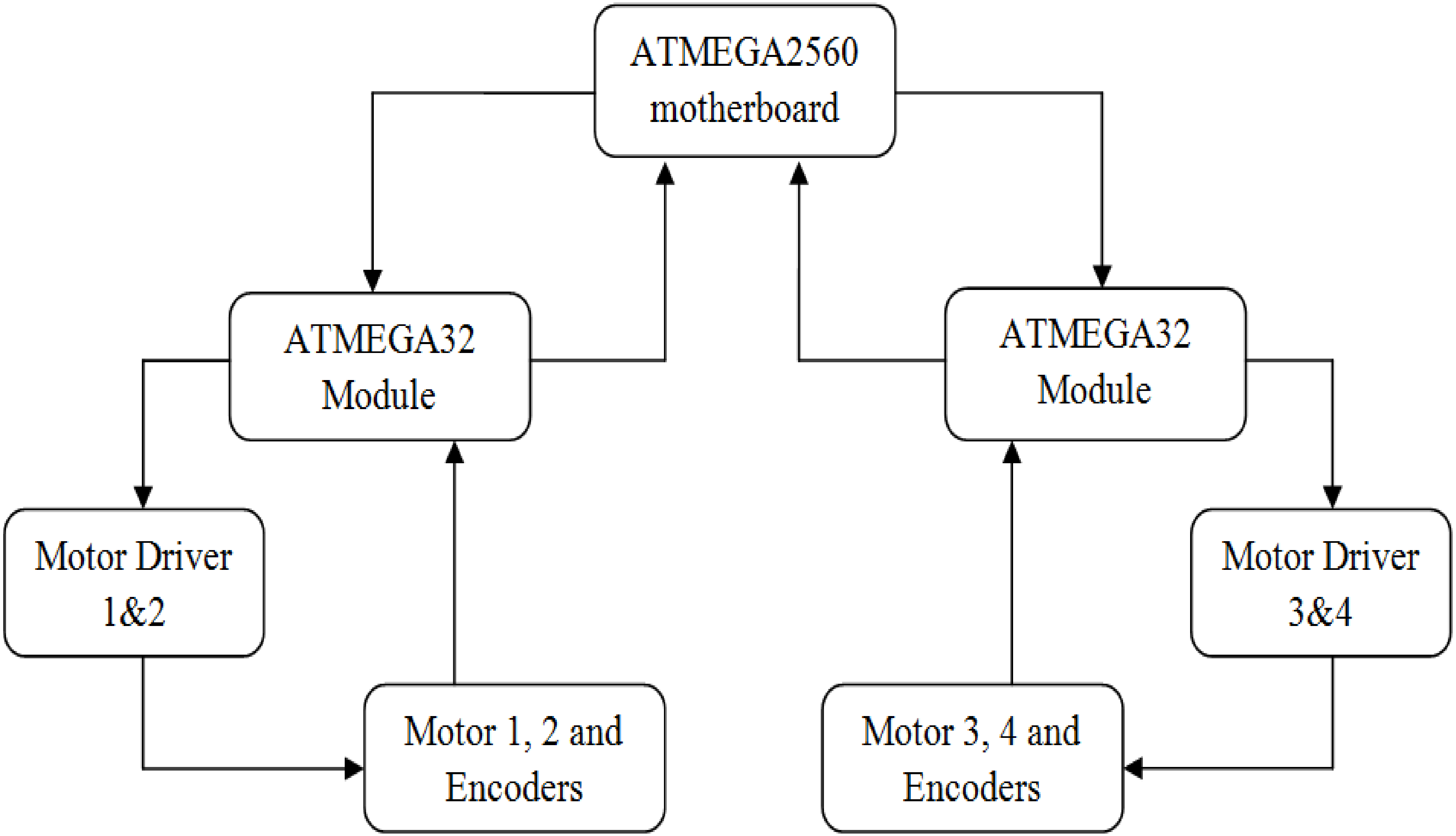
- Total DOF in 2D space are 3 (x, y, theta)
- Total Controllable Degree of Freedom (CDOF) in 2D space are also 3 (x, y, theta)

Mecanum Wheel:

We are using a special wheel called ***Mecanum wheel*** to accomplish the holonomic drive.

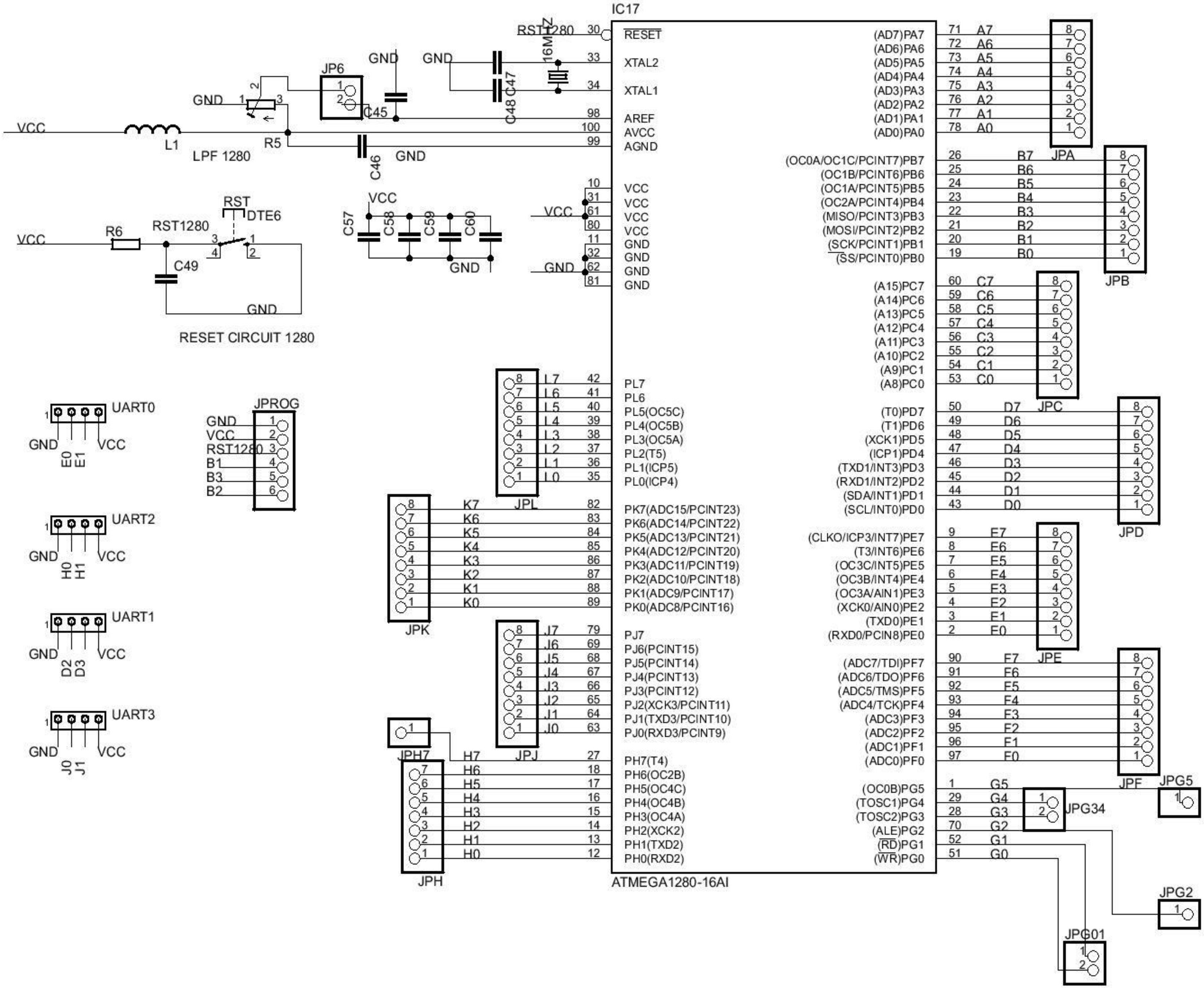
- Mecanum wheels have *free-rolling sub-wheels* called rollers mounted along its periphery
- These rollers make an offset angle of *45 degrees* to the axis of rotation of the wheel
- They allow *sideways motion* of the wheel i.e. motion along the axis of the wheel. This makes the wheel to be able to move in x and y direction simultaneously.

System Block Diagram :



Atmega 2560 Motherboard:

- It functions as a master of all the modules present.
- While going from position A to position B, it calculates the velocity vectors to be given to respective modules.
- Communicates with slave Atmega32 modules using UART protocol
- It is updated with current orientation (angle theta) of the robot by the inertial sensor (IMU).

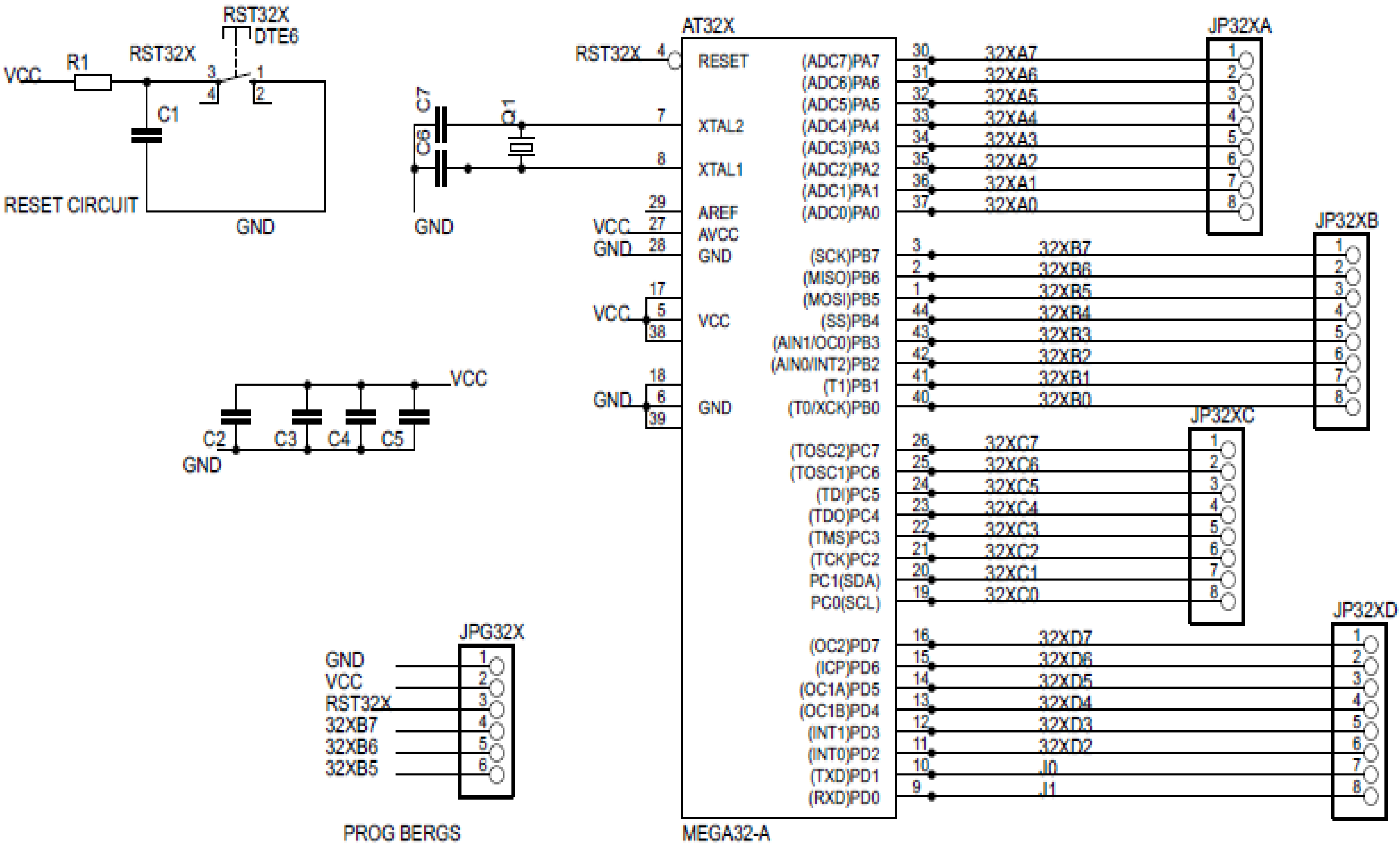

IC17
ATMEGA1280-16AI
RST1280 30 RESET
16MHZ
33 XTAL2
34 XTAL1
98 AREF
100 AVCC
99 AGND
10 VCC
31 VCC
61 VCC
80 VCC
11 GND
32 GND
62 GND
81 GND
GND
JP6
C45
C47
C48
VCC
L1
LPF 1280
R5
C46
RST
DTE6
R6
RST1280
C49
RESET CIRCUIT 1280
C57
C58
C59
C60
UART0
GND E0 E1 VCC
UART2
GND H0 H1 VCC
UART1
GND D2 D3 VCC
UART3
GND J0 J1 VCC
JPROG
GND VCC RST1280 B1 B3 B2
JPL
L7 42 PL7
L6 41 PL6
L5 40 PL5(OC5C)
L4 39 PL4(OC5B)
L3 38 PL3(OC5A)
L2 37 PL2(T5)
L1 36 PL1(ICP5)
L0 35 PL0(ICP4)
JPK
K7 82 PK7(ADC15/PCINT23)
K6 83 PK6(ADC14/PCINT22)
K5 84 PK5(ADC13/PCINT21)
K4 85 PK4(ADC12/PCINT20)
K3 86 PK3(ADC11/PCINT19)
K2 87 PK2(ADC10/PCINT18)
K1 88 PK1(ADC9/PCINT17)
K0 89 PK0(ADC8/PCINT16)
JPJ
J7 79 PJ7
J6 69 PJ6(PCINT15)
J5 68 PJ5(PCINT14)
J4 67 PJ4(PCINT13)
J3 66 PJ3(PCINT12)
J2 65 PJ2(XCK3/PCINT11)
J1 64 PJ1(TXD3/PCINT10)
J0 63 PJ0(RXD3/PCINT9)
JPH7
JPH
H7 27 PH7(T4)
H6 18 PH6(OC2B)
H5 17 PH5(OC4C)
H4 16 PH4(OC4B)
H3 15 PH3(OC4A)
H2 14 PH2(XCK2)
H1 13 PH1(TXD2)
H0 12 PH0(RXD2)
JPA
(AD7)PA7 71 A7
(AD6)PA6 72 A6
(AD5)PA5 73 A5
(AD4)PA4 74 A4
(AD3)PA3 75 A3
(AD2)PA2 76 A2
(AD1)PA1 77 A1
(AD0)PA0 78 A0
JPB
(OC0A/OC1C/PCINT7)PB7 26 B7
(OC1B/PCINT6)PB6 25 B6
(OC1A/PCINT5)PB5 24 B5
(OC2A/PCINT4)PB4 23 B4
(MISO/PCINT3)PB3 22 B3
(MOSI/PCINT2)PB2 21 B2
(SCK/PCINT1)PB1 20 B1
(SS/PCINT0)PB0 19 B0
JPC
(A15)PC7 60 C7
(A14)PC6 59 C6
(A13)PC5 58 C5
(A12)PC4 57 C4
(A11)PC3 56 C3
(A10)PC2 55 C2
(A9)PC1 54 C1
(A8)PC0 53 C0
JPD
(T0)PD7 50 D7
(T1)PD6 49 D6
(XCK1)PD5 48 D5
(ICP1)PD4 47 D4
(TXD1/INT3)PD3 46 D3
(RXD1/INT2)PD2 45 D2
(SDA/INT1)PD1 44 D1
(SCL/INT0)PD0 43 D0
JPE
(CLKO/ICP3/INT7)PE7 9 E7
(T3/INT6)PE6 8 E6
(OC3C/INT5)PE5 7 E5
(OC3B/INT4)PE4 6 E4
(OC3A/AIN1)PE3 5 E3
(XCK0/AIN0)PE2 4 E2
(TXD0)PE1 3 E1
(RXD0/PCIN8)PE0 2 E0
JPF
(ADC7/TDI)PF7 90 F7
(ADC6/TDO)PF6 91 F6
(ADC5/TMS)PF5 92 F5
(ADC4/TCK)PF4 93 F4
(ADC3)PF3 94 F3
(ADC2)PF2 95 F2
(ADC1)PF1 96 F1
(ADC0)PF0 97 F0
(OC0B)PG5 1 G5
(TOSC1)PG4 29 G4
(TOSC2)PG3 28 G3
(ALE)PG2 70 G2
(RD)PG1 52 G1
(WR)PG0 51 G0
JPG5
JPG34
JPG2
JPG01

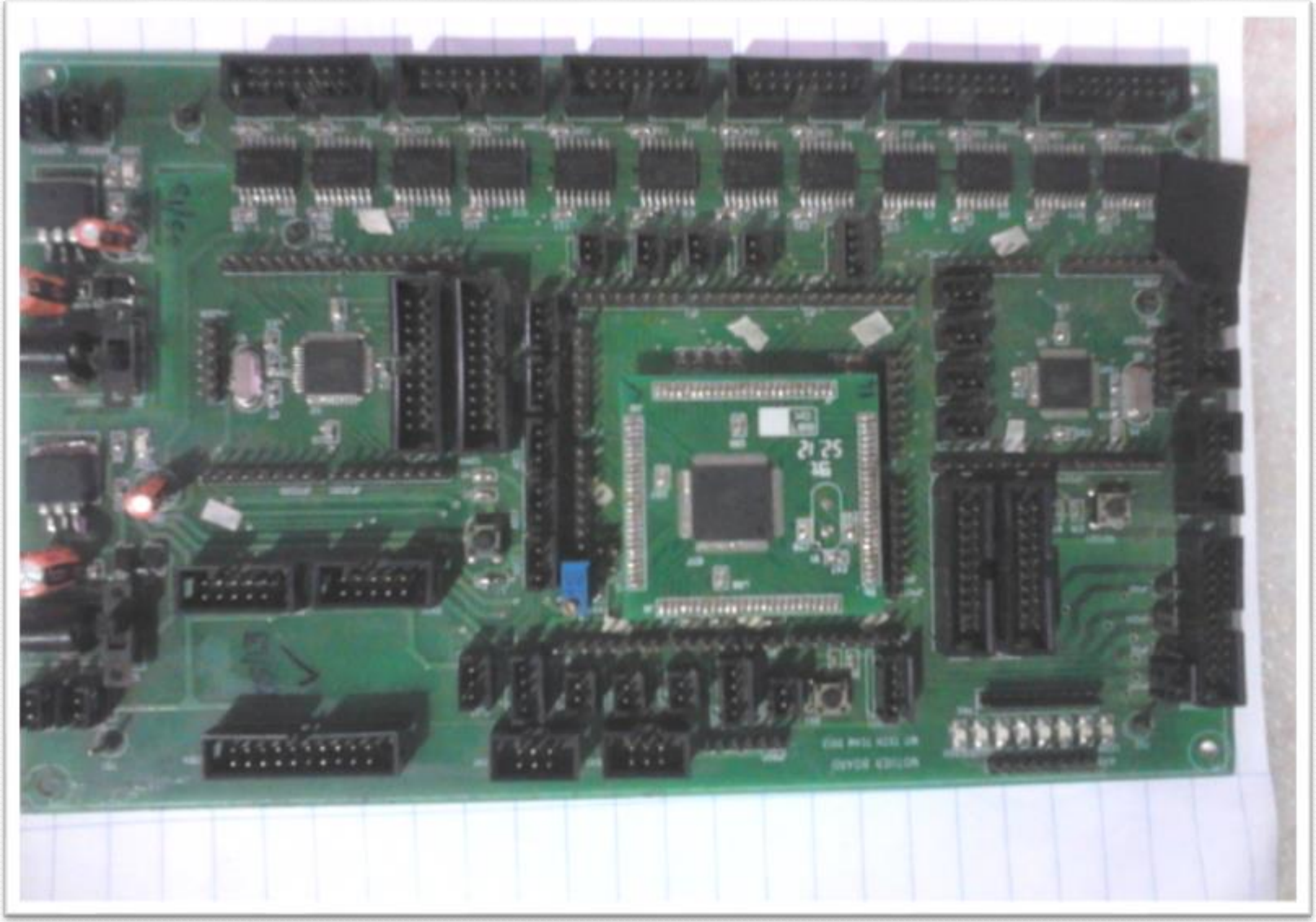
Atmega 32 modules :

Functions are :

- Receiving corrected velocity vectors (V1 through V4) from master controller
- Applying RPM control algorithm for individual motor
- Calculating current velocities using feedback taken from *encoders*
- Sending the current speed in terms of counts from individual motor's encoder to master controller.

ATMEGA 32X


RST32X
DTE6
VCC
R1
RST32X
3
1
4
2
C1
RESET CIRCUIT
GND
C7
Q1
C6
GND
VCC
C2
C3
C4
C5
GND
AT32X
RST32X
RESET
XTAL2
XTAL1
AREF
AVCC
GND
VCC
GND
VCC
GND
VCC
GND
MEGA32-A
(ADC7)PA7
(ADC6)PA6
(ADC5)PA5
(ADC4)PA4
(ADC3)PA3
(ADC2)PA2
(ADC1)PA1
(ADC0)PA0
(SCK)PB7
(MISO)PB6
(MOSI)PB5
(SS)PB4
(AIN1/OC0)PB3
(AIN0/INT2)PB2
(T1)PB1
(T0/XCK)PB0
(TOSC2)PC7
(TOSC1)PC6
(TDI)PC5
(TDO)PC4
(TMS)PC3
(TCK)PC2
PC1(SDA)
PC0(SCL)
(OC2)PD7
(ICP)PD6
(OC1A)PD5
(OC1B)PD4
(INT1)PD3
(INT0)PD2
(TXD)PD1
(RXD)PD0
JP32XA
32XA7
32XA6
32XA5
32XA4
32XA3
32XA2
32XA1
32XA0
JP32XB
32XB7
32XB6
32XB5
32XB4
32XB3
32XB2
32XB1
32XB0
JP32XC
32XC7
32XC6
32XC5
32XC4
32XC3
32XC2
32XC1
32XC0
JP32XD
32XD7
32XD6
32XD5
32XD4
32XD3
32XD2
J0
J1
JPG32X
GND
VCC
RST32X
32XB7
32XB6
32XB5
PROG BERGS



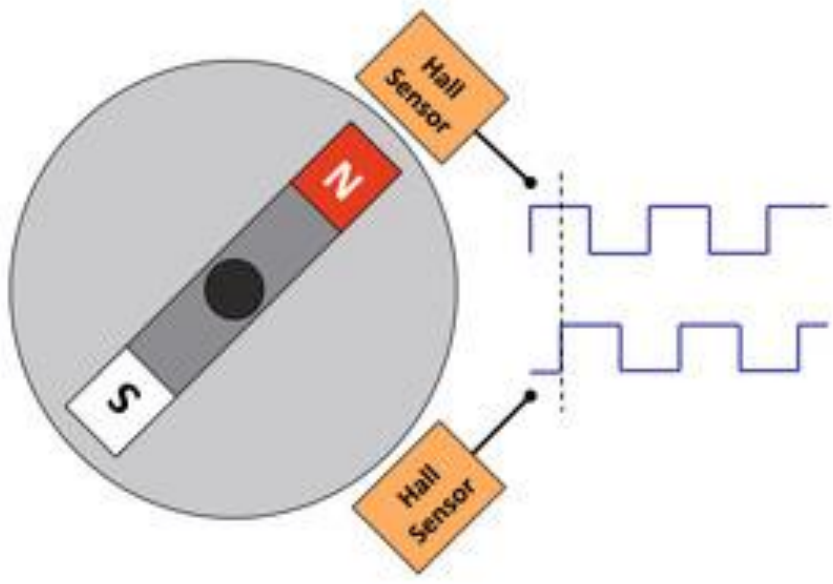
Motors and encoders:

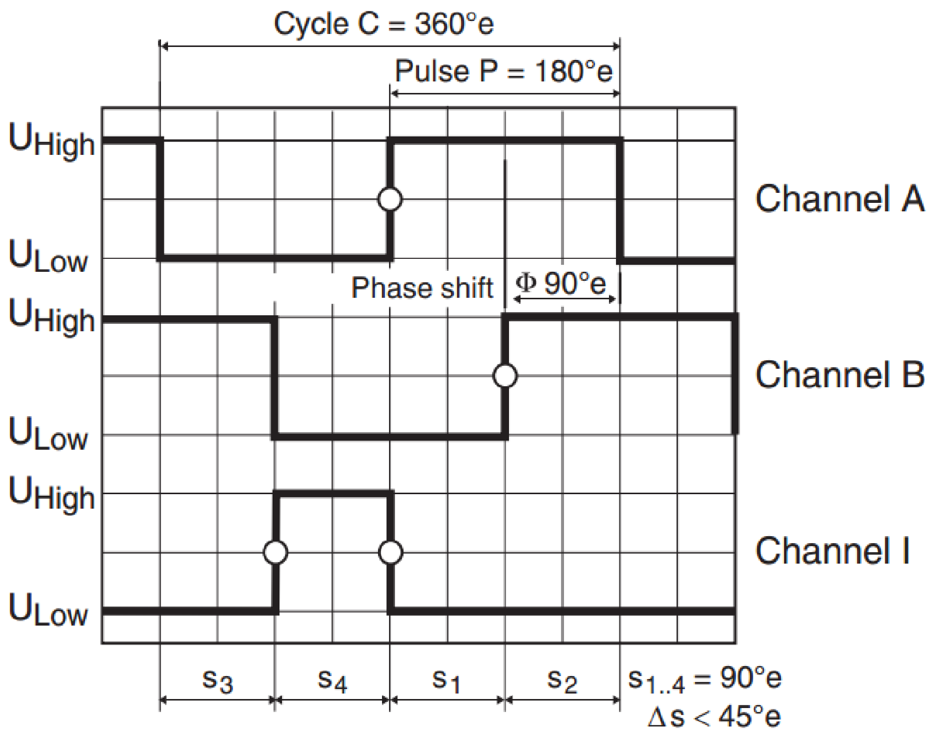
- Four Motors are used each mounted with mecanum wheel
- Feedback of velocities is taken from the respective encoders
- Four *rotary encoders* are used.

ROTARY ENCODER:

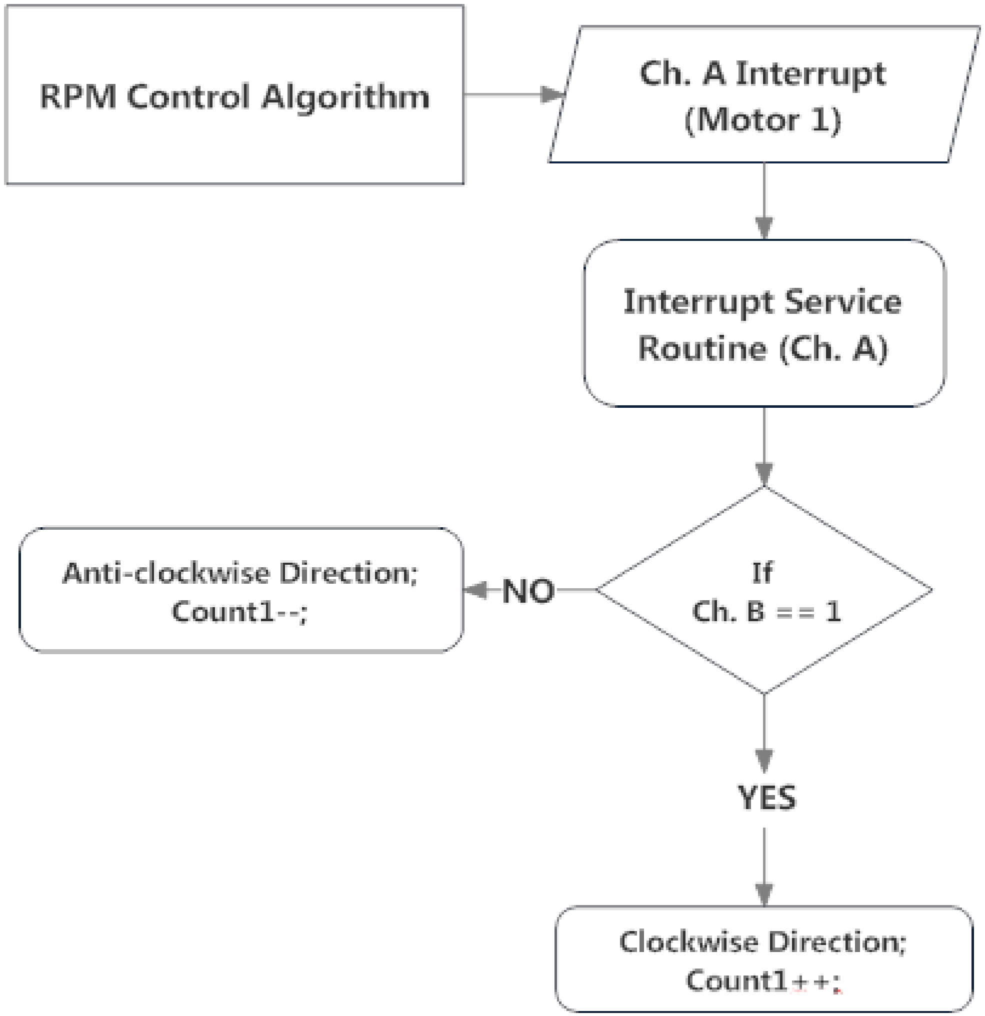
- An encoder is an electromechanical device that is capable of measuring motion or position.
- Encoder has pulse trains are usually 3 types of trains: Channel A, B and Index (I)
- A channel is an electrical output signal from an encoder.
- Channel A and B are 90 degrees phase shifted

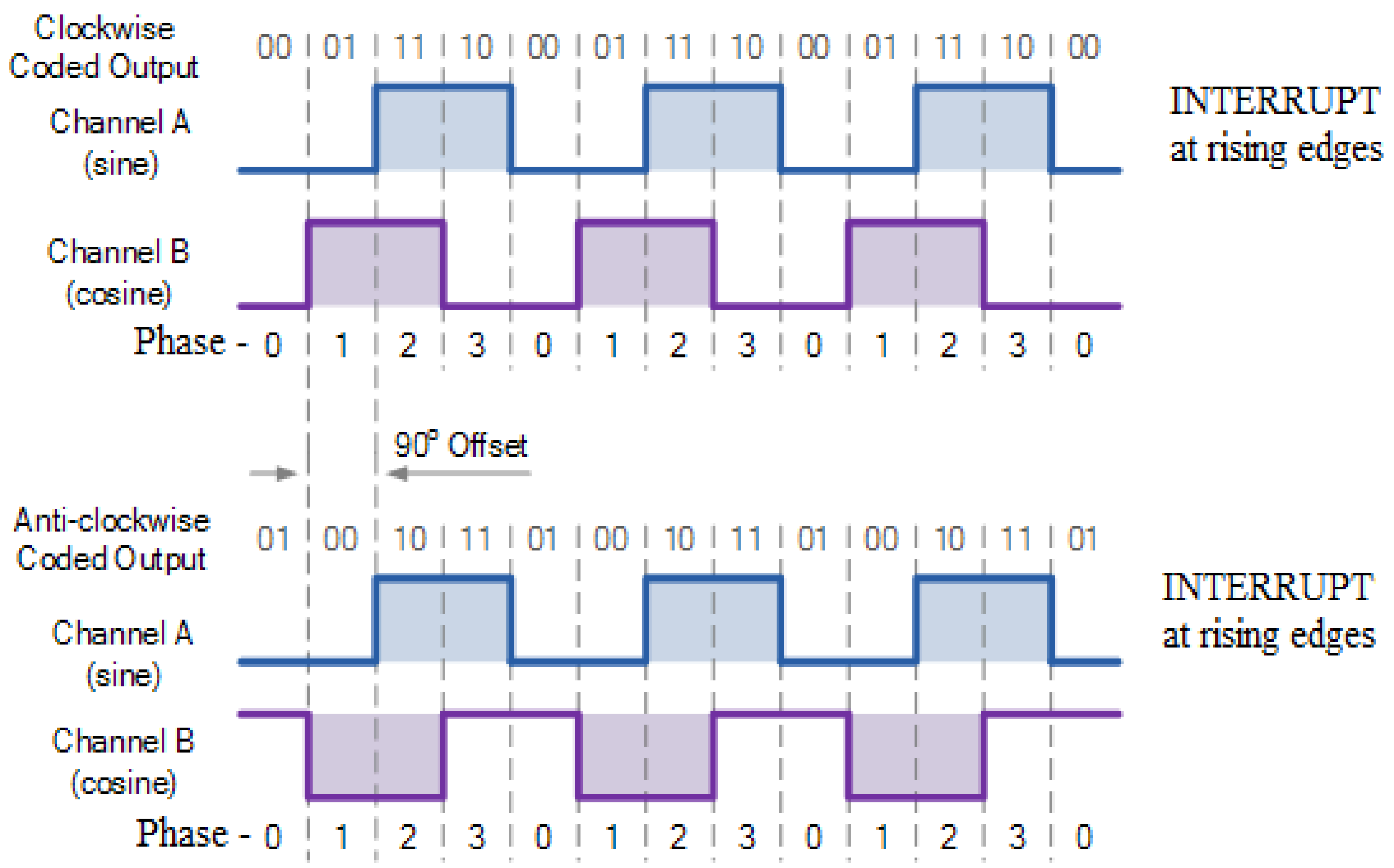
ROTARY ENCODER:





To find Direction using channels:




Clockwise
Coded Output
00 01 11 10 00 01 11 10 00 01 11 10 00
Channel A
(sine)
INTERRUPT
at rising edges
Channel B
(cosine)
Phase - 0 1 2 3 0 1 2 3 0 1 2 3 0
90° Offset
Anti-clockwise
Coded Output
01 00 10 11 01 00 10 11 01 00 10 11 01
Channel A
(sine)
INTERRUPT
at rising edges
Channel B
(cosine)
Phase - 0 1 2 3 0 1 2 3 0 1 2 3 0

Working and Algorithm:

- We have connected channel A of first motor’s encoder to an External Interrupt Pin and Channel B to a General Purpose Input Output (GPIO) Pin of ATMEGA32.

- Whenever a rising edge is detected at the *External Interrupt Pin* of the controller, the code jumps to the *Interrupt Service Routine* written for that hardware interrupt.

- Sample the logic state of corresponding channel B of the same encoder.

- If that comes out to be ***HIGH (5 V) => clockwise direction*** and ***increment*** count by 1.
- On contrary, if the logic state sampled is ***LOW (0 V) => anti-clockwise*** and ***decrement*** the counts by 1

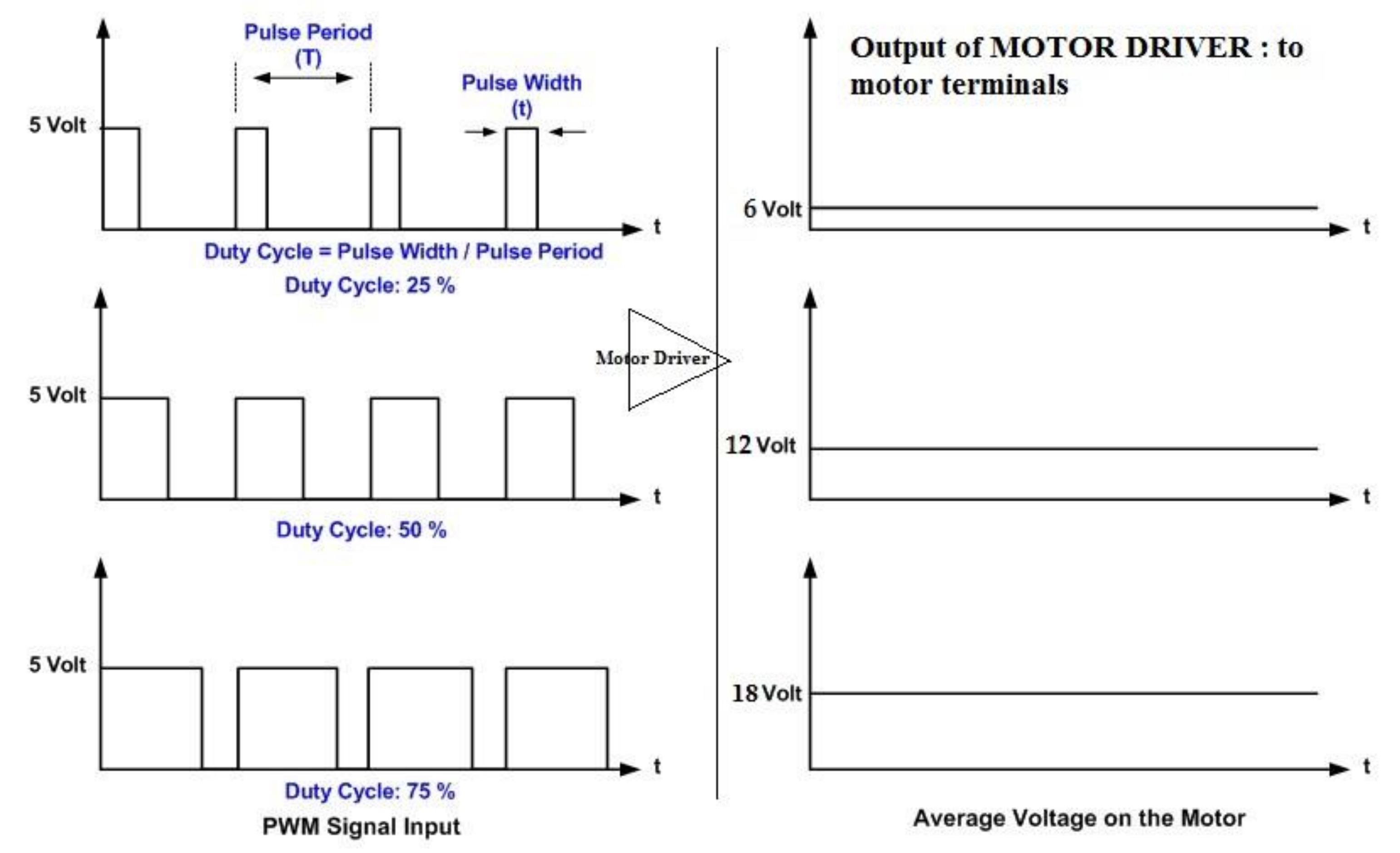
Motor Divers:

- The velocities which are in the form of variable PWM generated by RPM control are applied to the respective motor driver ICs

- It amplifies PWM into 0-24V logic

- They control the voltage given to the motor, which in turn control the speed and direction of each motor.

Function of Motor Driver



PWM Timing Diagram

Electronic specifications

Input Supply:

- Digital Block and Analog Block: 7 to 20 V
- Power Block: 23 to 30 V

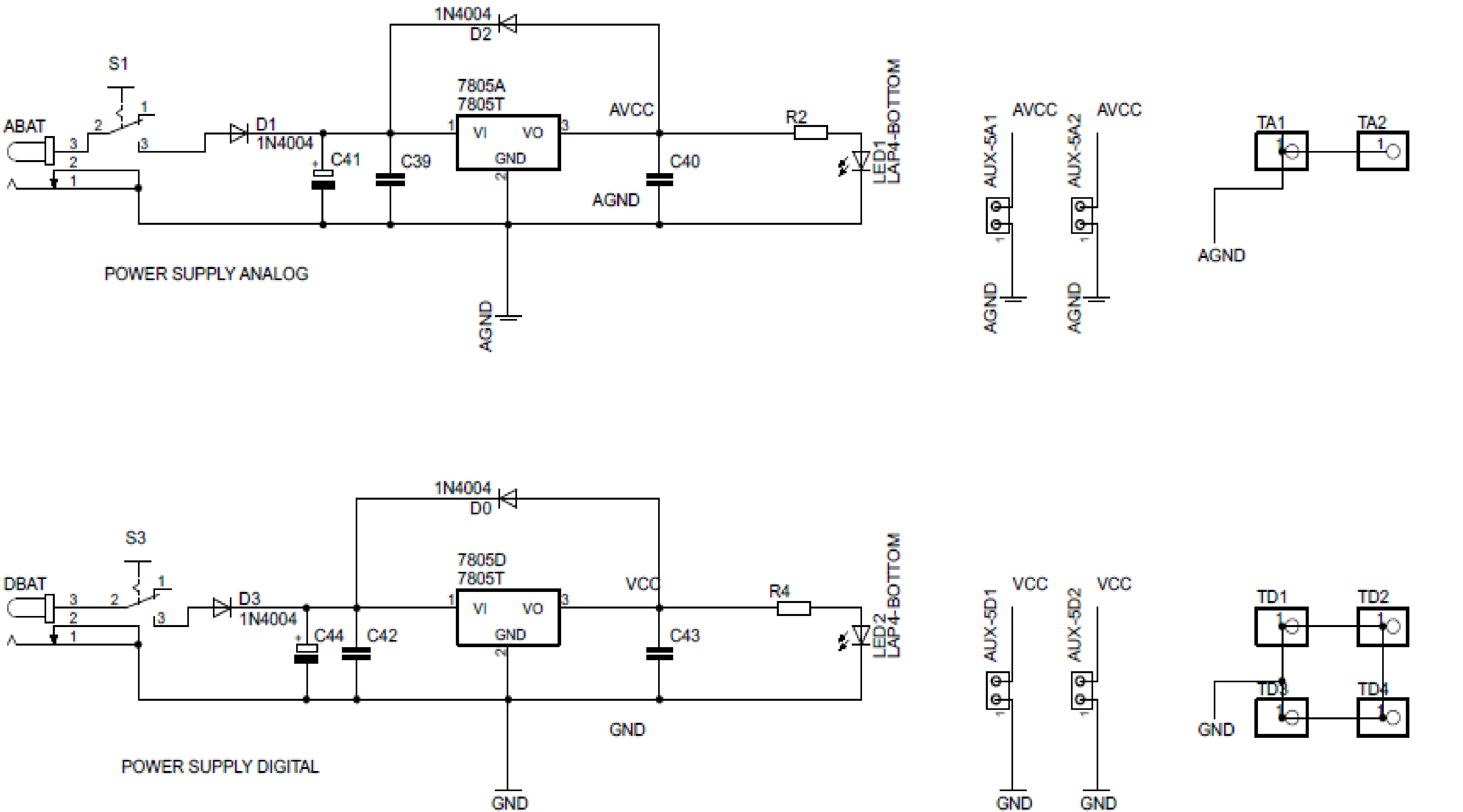
Operating Voltages (Ideal):

- Digital Block and Analog Block: 5V
- Power Block: 24 V

Maximum Power Consumption:

- Digital Block: 0.70 W
- Analog Block: 0.45W
- Power Block: 170W

POWER SUPPLY



General Motherboard Specification:

- No. of Bits of processing: 8 bits and Operating (Oscillator) Frequency: 16MHz
- Maximum Flash Memory: 256 Kb (Master) and 32Kb (Slave)
- Maximum Current by each motor: 2 A
- Encoder Counts: 512 PPR
- Inter IC communication: UART
- Programming: SPI

Motherboard Extendable up to:

- 8 motors (Master controlled)
- Sensor Inputs on 16 ADCs
- 1 UART
- Limit switch interface :12
- 2 external encoders
- 2 parallel ports (8 bit)

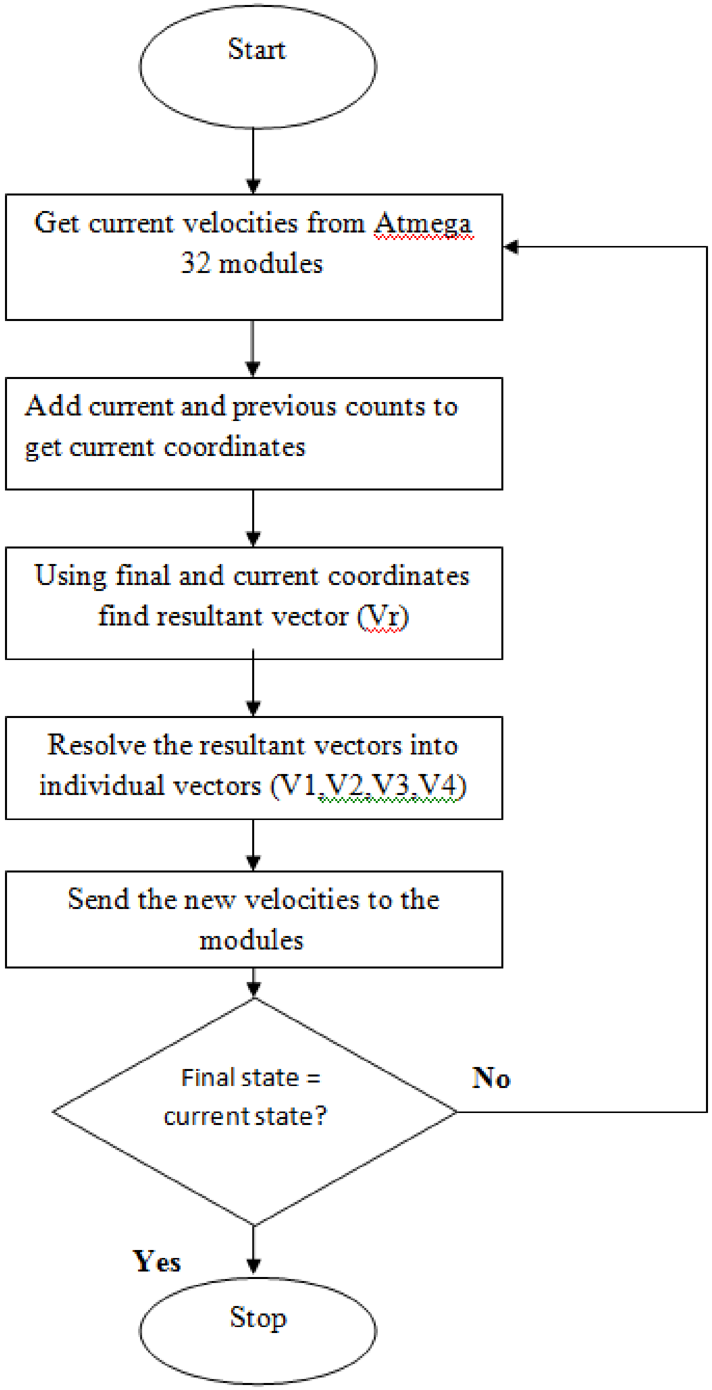
Atmega 2560

- High performance, low power RISC AVR 8 bit microcontroller
- 256 kb programmable flash and 16 MIPS at 16 MHz operation
- Four programmable UART
- Speed : 0 – 16 MHz at 4.5V to 5.5V
- Four 8 bit PWM channels
- 100 pin TQFP package
- -40 to 85 degree (Industrial)

Atmega 32 modules :

- High performance, low power RISC AVR 8 bit microcontroller
- 32 kb programmable flash and 16 MIPS at 16 MHz operation
- one programmable UART
- Speed : 0 – 16 MHz at 4.5V to 5.5V
- Four 8 bit PWM channels
- -40 to 85 degree (Industrial)
- 44 pin TQFP package

Position Control

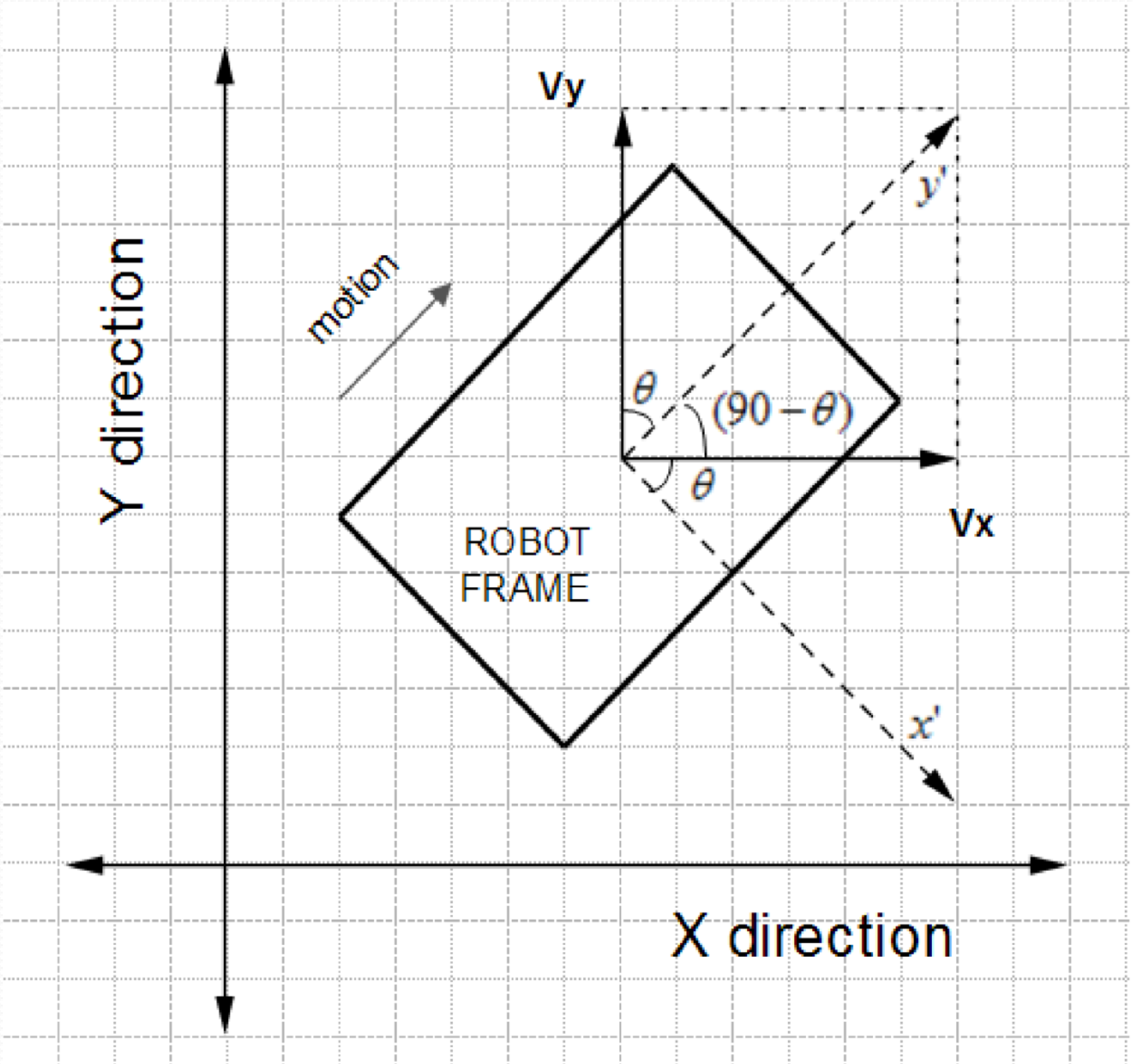


Forward equations :

$$\begin{bmatrix} V1 \\ V2 \\ V3 \\ V4 \end{bmatrix} = \begin{bmatrix} -\sin(45+\theta) & \cos(45+\theta) & 1 \\ -\sin(45-\theta) & -\cos(45-\theta) & 1 \\ \sin(45+\theta) & -\cos(45+\theta) & 1 \\ \sin(45-\theta) & \cos(45-\theta) & 1 \end{bmatrix} * \begin{bmatrix} Vx \\ Vy \\ R\omega \end{bmatrix}$$

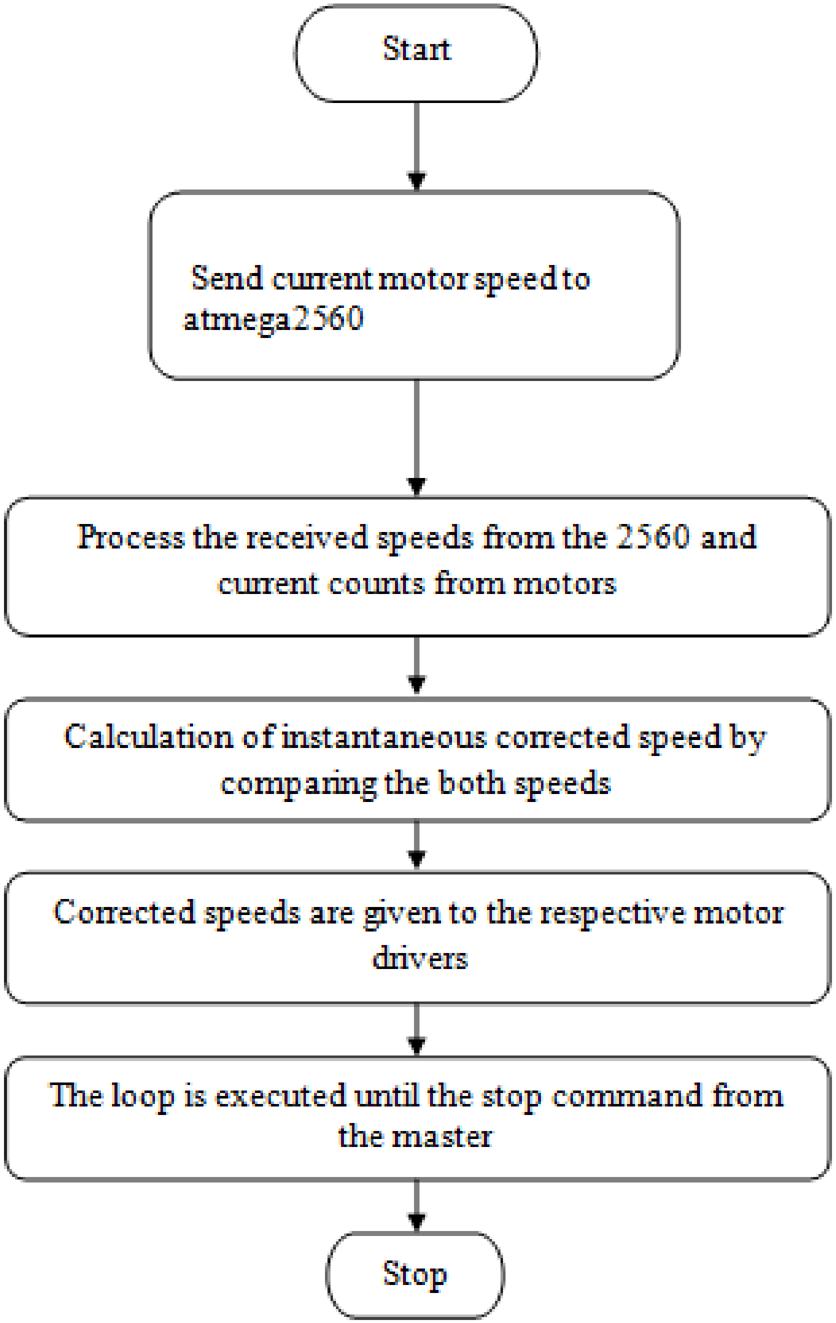
Where ,

- V1, V2, V3, V4 are the required velocities calculated and given to respective motors
- Vx, Vy, Rw are resolved components of in order to final destination
- Theta is the angle of the robot frame w.r.t. its initial position



Vy
y'
motion
Y direction
θ
(90 − θ)
θ
Vx
ROBOT
FRAME
x'
X direction

Algorithm for ATMEGA32 : (RPM control)



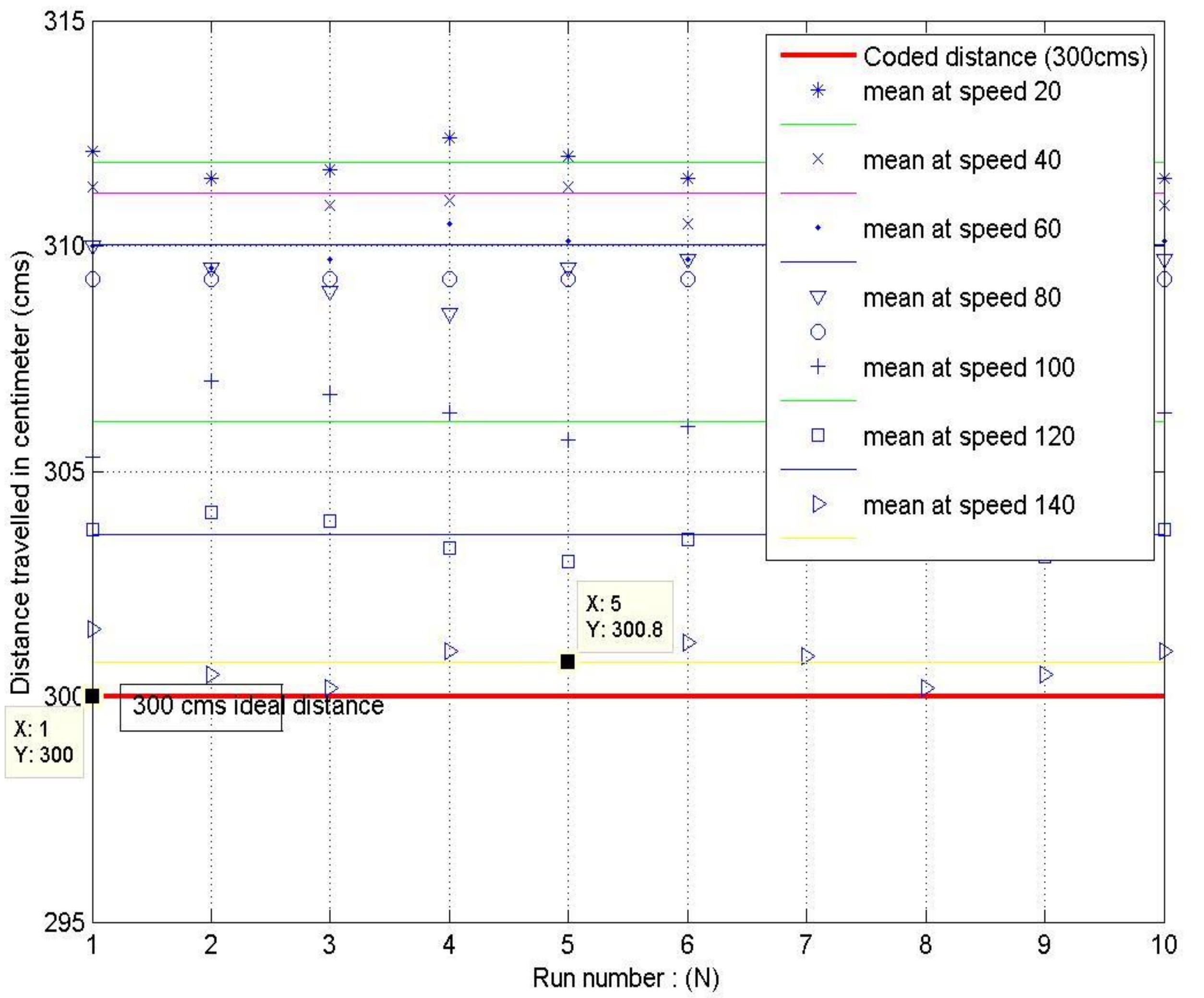
Results

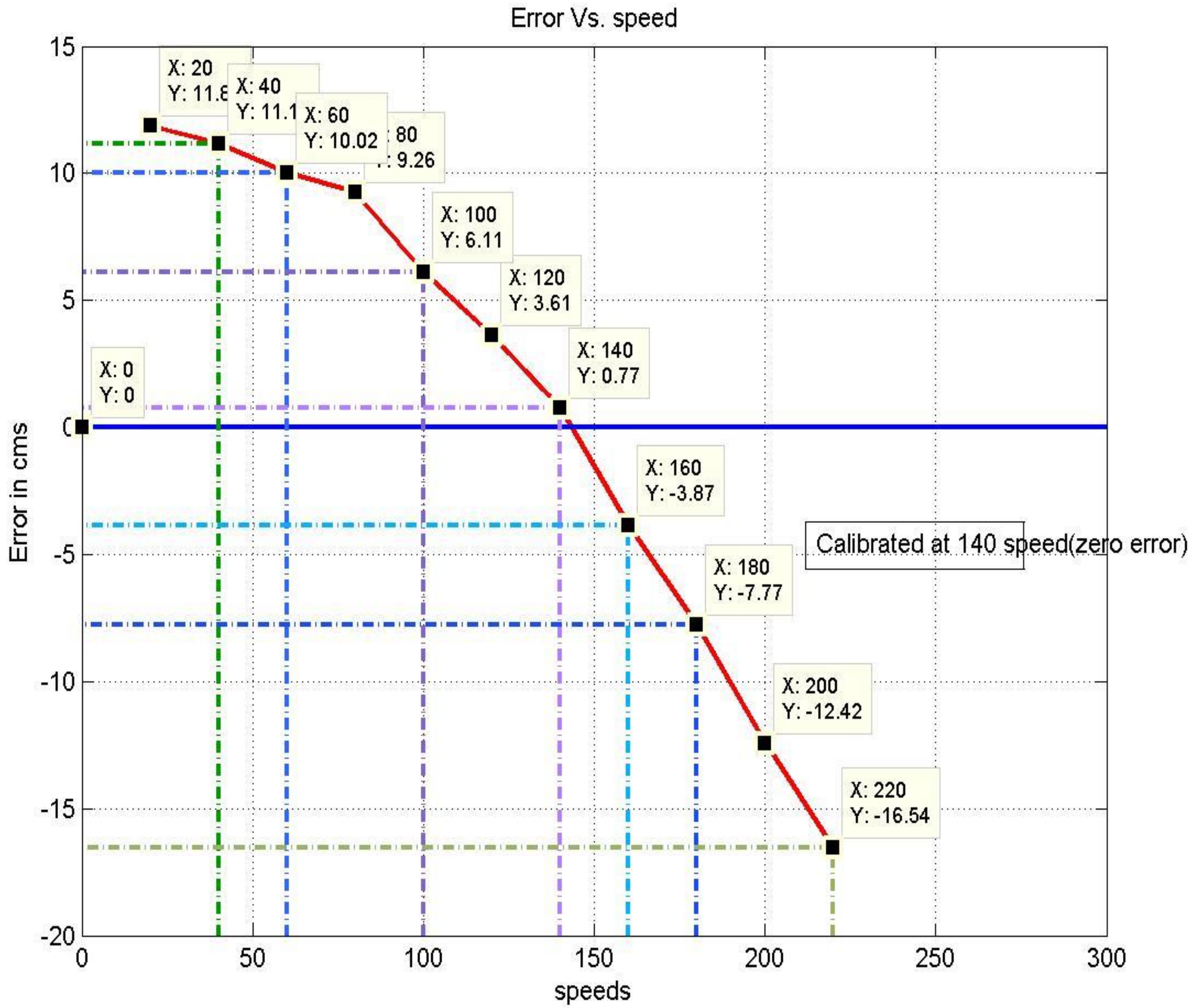
PART 1:

- Closed loop algorithm was implemented at various speeds. Error is the distance reached less the distance coded into the controller. It is possible to reduce the error down to a very small value (< 2cm) for a particular speed, by assigning an appropriate value to calibration factor. In our case we selected the speed of 140 as the calibration speed and made the error less than 1 cm for this speed. Thus, as shown in graph the error at 140 is about 0.9cm.

PART 2:

- Magnitude of error goes on increasing as the speed deviates from its calibrated value (in this case 140). It is verified in the graph that at lower speeds the error increases and reaches up to 12 cm for the speed of 50. While if the speed increases above 140, the error increases in negative direction as depicted in graph.
- Thus, we can conclude that, once the system is calibrated at a particular speed then the error occurred as we deviate from the calibrated value can be predicted using the graph below. The maximum error (which is found after calibration at half way mark approx. 130 / 140) is -5.5 %. Calibration around half way mark gives least percents of error as we deviate from the mark.


Coded distance (300cms)
mean at speed 20
mean at speed 40
mean at speed 60
mean at speed 80
mean at speed 100
mean at speed 120
mean at speed 140
X: 5
Y: 300.8
X: 1
Y: 300
300 cms ideal distance
Distance travelled in centimeter (cms)
Run number : (N)
315
310
305
300
295
1
2
3
4
5
6
7
8
9
10



Error Vs. speed
Error in cms
speeds
15
10
5
0
-5
-10
-15
-20
0
50
100
150
200
250
300
X: 0
Y: 0
X: 20
Y: 11.
X: 40
Y: 11.1
X: 60
Y: 10.02
80
9.26
X: 100
Y: 6.11
X: 120
Y: 3.61
X: 140
Y: 0.77
X: 160
Y: -3.87
X: 180
Y: -7.77
X: 200
Y: -12.42
X: 220
Y: -16.54
Calibrated at 140 speed(zero error)

CONCLUSION

- This system is implemented on a mecanum robot frame that has least mechanical error. Testing in various conditions of surface friction as well as in various paths of navigation gradually showed significant improvement in navigational accuracy as we go on improvising the algorithm.

- The error (difference between applied and actual output) is reduced down to a small value (< 5 percent0 .

- It is further possible to improve the performance by using a high end modular system which uses a dedicated microcontroller for the count calculation process. This method could reduce the number of counts skipped during the count processing iteration.

System requirements:

- ***Software:***
  - ATMEL STUDIO for programming and compiling
- ***Hardware:***
  - ATMEGA 2560 based motherboard
  - ATMEGA 32 modules
  - Basic Robot frame with motors and encoders

References:

[1]M. Udengaard, K.Iagnemma: “Analysis, Design and Control of an Omnidirectional Mobile Robot in Rough Terrain”, Department of Mechanical Engineering, MIT,USA.

[2]R. Rojas, A. G. Forster, “Holonomic Control of a robot with an omnidirectional drive” KI - Kunstliche Intelligenz, Bottcher IT Verlag, 2006.

[3]O. Diegel et al.: “Improved Mecanum Wheel Design for Omni-directional Robots”, Proc. 2002 Australian Conf. on Robotics and Automatio, 2002

[4] S.G. Kulkarni et al. :”Automated high speed omnidirectional navigation using closed loop implementation of four wheel holonomic mecanum drive”, ICERTS India, 2013

[5]D. J. Daniel et al.: “Kinematics and Open-Loop Control of an Ilonator-Based Mobile Platform”, Proc. of the IEEE Int. Conf. on Robotics and Automation, 1985.

Books:

- Mobile Robotics for Multidisciplinary Study By Carlotta A. Berry

Internet:

- www.atmel.com
- www.avrfreaks.com
- www.ni.com
- http://ieeexplore.ieee.org
- http://www.docstoc.com/docs/81669071/Holonomic-Robot

THANK YOU