

n
32

Eta32

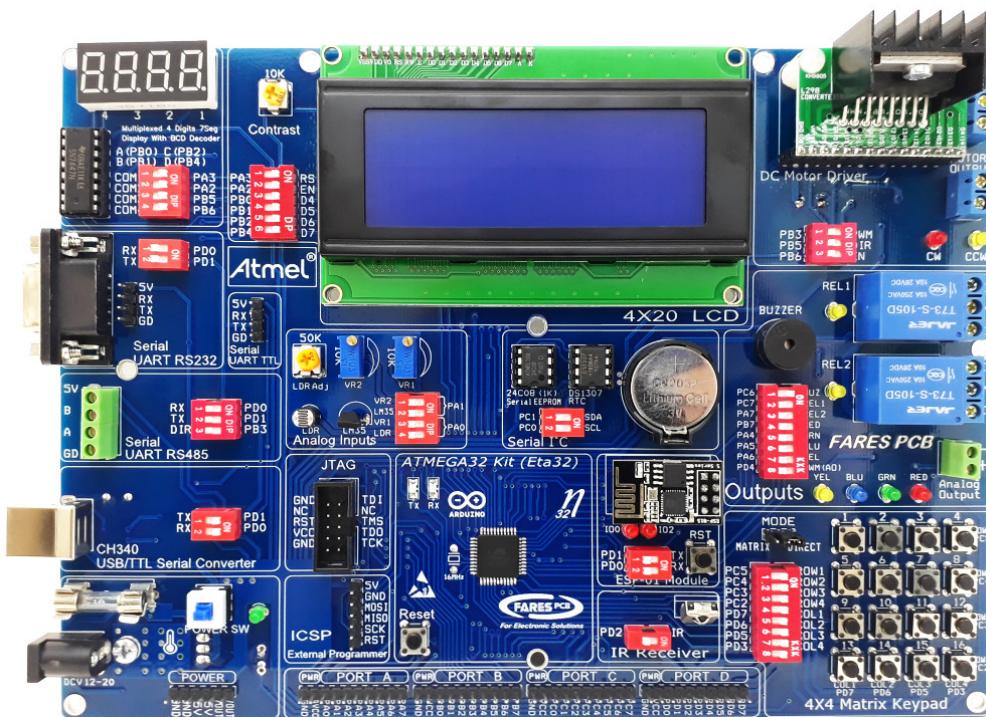
ATMEL AVR Development Kit

General Description

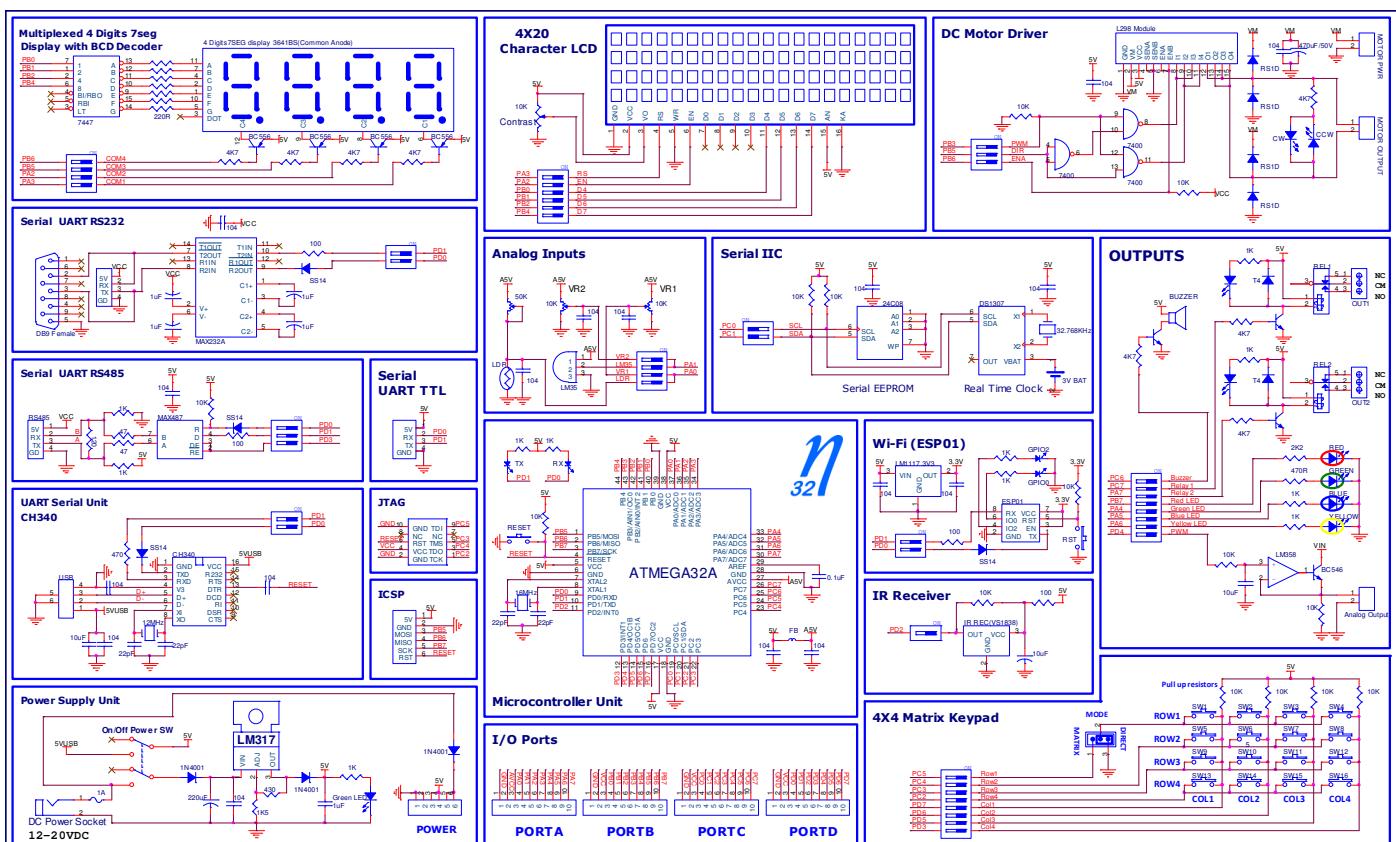
Eta32 kit is an AVR development board designed particularly for students, beginners and recently graduated engineers to provide easy developing of AVR microcontroller projects.

Eta32 supports ATMEGA32 microcontroller chip. **Eta32** kit provides the most common primary devices and circuits, such as LCD, KEYPAD, serial interface, and more. Thus, it reduces time and effort in hardware design and test, hence, developer can focus his efforts on firmware development. All I/O pins are brought out via pin header for direct port accessing.

Eta32 kit comes preprogrammed with UART bootloader, which eliminates the need for external burner where Hex code is downloaded using **Eta Burner** tool from FARESPCB Co. However, kit can be programmed using Arduino IDE (another bootloader). Also, standard 6 pin header socket is included (ICSP) for external programmers.



Eta32 kit



Eta32 kit (schematic)

Eta32 features

- **Power supply unit**
 - 1) Wide input voltage range (12-20V). Also can be powered from USB port.
 - 2) Fuse 1A for high current protection.
 - 3) Reverse polarity protection.
 - 4) ON/OFF switch.
 - 5) Green LED indicator.
- **Microcontroller**
 - 1) ATMEGA32A Microcontroller (44 TQFP package).
 - 2) 16MHz crystal.
 - 3) Reset switch.
 - 4) Two LED indicators for Tx and Rx.
 - 5) Ferrite bead for generating analog VCC.
 - 6) Bypass capacitors on power pins and VREF/GND.
- **Display**
 - 1) 80 characters LCD (4X20) with backlight.
 - 2) 4 Digits (common anode) 7 segment display with BCD decoder 7447.
 - 3) General purpose indicator LEDs (**Red** , **Green** , **Blue** and **Yellow**).
- **Input switches**
 - 1) 16 switches configured as 4X4 matrix keypad with pull up resistors.
 - 2) 4 Switches configured as direct input switches with pull up resistors.
- **Serial Communication and wireless modules**
 - 1. USB/Serial TTL converter (CH340) for PC communication.
 - 2. UART Communication
 - 1) TTL (Full duplex) pin header.
 - 2) RS232 (MAX232) (Full duplex) standard DB9 female socket.
 - 3) RS485 (MAX487) (Half duplex) screw terminal.
 - 3. I²C
 - 1) DS1307 RTC (Real Time Clock) with backup battery.
 - 2) 24C08 Serial EEPROM (1KB).

- **Analog Inputs/outputs and sensors**

- 1) Two adjustable analog inputs using high precision multi-turn potentiometers.
- 2) Temperature sensor LM35 (0°C - 150°C) (0V - 1500mV).
- 3) Light Dependant Resistor LDR with adjustable sensitivity.
- 4) Analog output 0-5V (PWM-controlled).

- **Wireless**

1. Wi-Fi module (ESP-01) with reset switch and LED indicators for IO0 and IO2 ports.
2. IR (Infrared Receiver) sensor.

- **Outputs**

1. Two output relays (10A) both normally open and normally closed are available with status LED indicators.
2. DC motor control (speed and direction) up to 4A with CW/CCW direction status LED Indicator.
3. Output buzzer.

- **On-board connections**

- 6 pin header socket ICSP for connecting external programmers.
- 2X5 JTAG socket for Programming and debugging.
- All microcontroller I/O pins are brought out via pin header.

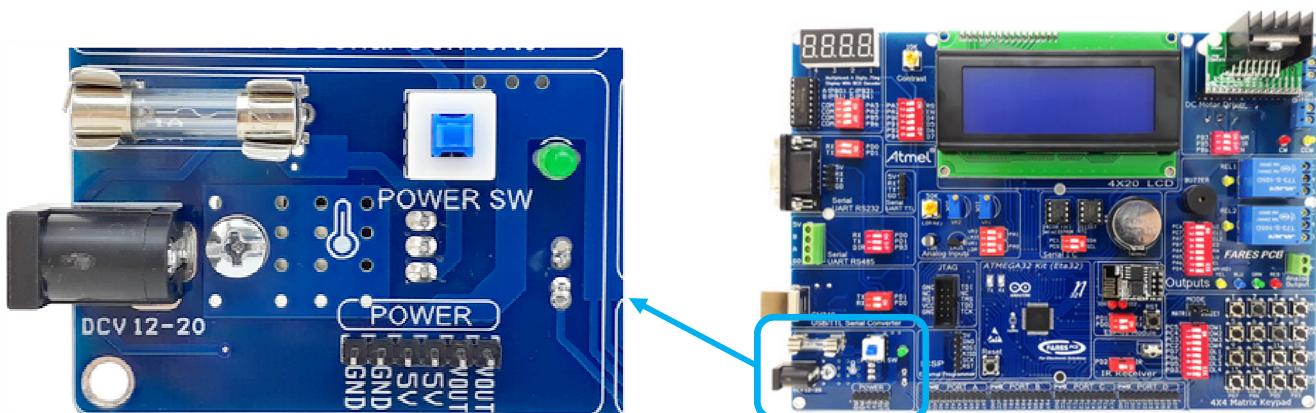
Power Supply Unit

Power could be supplied from DC power supply adaptor via DC power socket (12V - 20V) or from USB power via USB type B socket.

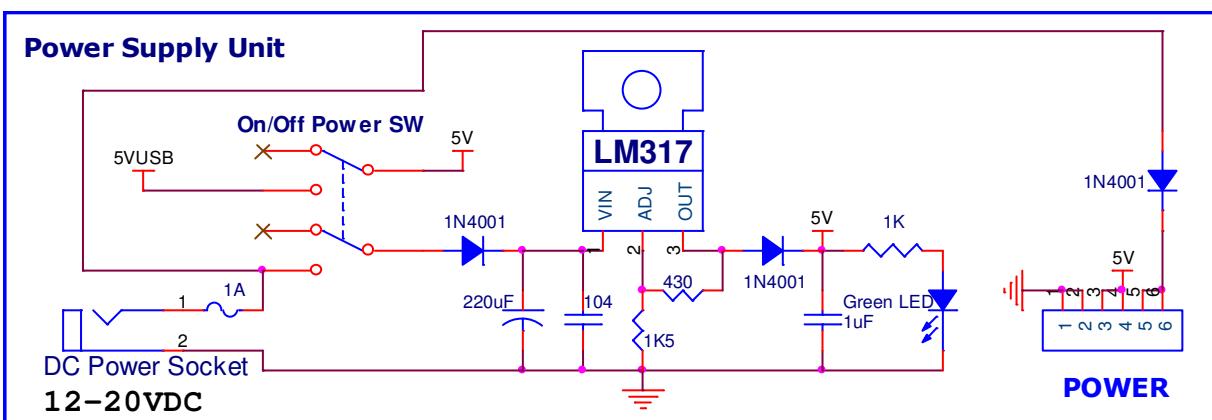
Power supply unit features:

- On/Off Power switch.
- Green LED power indicator.
- LM317 (Adjusted to 5V output regulator 5% tolerance).
- 1A Fuse for over current protection.
- GND , 5V and adaptor voltage(VOUT) are brought out for external using via header sockets.

Note: Eta32 kit is protected against reversed polarity of power.



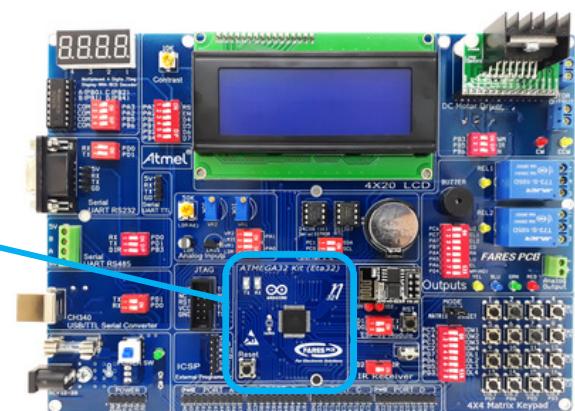
Power supply unit



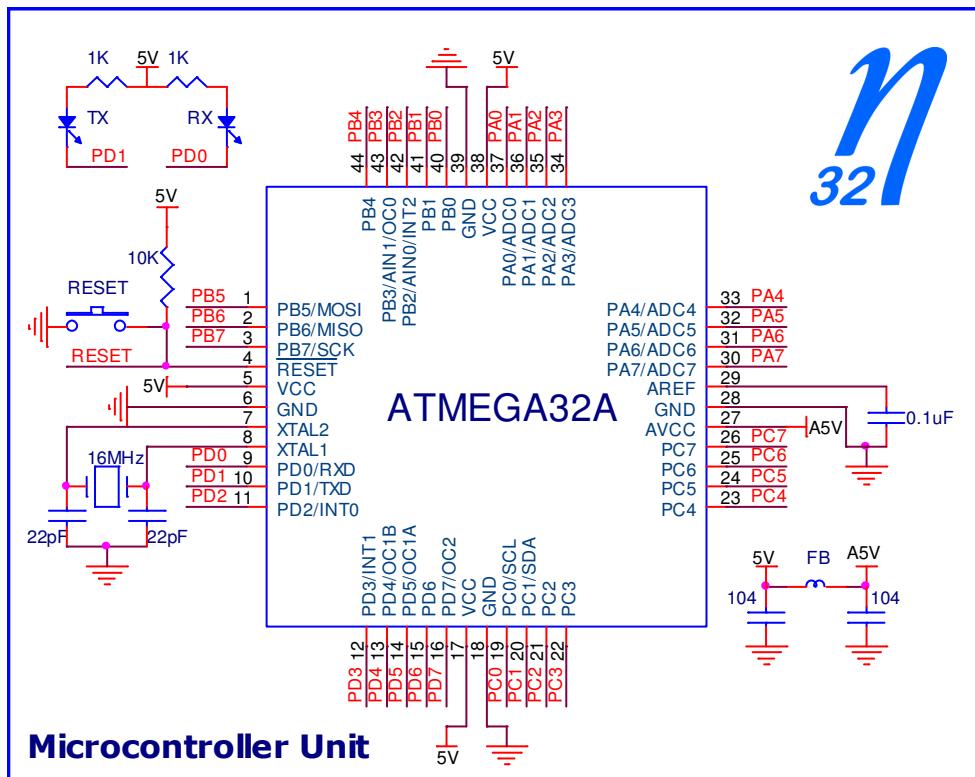
Power supply unit (schematic)

Microcontroller Unit

44 pin TQFP ATMEGA32A chip with 16 MHz crystal oscillator and Push button reset switch. Two red LEDs are included to indicate Tx and Rx status. All required bypass capacitors are included, in addition to ferrite bead to generate analog VCC (AVCC) from main power VCC for analog circuits biasing.



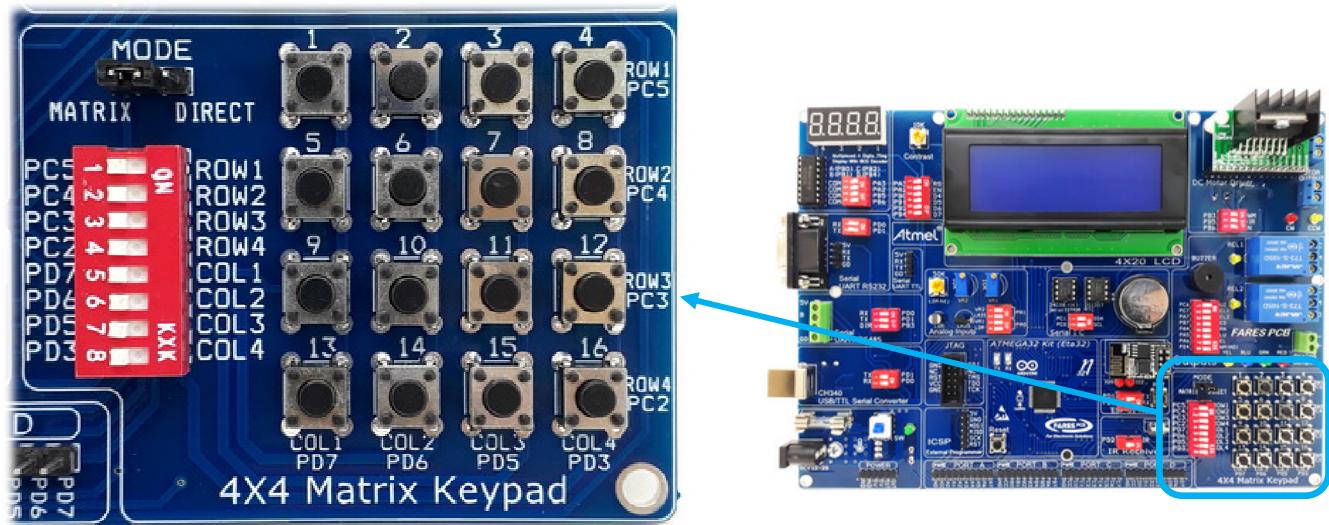
Microcontroller unit



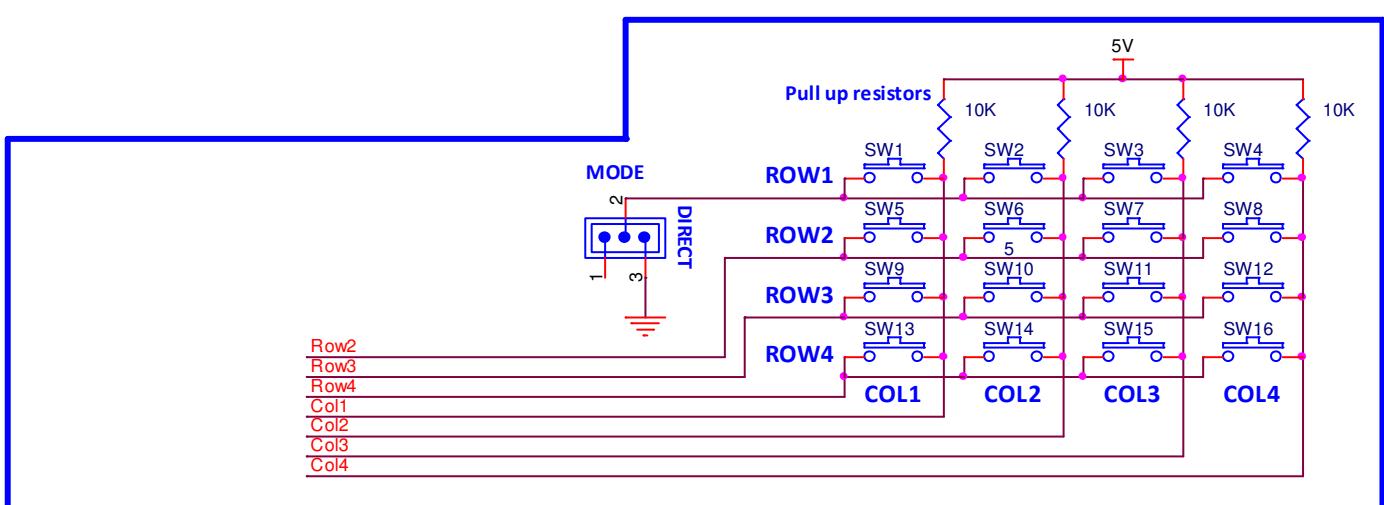
Microcontroller unit (schematic)

4X4 Matrix Keypad Unit

Eta32 Kit includes 16 push button switches. Switches are configured as 4 rows intersected by 4 columns. Each intersection creates a switch position.



4X4 matrix keypad unit

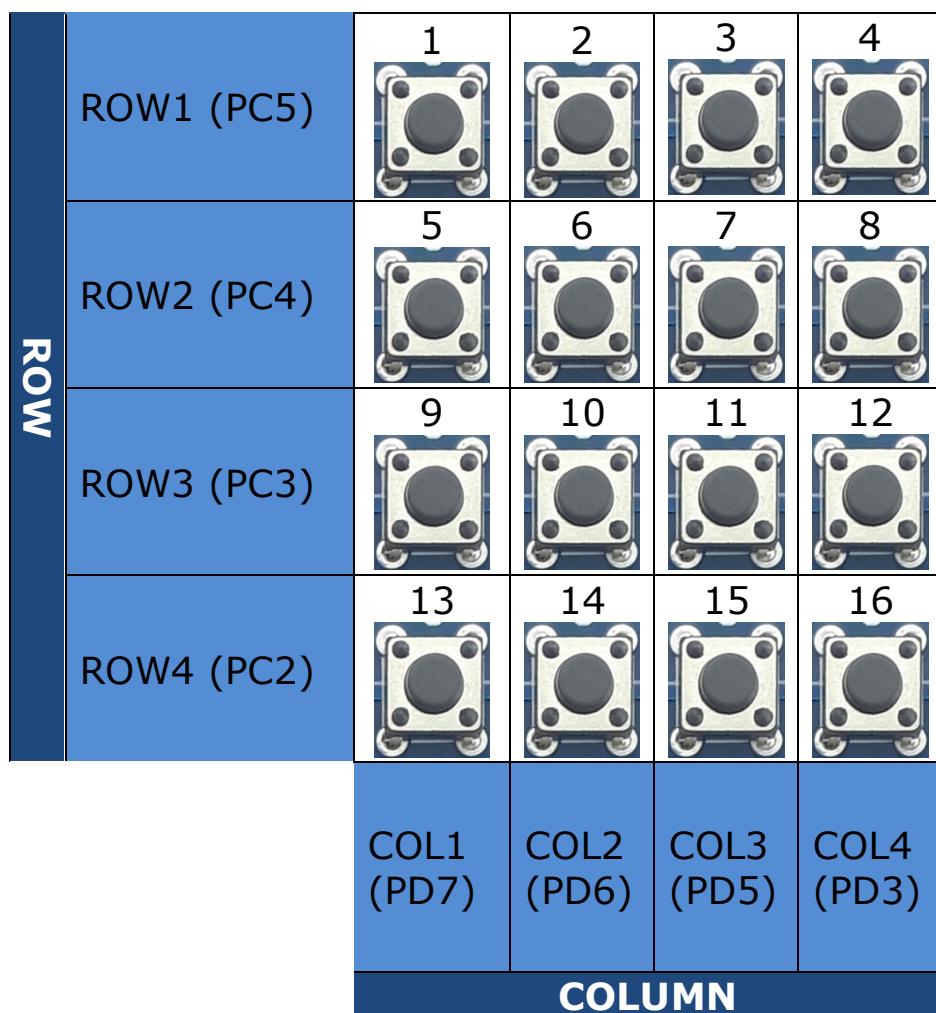


4X4 matrix keypad unit (schematic)

Rows and Columns are connected to microcontroller as following.

| Row number | Microcontroller pin | Arduino pin |
|------------|---------------------|-------------|
| ROW 1 | PC5 | 21 |
| ROW 2 | PC4 | 20 |
| ROW 3 | PC3 | 19 |
| ROW 4 | PC2 | 18 |

| Column number | Microcontroller pin | Arduino pin |
|---------------|---------------------|-------------|
| COL 1 | PD7 | 15 |
| COL 2 | PD6 | 14 |
| COL 3 | PD5 | 13 |
| COL 4 | PD3 | 11 |



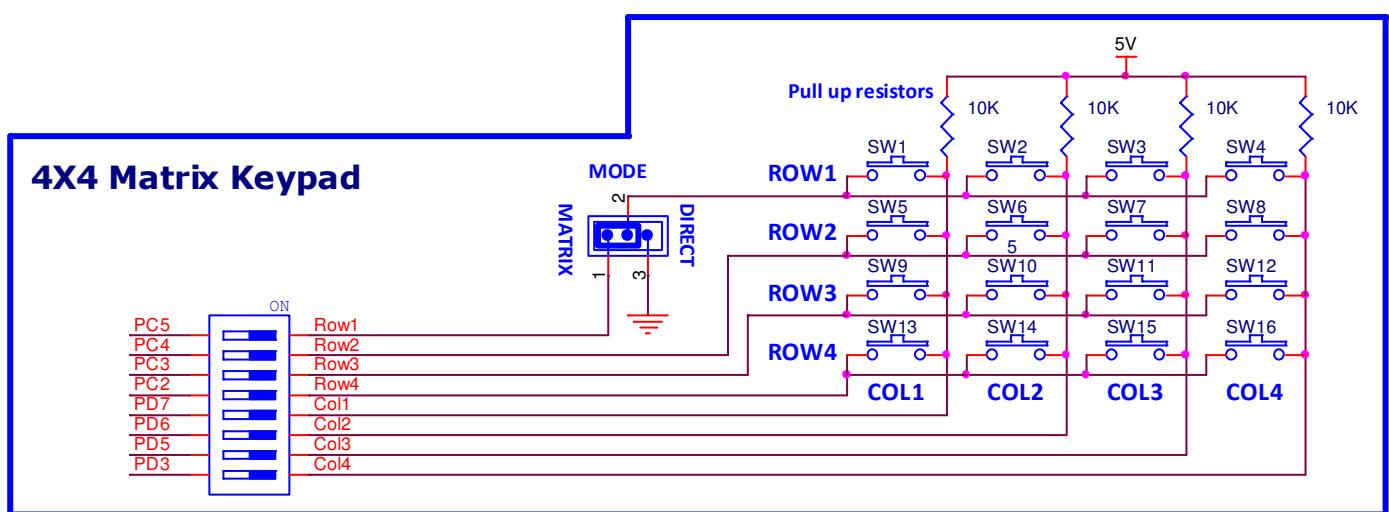
Keypad can be configured as 4X4 matrix keypad (16 switches used) or as 4 direct switches.

Matrix Mode

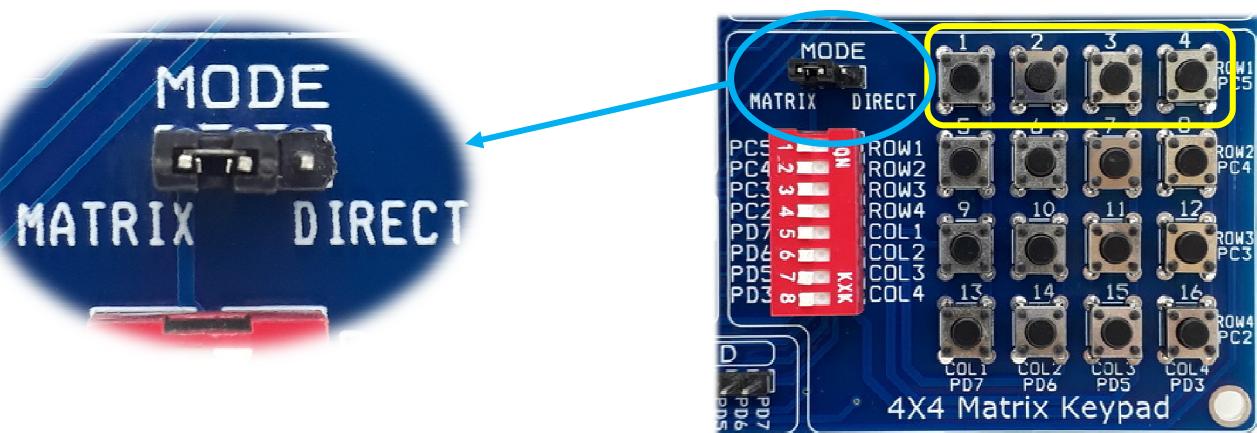
In matrix mode, all 16 switches are activated and configured as four row lines and four column lines. Microcontroller scans these lines to detect a button pressed state. Column lines are pulled up by $10\text{K}\Omega$ resistors. (i.e. microcontroller port pin reads high if no switch is pressed).

Scan process starts by setting all rows and columns as inputs.

To scan switches in a row, microcontroller configures it as output and sets it to low, then checks columns one at a time. If a column line goes low, microcontroller detects a pressed switch, otherwise, no pressed switch and goes to scan next row and so on.



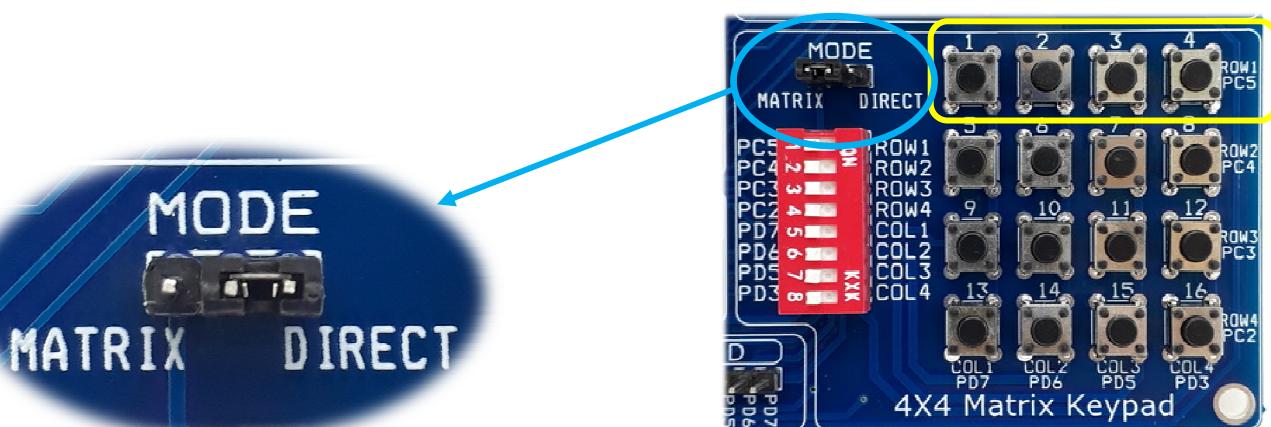
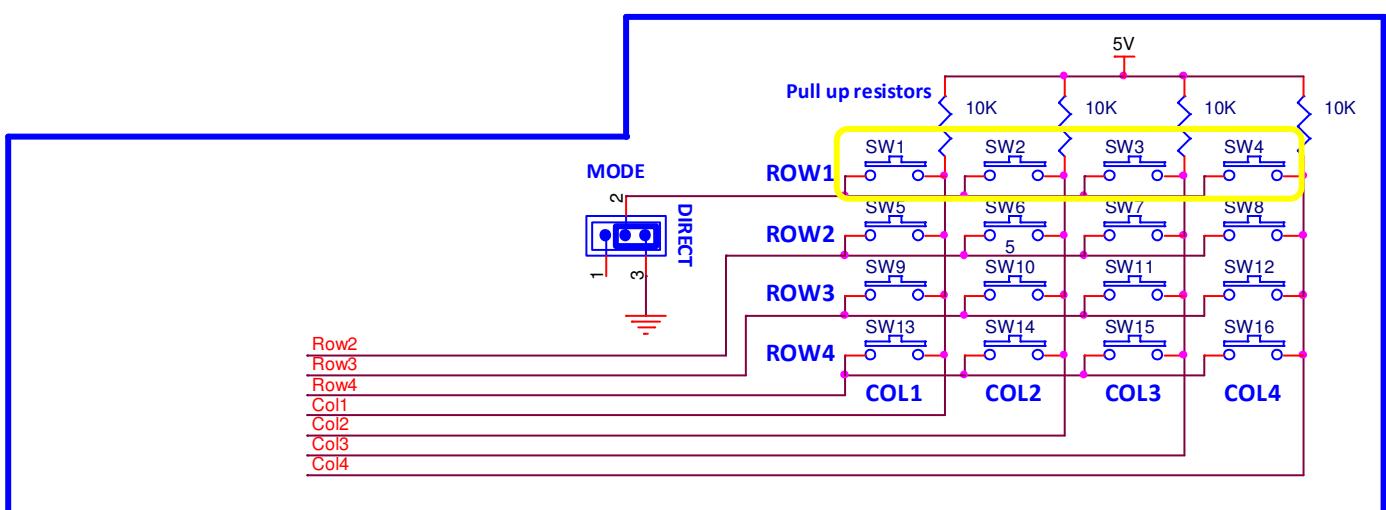
To set keypad in matrix mode, just set MODE jumper to Matrix position as shown below.



Direct Mode

In direct mode, just 4 switches are activated to save microcontroller pins in case of no need for more switches and reducing firmware complexity in scanning operation. Switches in first row only are enabled (i.e. SW1, SW2, SW3 and SW4) and can be read directly as inputs through column lines. Microcontroller checks the column lines. If a column line goes low, it means a switch is pressed.

To set keypad in direct mode, just set MODE jumper to direct position, as shown below:



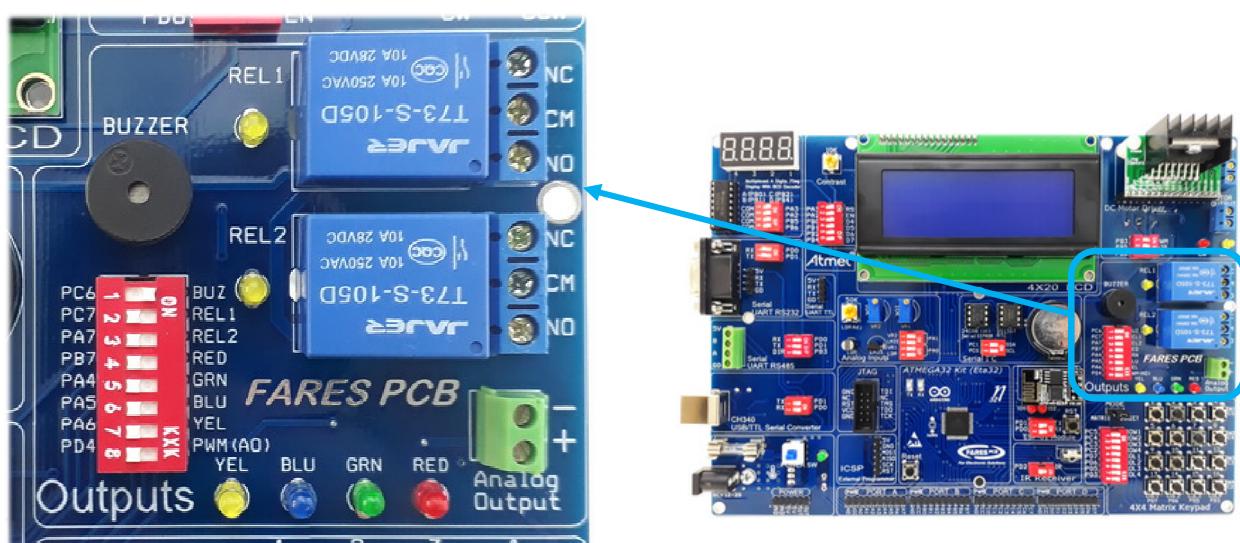
Outputs Unit

This unit contains eight outputs divided as following

- Four output LEDs.
- Two output relays.
- One output Buzzer.
- One analog output.

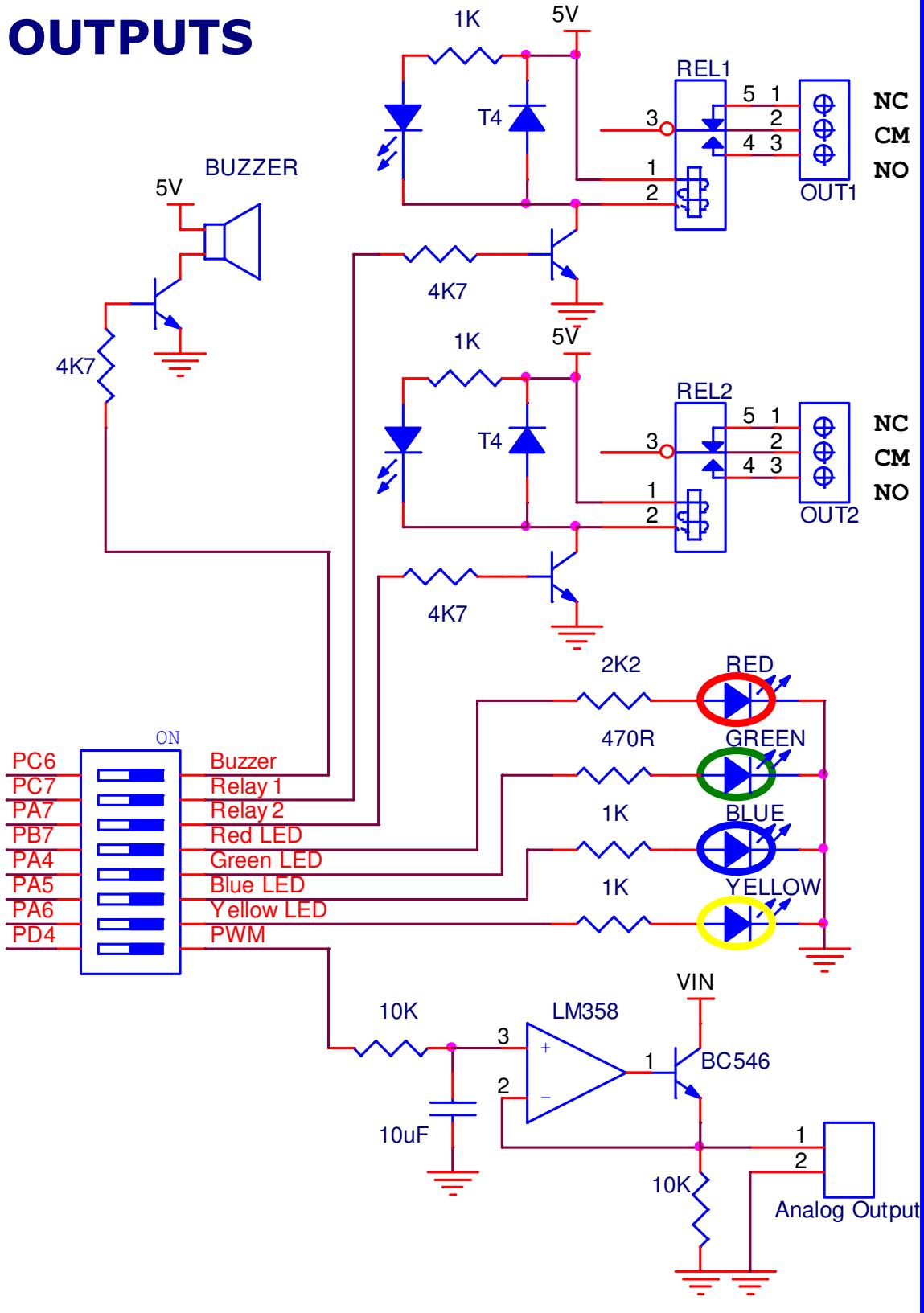
Outputs are connected to microcontroller as following:

| Output | Microcontroller pin | Arduino pin |
|---------------------|---------------------|-------------|
| Buzzer | PC6 | 22 |
| Relay1 | PC7 | 23 |
| Relay2 | PA7 | A7 |
| Red LED | PB7 | 7 |
| Green LED | PA4 | A4 |
| Blue LED | PA5 | A5 |
| Yellow LED | PA6 | A6 |
| Analog output (PWM) | PD4 | 12 |



Output unit

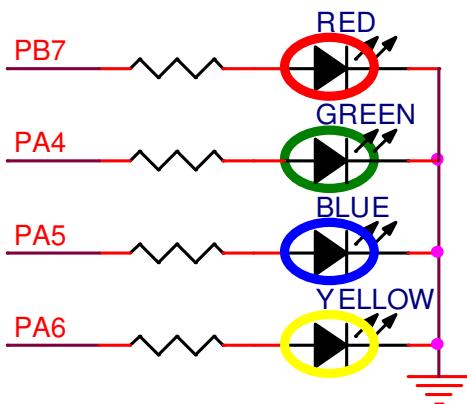
OUTPUTS



Output unit (Schematic)

Output LEDs

Four LEDs with current limiting resistors.



Red LED is connected to **PB.7**

Green LED is connected to **PA.4**

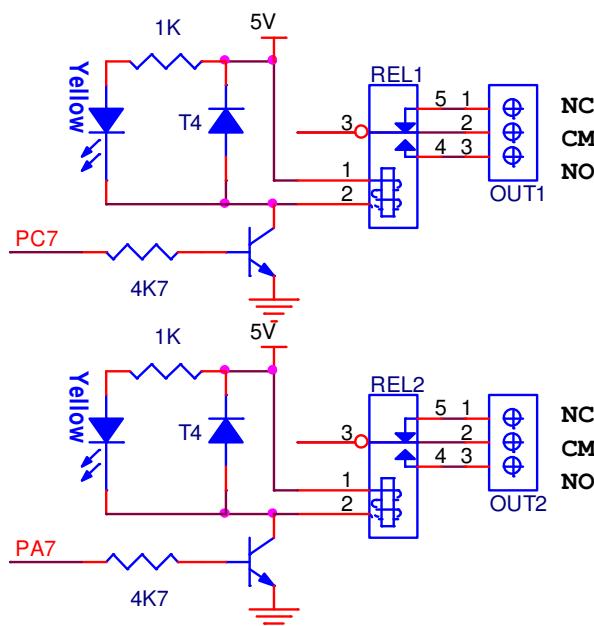
Blue LED is connected to **PA.5**

Yellow LED is connected to **PA.6**

Each LED can be enabled or disabled via DIP switch. LEDs are active high. i.e. output high turns LED on and output low turn it off.

Output relays

Two output relays are added to **Eta32** kit to allow dry contact switches which is suitable for AC or DC switching applications. Each relay has its own related LED for status indication and can be individually enabled via DIP switch. Relays are driven from NPN transistors. Freewheeling diodes are included to protect transistors from back EMF voltage that arises on relay coil during switching off.

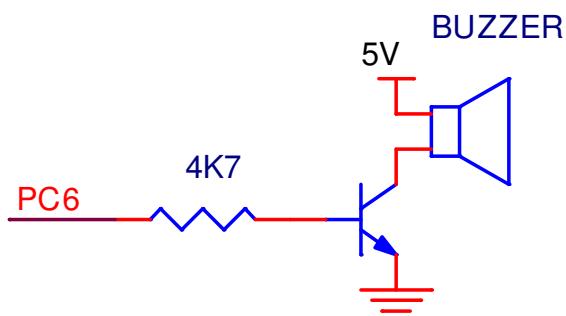


Relay1 is attached to port pin **PC7** and relay2 is attached to port pin **PA7**. Relays are 5V coil and rated to 10A contacts (resistive load)(practically limited for about 5A).

Both of normally open and normally closed contacts are brought out via screw clamp terminals.

Output buzzer

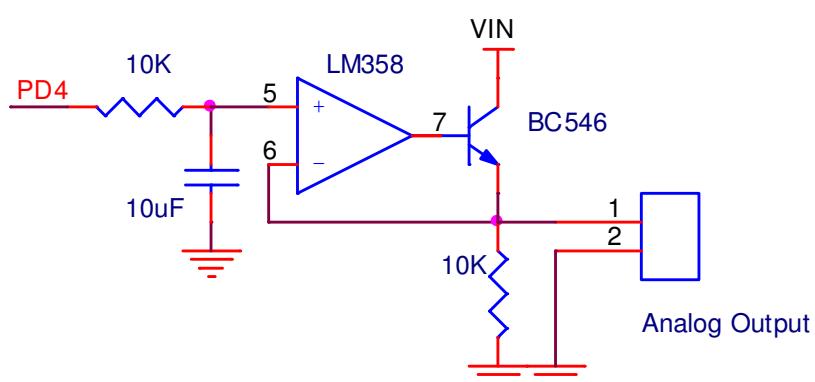
One output buzzer (5VDC) is included in output unit to port pin **PC6**. Also, it can be enabled using DIP switch.



Analog Output

ATMEGA32 microcontroller doesn't provide digital to analog conversion. Instead, it provides pulse width modulated outputs which can be used to generate analog voltage with help of a simple external circuit. The circuit is a simple RC low pass filter with buffer. This circuit converts the PWM signal to an analog voltage **inversely** proportional to the duty cycle. The circuit receives a PWM signal from microcontroller port pin PD4 (OC1B). this port pin can be configured to generate PWM with few instructions written to configure timer1 to operate in PWM Mode.

Circuit is designed to generate an analog voltage range from 0 to 5V and output current up to 100mA.



7Segment Display

7segment display is used to indicate numerical data. It can display digits from 0 to 9. 7segment display is very popular and has many applications.

Eta32 includes multiplexed common anode four digits 7segment display in addition to 7447 BCD decoder to simplify firmware.

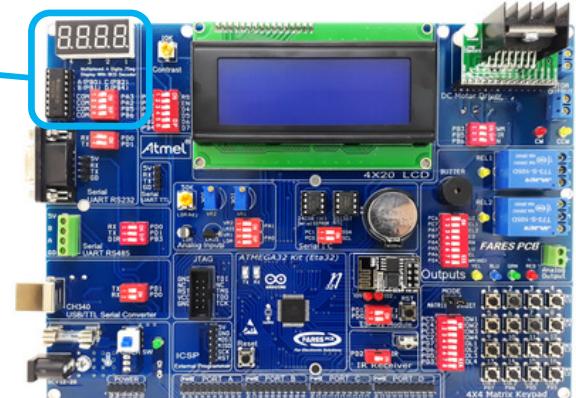
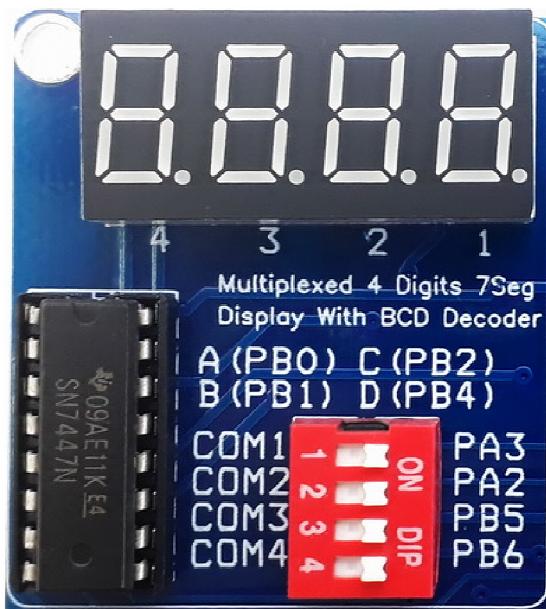
Multiplexed Four Digits 7seg Display Unit

Four multiplexed 7segment digits are included in **Eta32** kit. 7segments are referred to by letters "A", "B", "C", "D", "E", "F", "G". All digits share the same segments. i.e. segment "a" is the same for all digits. 7segment code is generated from 7447 BCD decoder IC, 7447 converts binary data received from microcontroller into 7 segment code.

Table below shows binary data vs the generated 7 segment code.

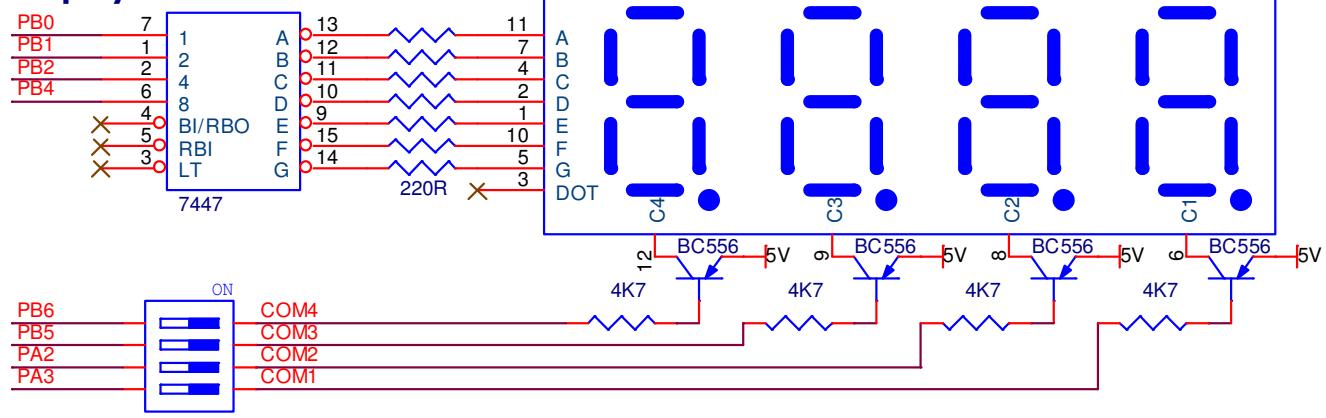
| Digit | Decoder input | | | | Decoder output | | | | | | |
|-------|----------------|----------------|----------------|----------------|----------------|---|---|---|---|---|---|
| | D(Bit3) PB4 | C(Bit2) PB2 | B(Bit1) PB1 | A(Bit0) PB0 | A | B | C | D | E | F | G |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 2 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 4 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 6 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

Each digit has its own common (Anode type), which is derived by discrete PNP transistors. Transistor switch is connected to microcontroller via DIP switch. To enable a digit, initiate the specific pin to low level to bias transistor which in turn pull the anode common to 5V, and up on the decoder outputs, data will be displayed.



7Segment display

Multiplexed 4 Digits 7seg Display with BCD Decoder



7Segment display (schematic)

BCD decoder inputs are connected to microcontroller ports as shown in table:

| BCD Decoder Input | microcontroller Pin | Arduino Pin |
|-------------------|---------------------|-------------|
| A (Bit0) | PB0 | 0 |
| B (Bit1) | PB1 | 1 |
| C (Bit2) | PB2 | 2 |
| D (Bit3) | PB4 | 4 |

7Seg Commons are connected to microcontroller ports as shown in table:

| Common | microcontroller Pin | Arduino Pin |
|--------|---------------------|-------------|
| COM 1 | PA3 | A3 |
| COM 2 | PA2 | A2 |
| COM 3 | PB5 | 5 |
| COM 4 | PB6 | 6 |

Note:-

- Each 7seg digit can be enabled or disabled individually using DIP switch.
- 7seg module and 4X20 LCD share some of port pins. So, disable LCD (turn off DIP LCD switch) before using 7Segments display.

4X20 LCD Display Unit

LCD (Liquid crystal display) is a more informative output device than 7segment digits. The power of LCD lies in its capability of showing characters.

Standard LCD module pin out number, symbol and function is shown in table.

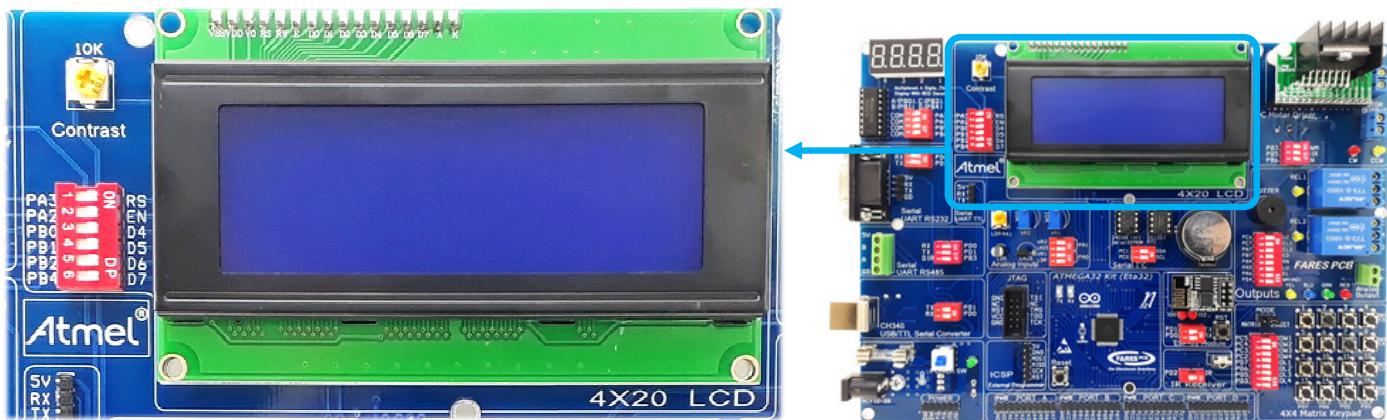
| LCD pin number | LCD pin symbol | LCD pin function |
|----------------|----------------|-------------------------------------|
| 1 | VSS | Ground |
| 2 | VCC | +5V |
| 3 | VO | Contrast adjustment |
| 4 | RS | Register Select(0:Command , 1:Data) |
| 5 | R/W | R/W(0:Write , 1:Read) |
| 6 | EN | Enable |
| 7 | D0 | Data bit 0 |
| 8 | D1 | Data bit 1 |
| 9 | D2 | Data bit 2 |
| 10 | D3 | Data bit3 |
| 11 | D4 | Data bit4 |
| 12 | D5 | Data bit5 |
| 13 | D6 | Data bit6 |
| 14 | D7 | Data bit7 |
| 15 | A | Back light anode(+) |
| 16 | K | Back light cathode(-) |

Eta32 kit contains 4X20 character LCD. Four lines of characters each has 20 characters line length is quite sufficient to show quite amount of information. LCD is configured in 4 bit mode and connected to microcontroller via DIP switch as shown in table:

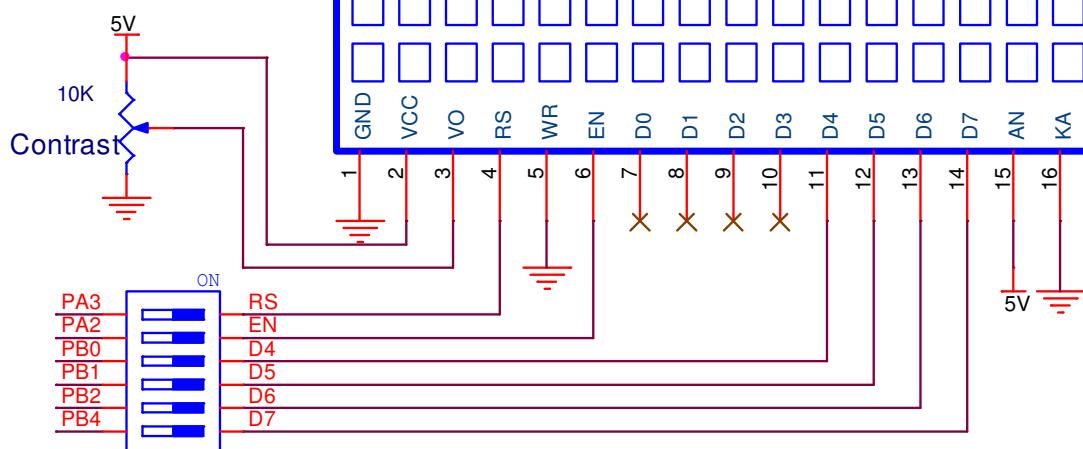
| LCD pin number | LCD pin symbol | Microcontroller pin | Arduino Pin |
|----------------|----------------|---------------------|-------------|
| 4 | RS | PA3 | A3 |
| 6 | EN | PA2 | A2 |
| 11 | D4 | PB0 | 0 |
| 12 | D5 | PB1 | 1 |
| 13 | D6 | PB2 | 2 |
| 14 | D7 | PB4 | 4 |

Note:

- LCD R/W signal is tied to ground.
- 10KΩ variable resistor labeled "Contrast" is adjusted to control the LCD contrast.
- Disable 7Segment module (Turn off 7seg DIP switch) before using LCD module.

**2X16 LCD display unit**

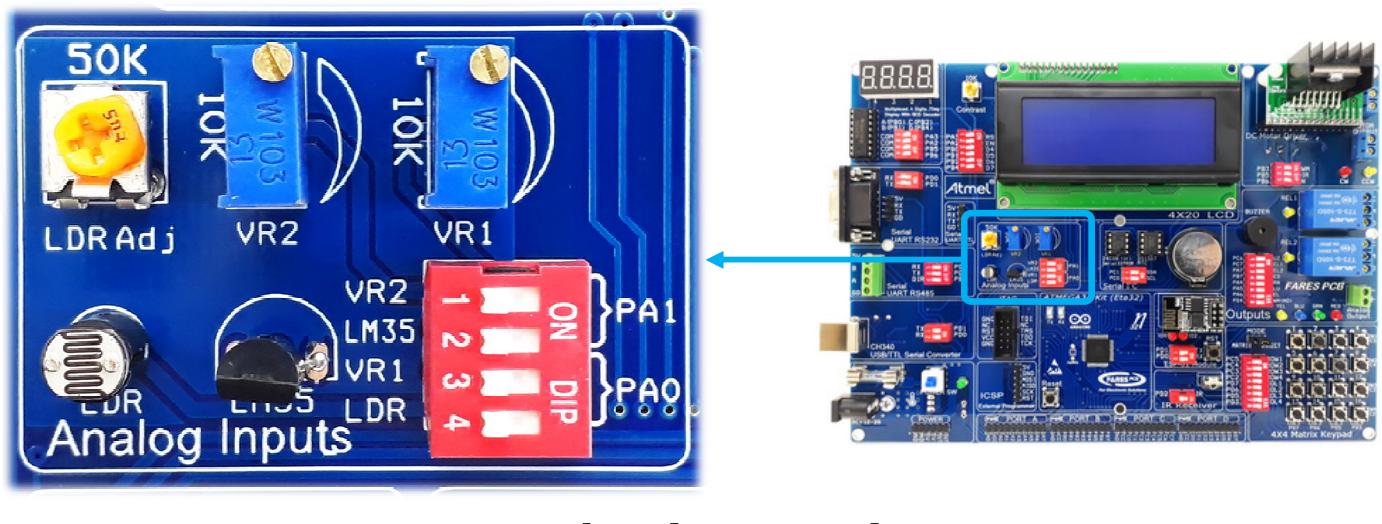
4X20 Character LCD

**2X16 LCD display unit (Schematic)**

Analog Input Unit

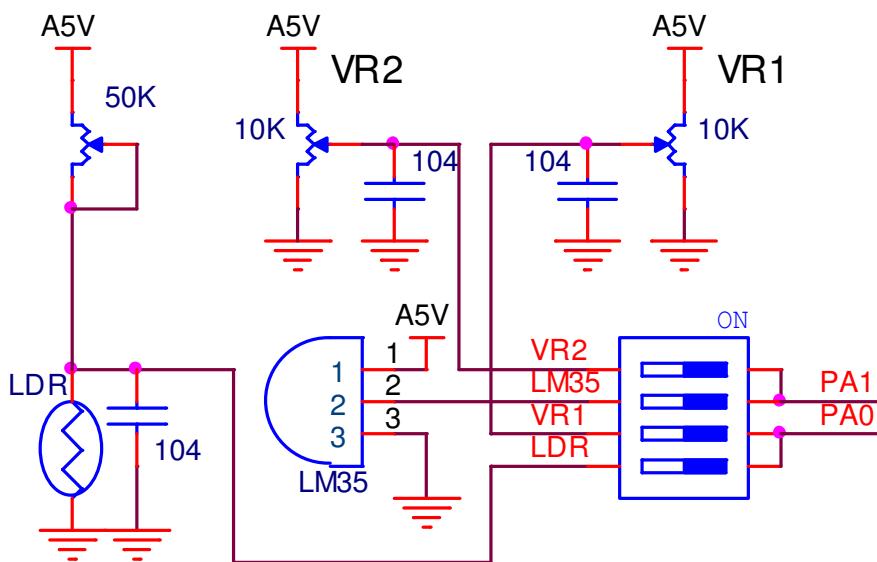
Eta32 kit includes four sources of analog input signal that can be measured by microcontroller.

1. Variable Resistor (VR1).
2. Variable Resistor (VR2).
3. Light Dependent Resistor (LDR).
4. Temperature sensor (LM35).



Analog input unit

Analog Inputs



Analog input unit (schematic)

Variable Resistor (VR1)

Multi-turn potentiometer provides stable and precise resistance so it is used mainly for trimming and precision adjustments.

VR1 is a 10KΩ multi-turn variable resistor. The fixed terminals of variable resistor are connected to GND and AVCC(5V) whereas the variable terminal is connected to microcontroller PA0 via DIP switch (channel 3). The resistor can be adjusted precisely to the required voltage (0.00V to 5.00V).

Variable Resistor (VR2)

It's the same as VR1 but it connected to PA1 via DIP switch (channel 1).

Light Dependent Resistor LDR

LDR is a passive electronic component that has a resistance which varies depending on the light intensity. LDR resistance is ranged from Several hundreds of Kilo ohms in darkness to few hundreds of ohms when a light falls on it.

LDR and a 50K ohm potentiometer form a voltage divider circuit which provides a variable voltage that is inversely proportional to the light intensity. Adjust the variable resistor to obtain the required sensitivity. The analog voltage is delivered to microcontroller PA0 Via DIP switch(channel 4).

Temperature sensor (LM35)

LM35 is a precession integrated circuit temperature sensor whose output voltage varies based on the temperature around it. It can be used to measure temperature anywhere between -55°C to 150°C. It doesn't require any calibration. LM35 provides analog voltage that is linearly proportional to the Celsius temperature. Each increase in one degree Celsius cause increase of 10mV in output voltage.

Follow the next equation to calculate the temperature:

$$\text{Temp}({}^{\circ}\text{C}) = \text{Output Voltage (mV)} / 10$$

Output voltage connected to PA1 via DIP switch (channel 2).

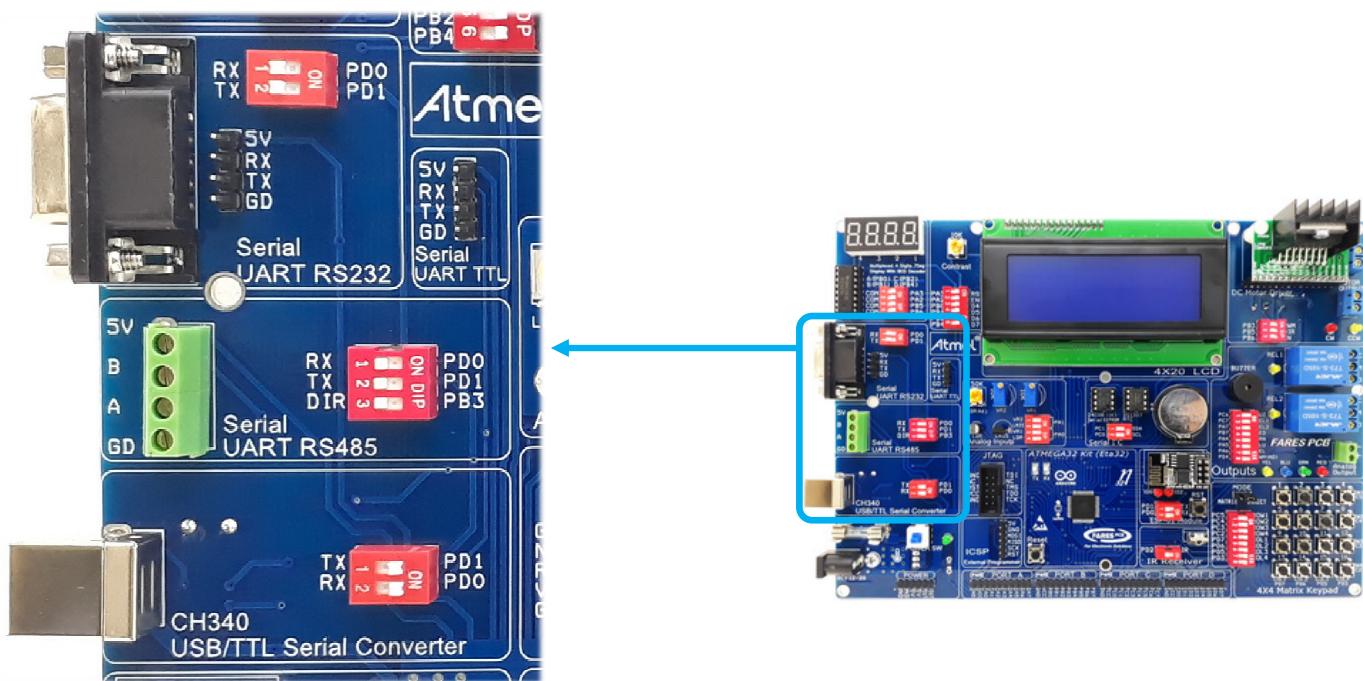
Analog input connected to microcontroller as shown in table:

| Analog Input | microcontroller Pin | Arduino Pin |
|---------------------------|---------------------|-------------|
| Variable resistor (VR1) | PA0 | A0 |
| LDR | | |
| Variable resistor (VR2) | PA1 | |
| Temperature sensor (LM35) | | A1 |

UART serial communication

UART is one of the most famous ways to communicate between microcontrollers or microcontroller and PC. UART is a byte oriented protocol that determines a specific way to transmit a byte serially between two devices.

This type of protocol sends data as a string of characters. UART only requires two signal lines to successfully communicate, a TXD (transmit data) and RXD (receive data) line as well as a common ground line (used as a reference point). When communicating with another UART device, the TXD line will be attached to a corresponding RXD line, and vice-versa. No clock line is used with the UART protocol. Rather, users instead specify a particular baud rate for the two devices to operate at. A baud rate indicates how many bits, including data, start, stop, and parity bits, are transferred over the data lines in one second time frame. A common UART communication configuration uses a start bit, 8 data bits, no parity, and a single stop bit.



TX and RX signals can be physically represented in various values. TTL , RS232 or RS485.

In UART TTL signals are represented as following

$$V_{OL} = 0V \text{ to } 0.4V$$

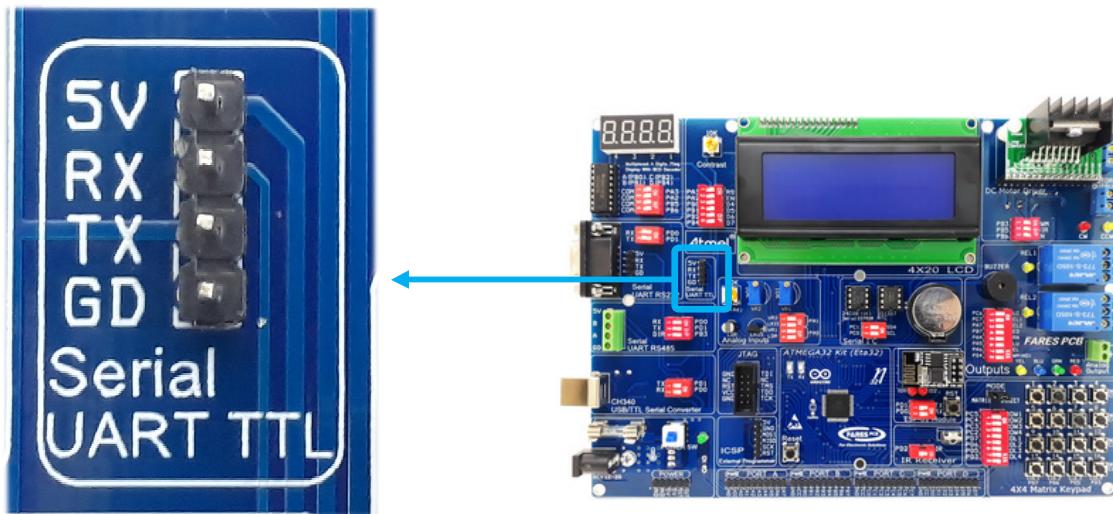
$$V_{OH} = 2.4V \text{ to } 5V$$

$$V_{IL} = 0V \text{ to } 0.8V$$

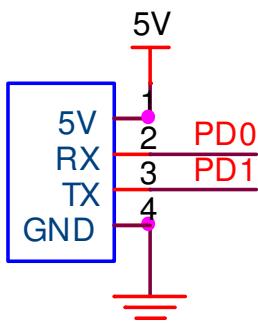
$$V_{IH} = 2V \text{ to } 5V$$

This voltage level is adequate for transmitting data between two adjacent devices (normally < 50cm) and has a limited baud rate vs distance.

Eta32 kit provides UART TTL via 4 pin header socket GND,TX,RX and 5V TX and Rx are connected directly to microcontroller pins PD1,PD0 respectively.



Serial UART TTL



5V can be used to supply power for external target board.

In UART RS232 signals are represented as following

$$V_{OL} = +5V \text{ to } +25V$$

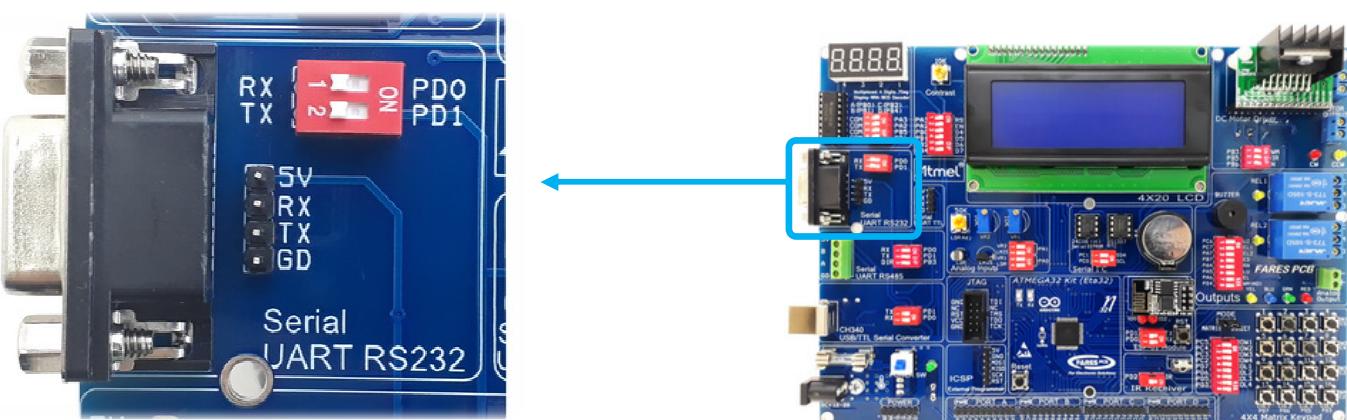
$$V_{OH} = -5V \text{ to } -25V$$

$$V_{IL} = +3V \text{ to } +25V$$

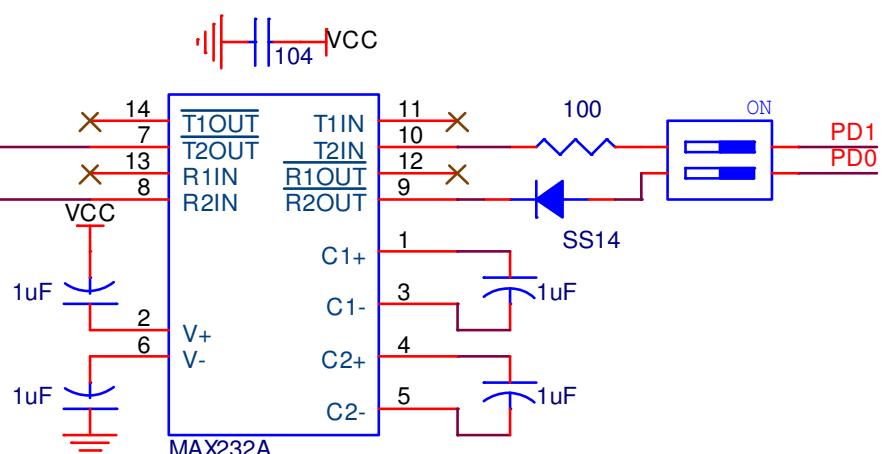
$$V_{IH} = -3V \text{ to } -25V$$

This voltage level is more suitable for transmitting data between more far devices (several meters) and improves baud rate vs distance.

Eta32 kit provides UART RS232 via standard DB9 socket for direct connection to PC's serial port or any other board that supports DB9 socket. Also, RS232 signals are provided via 4 pin header socket GND, TX, RX and 5V for external using flexibility. Logic level conversion from TTL (Microcontroller side) to RS232 (DB9 Side) is realized using MAX232 chip. TX and RX TTL signals are connected to microcontroller PD1 and PD0 respectively via DIP switch.



Serial UART RS232



In UART RS485, one signal is transmitted across a pair of lines. Line "A" and Line "B" (differential signal) . The two signaling states of the line are represented as following

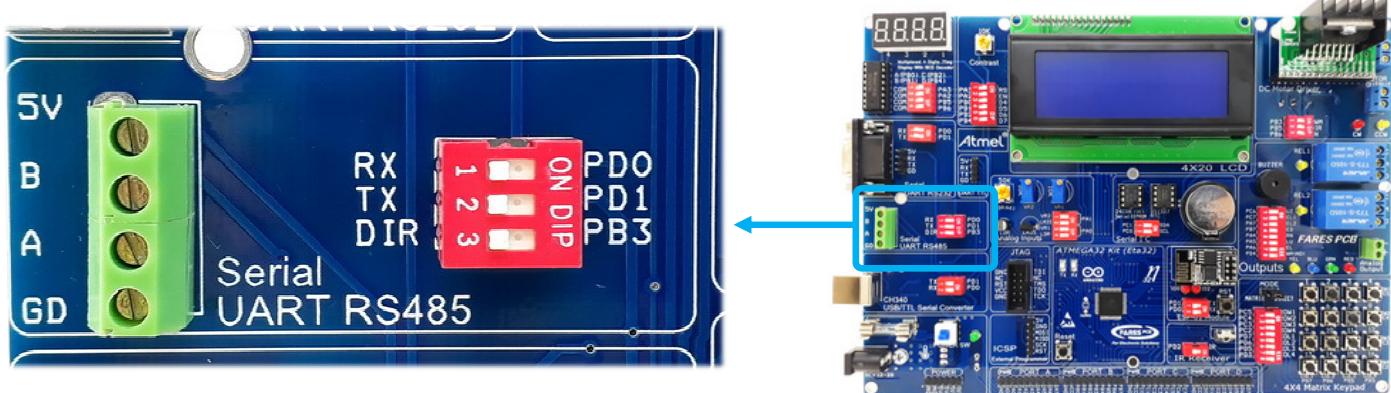
When A terminal is positive with respect to B terminal then the line is High ($6V > VA-VB \geq 200mV$)

When A terminal is negative with respect to B terminal then the line is Low ($-6V > VA-VB \geq -200mV$)

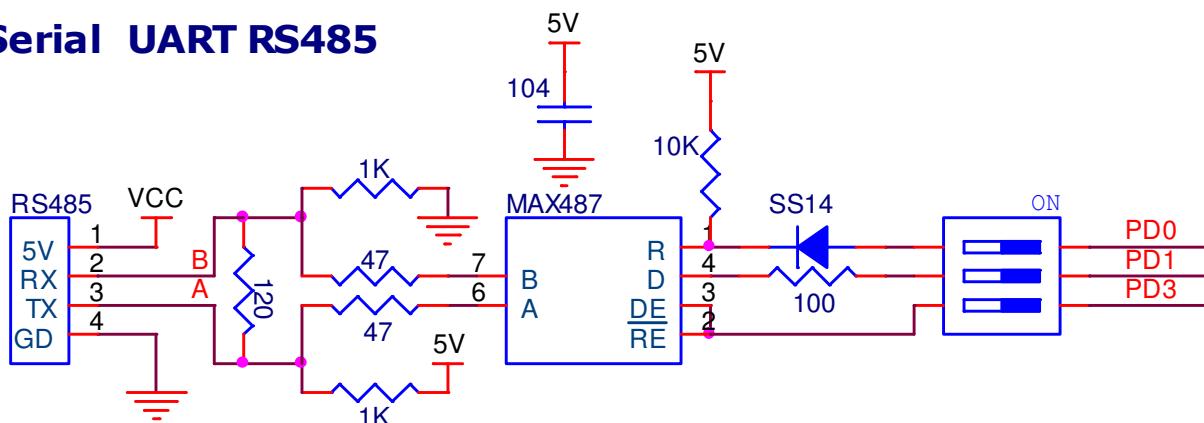
Differential signals are immune for noise far than single-ended signals. The differential signal representation is suitable for transmitting data for long distances 1KM with high baud rates.

Eta32 kit provides UART RS485. Logic level conversion from TTL to RS485 is realized using MAX487 Chip. MAX487 supports up to 128 transceiver on the bus.

An additional control pin (DIR) required for direction setting to switch between TX and RX mode and vice versa.



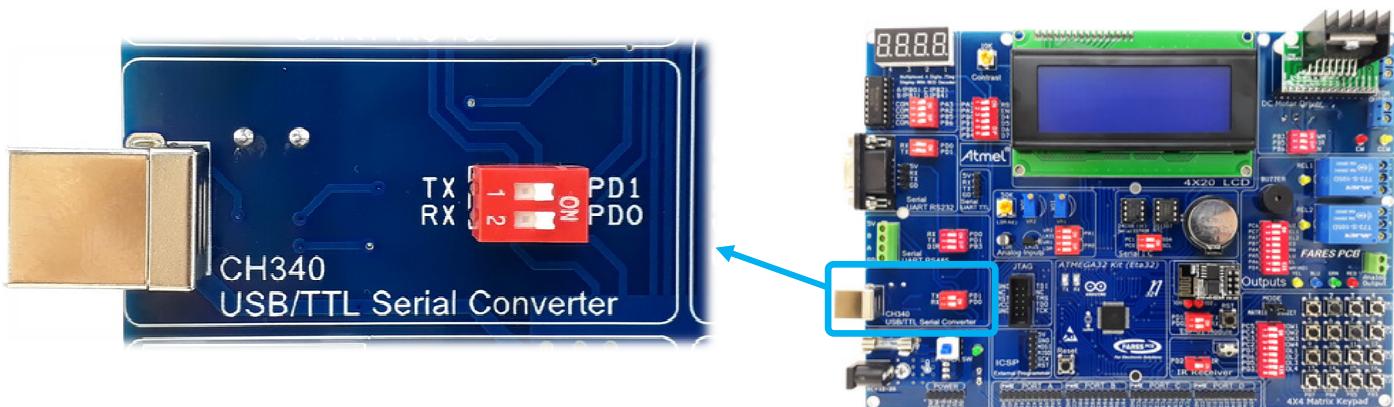
Serial UART RS485



ETA32 kit contains on board LED indicators for TX and RX signal statuses.

USB Virtual COM Unit

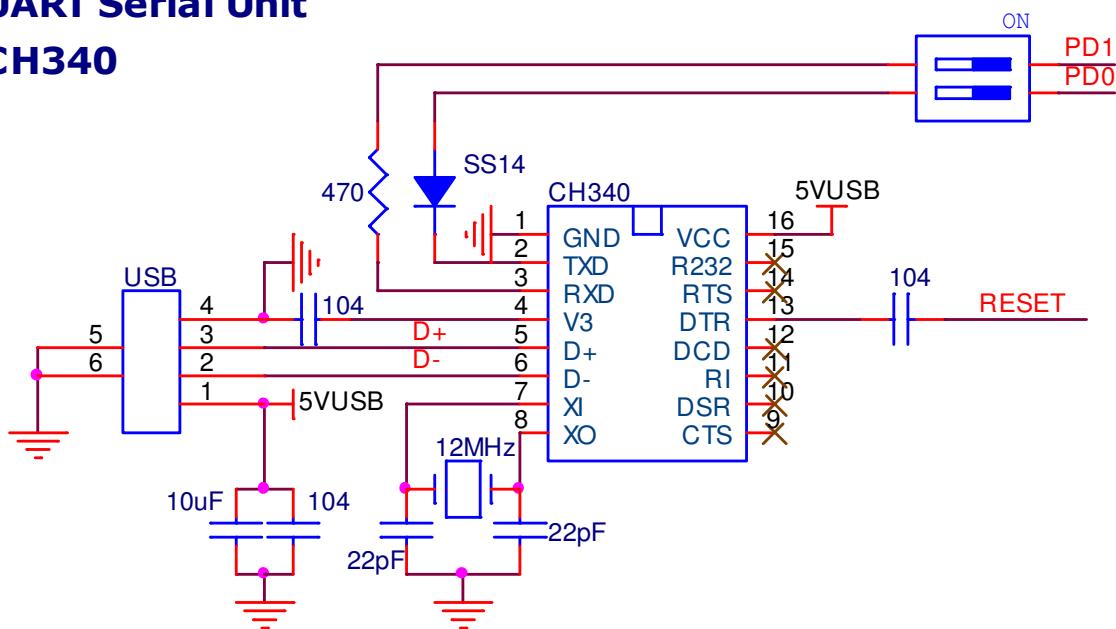
This unit includes USB/TTL logic converter using CH340 chip and provides USB type B socket for PC serial communication. Serial interface circuit can be enabled or disabled using DIP switch. USB/TTL Converter chip is connected to ATMEGA32A microcontroller port pins **PD0 (RX)** and **PD1 (TX)** via DIP switch.



USB/TTL Serial Converter

UART Serial Unit

CH340



USB/TTL Serial Converter (schematic)

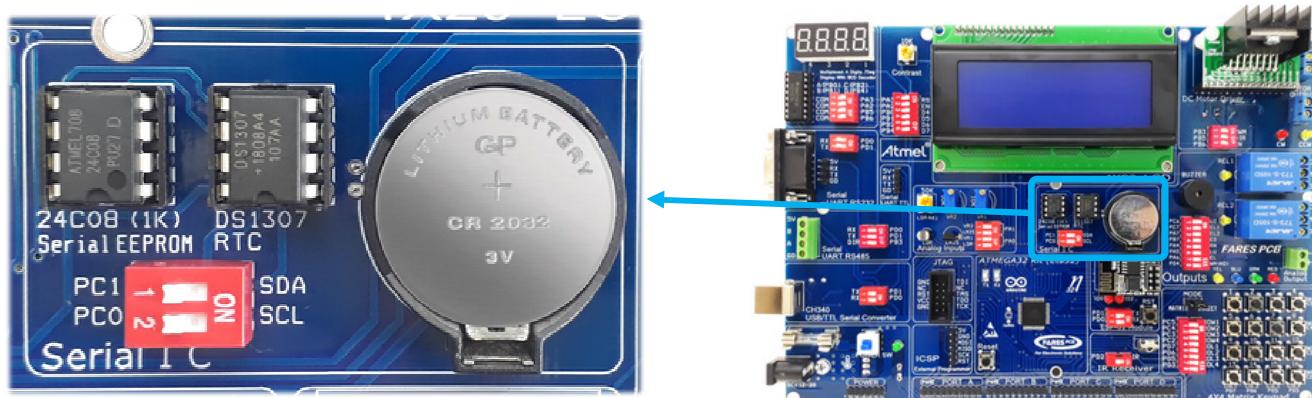
I²C serial communication

I²C is a serial protocol for two wire interface to connect low speed devices like microcontrollers, EEPROMs, A/D , D/A and other similar peripherals in embedded systems.

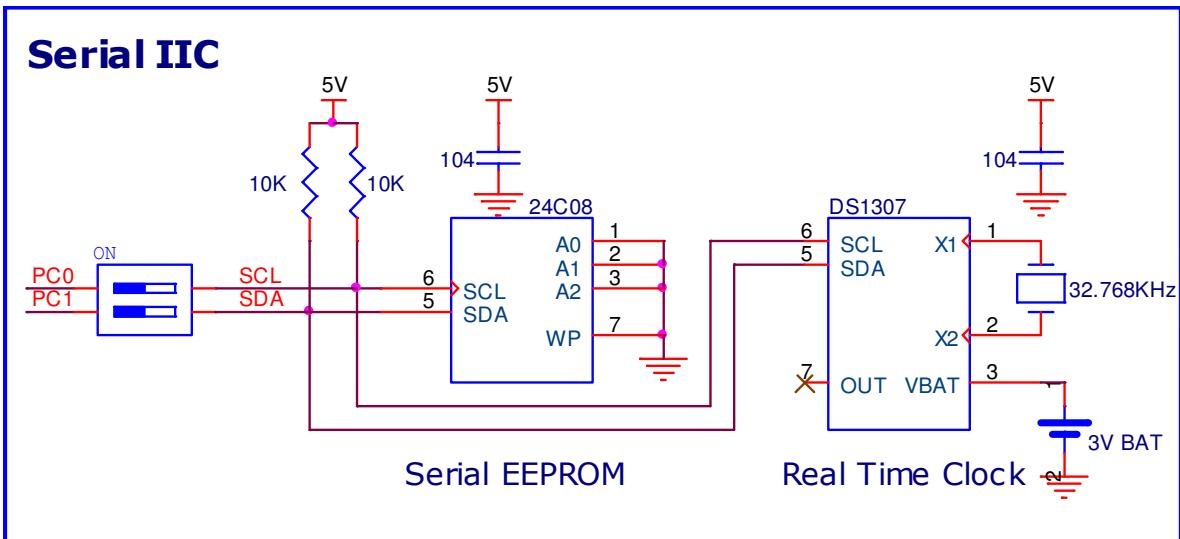
I²C uses two wires, SCL(Serial Clock) and SDA(Serial Data). Both lines need a pulled up resistor to 5V. Each data bit is transmitted on SDA line is synchronized with a clock pulse on the SCL line.

I²C bus allows multiple slaves to communicate with one or more masters. Each slave device has a unique address that is represented of 7 bits. In normal state, both lines (SCL and SDA) are high. The communication is initiated by the master upon generating a Start condition followed by the device address. If bit 0 of the address byte was set to 0 the master will write to slave otherwise the next byte will be read from the slave. Once all bytes are written or read the master generates a Stop condition. ATMEGA32A microcontroller provides two wire serial interface peripheral which supports speed up to 400kbps and can be configured as master or slave with 7 bit address space that allows up to 128 different slave addresses.

Eta32 kit includes two I²C serially interfaced devices. 1KB Serial EEPROM (AT24C08) and real time clock IC RTC (DS1307). Both chips share the I²C bus lines SCL(PC0) and SDA(PC1) via DIP switch.



Serial I²C

**Serial I²C (Schematic)**

Serial EEPROM (AT24C08)

The AT24C08 memory provides 8192 bits of serial electrically erasable and programmable read only memory (EEPROM) organized as 1024 bytes of 8 bit.

The device address byte of the chip (AT24C08) is &HA0

| | | | | | | | |
|---|---|---|---|---|---|---|-----|
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | R/W |
|---|---|---|---|---|---|---|-----|

For more details about AT24C08 chip, refer to its datasheets.

RTC (DS1307)

DS1307 is a low-power clock/calendar with 56 bytes of battery-backed SRAM. The clock/calendar provides seconds, minutes, hours, day, date, month, and year information. DS1307 operates as a slave device on the I²C bus. Access is obtained by implementing a START condition and providing a device identification code followed by a register address. Subsequent registers can be accessed sequentially until a STOP condition is executed. A backup battery is included. When VCC falls below VBAT, the device switches into a low current battery-backup mode. Upon power-up, the device switches from battery to VCC when VCC is greater than VBAT +0.2V.

The device address byte of the chip (DS1307) is &HD0

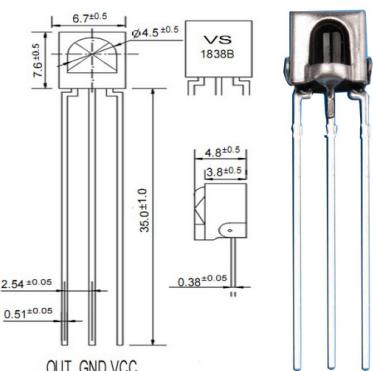
| | | | | | | | |
|---|---|---|---|---|---|---|-----|
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | R/W |
|---|---|---|---|---|---|---|-----|

For more details about DS1307 chip, refer to its datasheets.

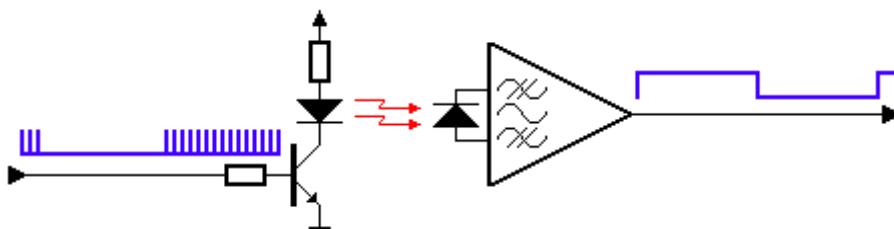
IR receiver (VS1838)

VS1838 is a miniaturized receiver for infrared remote control systems. It consists of PIN diode and preamplifier assembled on a lead frame while the epoxy package acts as an IR filter.

VS1838 can be used to receive all common IR remote control data formats. When you hit a key on your remote, the IR LED will transmit encoded data to IR receiver. Transmitted data is a stream of pulses of 38KHz frequency. VS1838 receives this signal and demodulates it to extract a binary waveform that can be read by microcontroller.

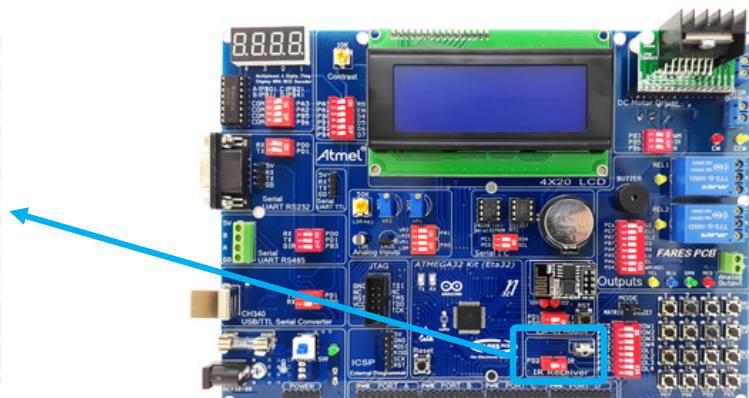


IR receiver terminals

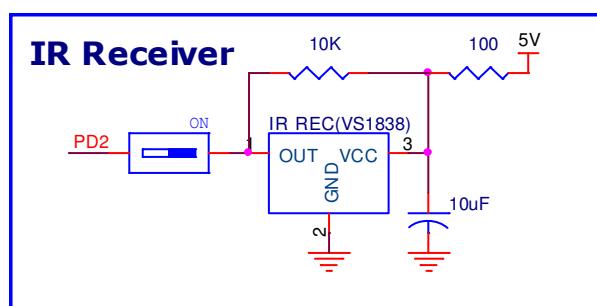


Conceptual view of how IR transmitter receiver pair works

VS1838 connected to microcontroller PD2 via DIP switch. PD2 pin can be used to trigger INT0, This help in simplifying firmware design.



IR Receiver

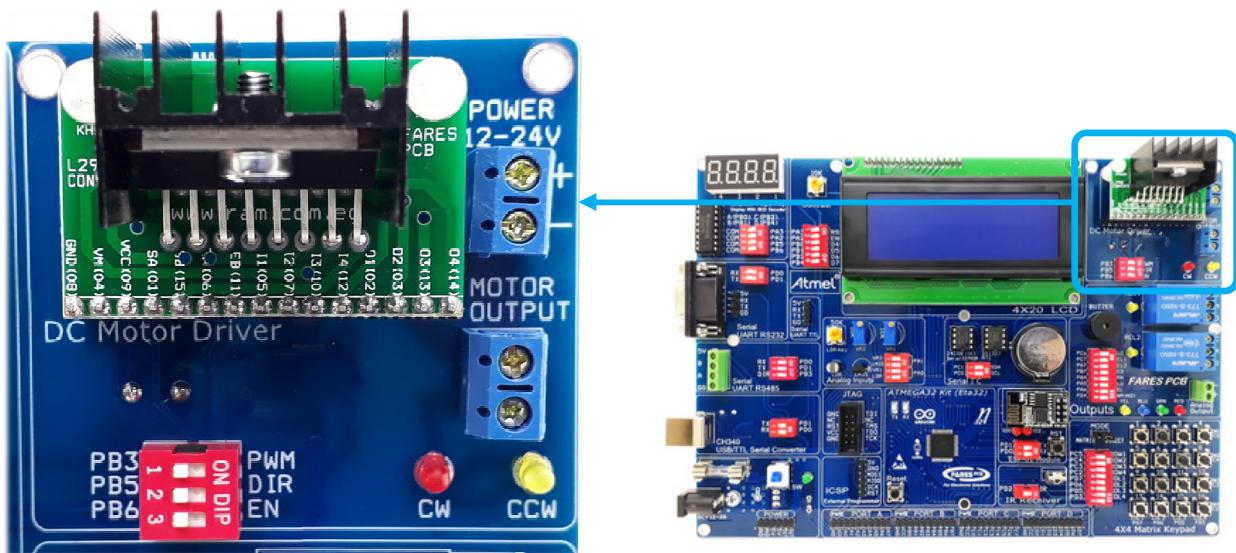


IR Receiver (Schematic)

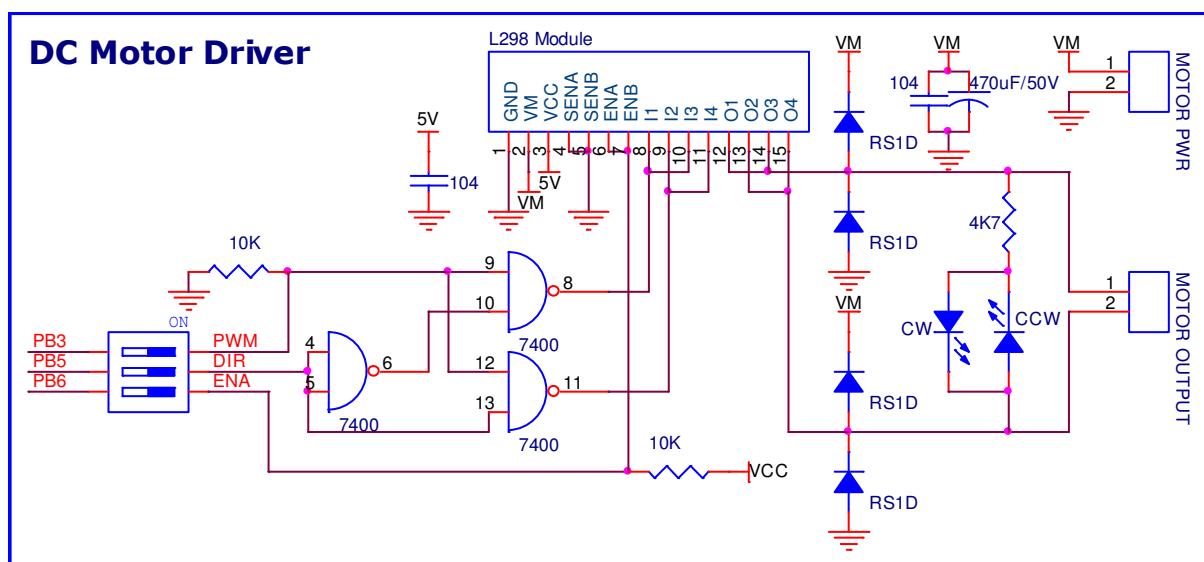
DC Motor Driver

DC motors have many applications in today's field of engineering and technology such as robots , air compressors , lifts , elevators , ... etc. If you plan on working with robots or just building moving objects you will eventually need to learn how to control a DC motor. One of the easiest and most inexpensive ways to control DC motors is to interface L298N motor driver with microcontroller. It can control both speed and direction of two 2A motors or one 4A motor.

Eta32 includes 4 ampere DC motor driver based on L298N. Circuit is designed to simplify written code to control speed and direction. There are three control signals required to drive motor PWM , DIR and EN. These signals are connected to microcontroller ports PB3, PB5, PB6 respectively via 3 way DIP switch.



DC Motor Driver



DC Motor Driver (Schematic)

| Motor Driver Control Signal | Microcontroller pin | Arduino Pin |
|-----------------------------|---------------------|-------------|
| PWM | PB3 | 3 |
| DIR | PB5 | 5 |
| EN | PB6 | 6 |

PWM signal is used to control speed. With PWM the motor is sent a series of pulses. each pulse is of the full voltage supplied. The width of pulses are varied to control the motor speed, pulses with a narrow width will cause the motor to rotate quite slowly. Increasing the pulse width will increase the speed of the motor. The average voltage applied to DC motor will depend on duty cycle. Duty cycle is the ratio of the ON time signal to the total period time. Duty cycle is usually expressed in percent. 50% means an average voltage of about half the supplied voltage. In order to run motor at full speed you just set PWM signal high for all period time Duty cycle = 100%. To stop the motor completely you just set PWM to low for all period time. PWM signal is derived from port PB3 which can be functioned as a PWM output.

DIR signal is used to control motor direction of rotation. To reverse direction just reverse the DIR state from high to low or vice versa. DIR signal connected to microcontroller port pin PB5 via DIP switch.

EN signal is used to enable driver. This signal can be omitted. To reverse direction just reverse the DIR state from high to low or vice versa. EN signal connected to microcontroller port pin PB6 via DIP switch.

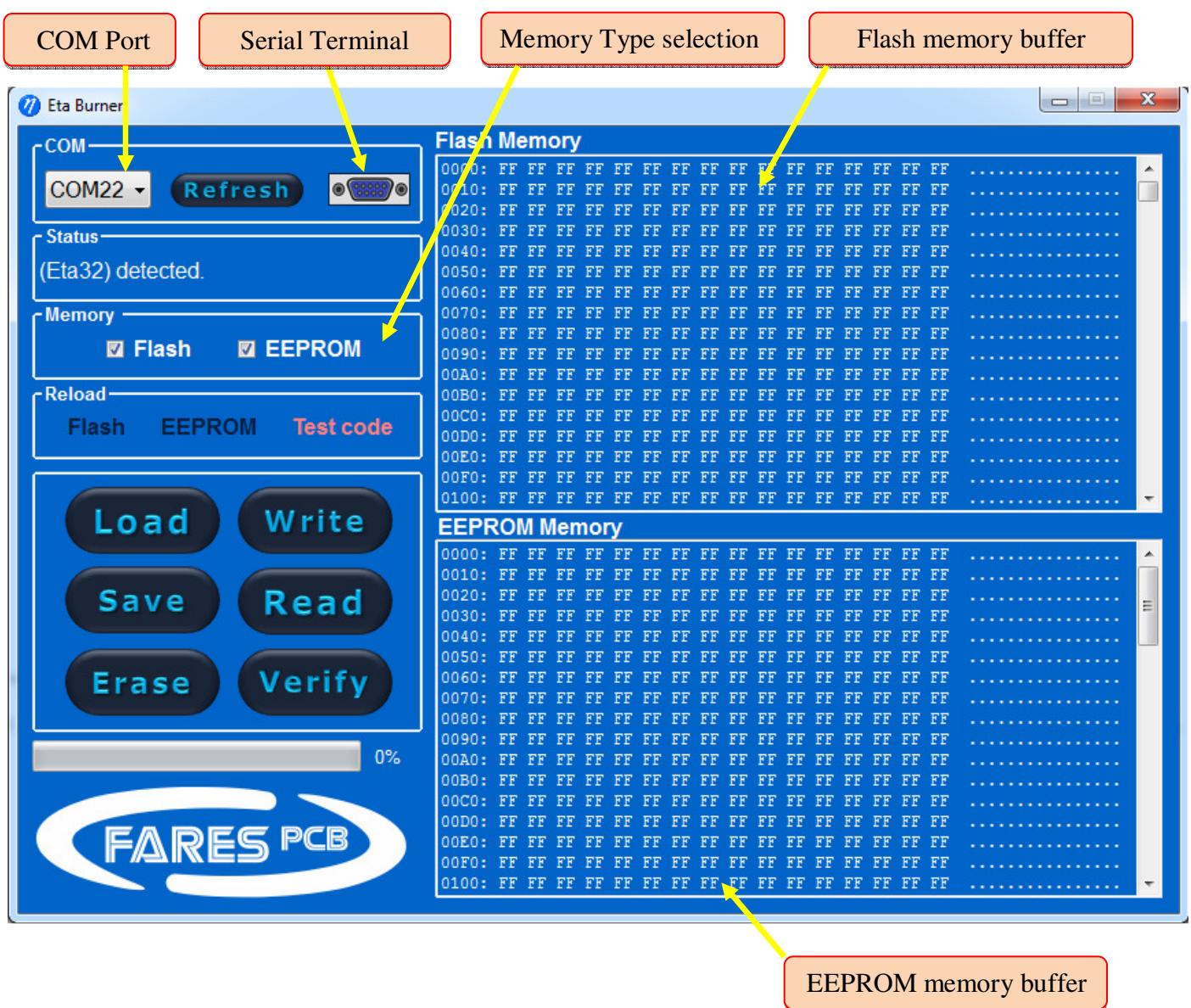
With PWM the motor is sent a series of pulses. each pulse is of the full voltage supplied. The width of pulses are varied to control the motor speed, pulses with a narrow width will cause the motor to rotate quite slowly. Increasing the pulse width will increase the speed of the motor. The average voltage applied to DC motor will depend on duty cycle. Duty cycle is the ratio of the ON time signal to the total period time. Duty cycle is usually expressed in percent.

There are three ways to burn your hex code on **Eta32** kit.

- 1) **Eta Burner** software tool (Eta32 bootloader from FARESPCB).
 - 2) Arduino IDE(Arduino bootloader).
 - 3) External programmer (ICSP or JTAG).

Programming using *Eta Burner* tool

Eta32 kit does not require any external burners to download hex code. it comes with Eta32 bootloader, which interfaces **Eta Burner** Software tool (from FARESPCB). **Eta Burner** is used to download/upload hex code from/to ROM and EEPROM contents of ATMEGA32A microcontroller.



Eta Burner tool is developed specifically for **Eta** kits. It detects any Eta kit automatically just after opening it or by clicking **Refresh** button. Select memory type (Flash, EEPROM or both) before any programming operation.

Use “**Load**” button to load hex file to buffer related to selected memory. Use “**Save**” button to store data in the buffer related to selected memory into a hex file (Intel format).

Use “**Write**” button to burn loaded hex file to selected memory on microcontroller.

Use “**Read**” button to read hex code from selected memory of microcontroller.

Use “**Erase**” button to erase selected memory of microcontroller.

Use “**Verify**” button to compare hex code from selected memory of microcontroller to the related memory buffer.

Fuse settings

By default, fuse bytes are set to the proper values as shown in table

| Fuse byte | Hex value |
|-----------|-----------|
| Low | 9F |
| High | CA |

Brown-out Detector(BOD): Enabled (2.7V).

Clock Source: External crystal oscillator.

Boot reset vector is selected. Bootloader occupies 2K bytes from flash memory

Note:

If fuse bytes are changed to any values other than those shown in the previous table, they must be reprogrammed to the proper values before using **Eta32** programming tool.

After programming is completed, a reset operation is performed to ensure microcontroller correct operation.

Warning:

Random fuse settings changing is risky. You should take special care while changing these settings. Incorrect fuse settings may cause incorrect microcontroller functioning.

Programming using Arduino IDE

If Arduino software tool is your preferred IDE, follow the next steps to add **Eta32** kit to Arduino boards,

1 - Upload Arduino bootloader firmware (ArduinoBL.hex) included in CD using any external programmer such as USBasp programmer. (See programming using external programmer).

Fuse Bytes must be set as shown in table below

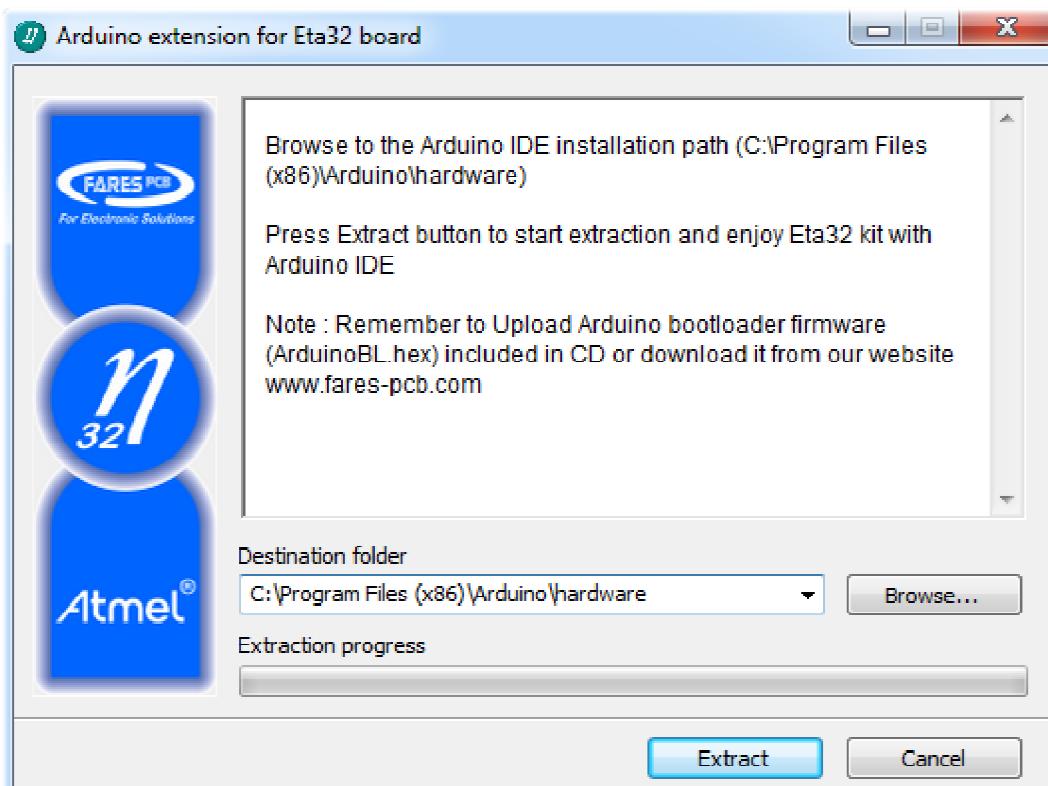
| Fuse byte | Hex value |
|-----------|-----------|
| Low | 9F |
| High | CE |

2 - Download Arduino IDE from the following link

<https://www.arduino.cc/en/Main/Software>

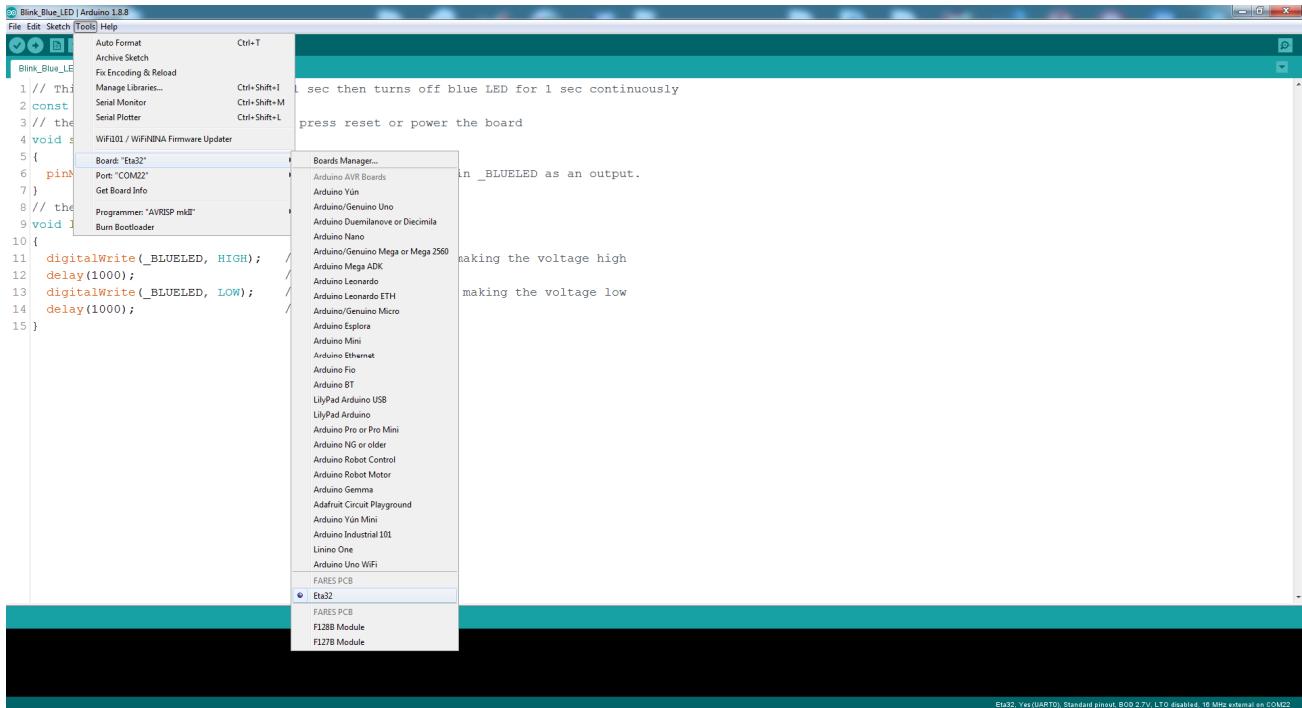
3 - Close Arduino IDE.

4 - Run the self-extracted file(**Eta32ArduinoExtension.exe**)included in CD, and browse to the Arduino IDE installation path (C:\Program Files (x86)\Arduino\hardware), and press Extract button to start extraction.



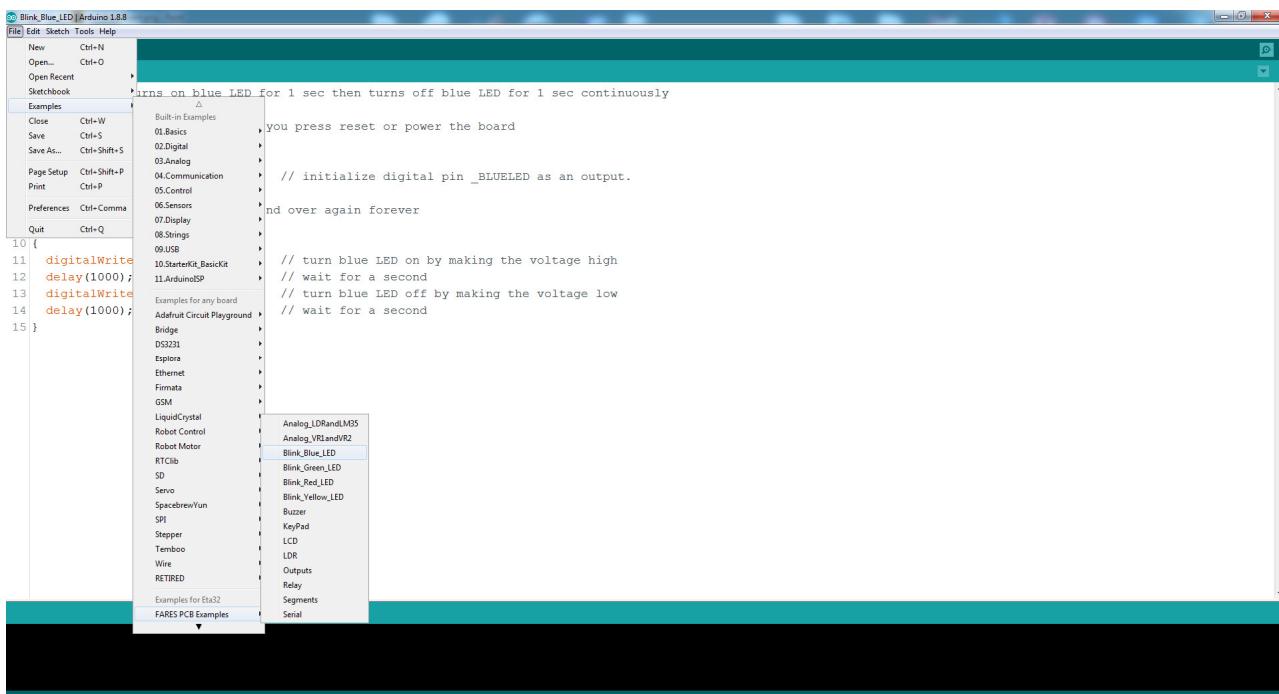
5 –After extraction, **Eta32** board is added to Arduino Boards. You can select Eta32 board as following:

Tools >> Board:>> Eta32.



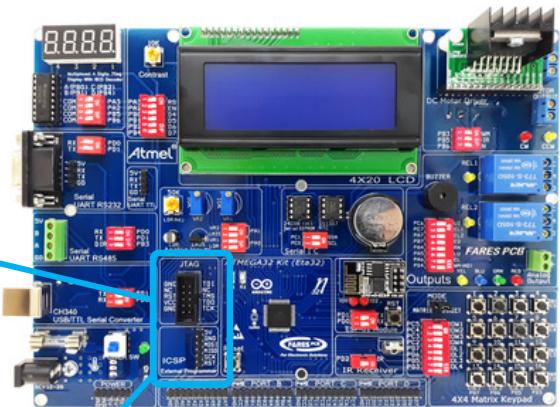
6 - Also, you will find many Arduino example codes to test all units in kit. You can go to:

File >> Examples >> Examples for Eta32.



Programming using external programmer

Eta32 is designed mainly to be programmed using bootloader. However, it offers ICSP and JTAG sockets to enable programming and debugging using any external programmer that supports standard 6 pin ICSP socket or JTAG connector, such as Atmel-ICE and USBasp programmer. Use external programmer to download your own application code or even a bootloader firmware such as Arduino bootloader.



For more details about programming using USBasp programmer, please refer to USBasp CD from FARESPCB products from this link:

<https://fares-pcb.com/product/usbasp-avr-programmer/>

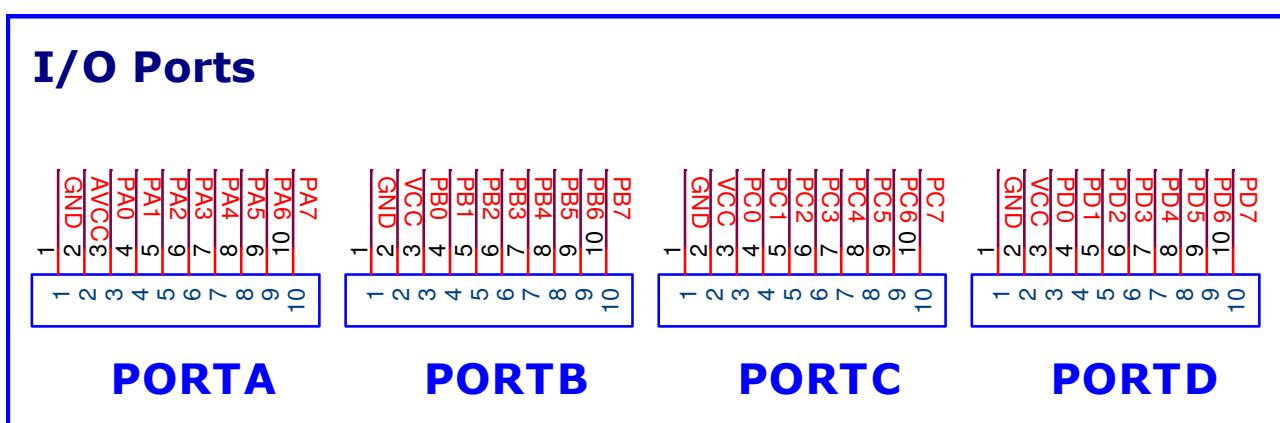
I/O PORT

In addition to the popular built-in circuits and devices included in **Eta32** kit, all microcontroller port pins are brought out for external using via header sockets.

External connectors are grouped into 4 units. Each represents one microcontroller port (8 I/O), in addition to GND and 5V for powering external circuits except PORTA which provides analog supply GND and AVCC. AVCC is the same as VCC but is filtered from high frequency noise. Use AVCC to supply power for external analog circuits.



Microcontroller port pins



Microcontroller port pins (Schematic)

Note:

Please remember to disconnect DIP switches associated to circuits attached to the port pins intended to use.

Eta32 Pin mapping table

| ATMEGA32A Pin | Arduino Pin | ETA32 Function |
|---------------|-------------|---|
| PA0 | A0 | VR1 / LDR |
| PA1 | A1 | VR2 / LM35 |
| PA2 | A2 | 7 Seg (COM2) / LCD (EN) |
| PA3 | A3 | 7 Seg (COM1) / LCD (RS) |
| PA4 | A4 | LED (Green) |
| PA5 | A5 | LED (Blue) |
| PA6 | A6 | LED (Yellow) |
| PA7 | A7 | Relay2 |
| PB0 | 0 | 7 Seg (Bit0) / LCD (D4) |
| PB1 | 1 | 7 Seg (Bit1) / LCD (D5) |
| PB2 | 2 | 7 Seg (Bit2) / LCD (D6) |
| PB3 | 3 | DC Motor (PWM) |
| PB4 | 4 | 7 Seg (Bit3) / LCD (D7) |
| PB5 | 5 | 7 Seg (COM3) / DC Motor (DIR) / ISP(MOSI) |
| PB6 | 6 | 7 Seg (COM4) / DC Motor (ENA) / ISP(MISO) |
| PB7 | 7 | LED (Red) / ISP(SCK) |
| PC0 | 16 | I ² C (SCL) |
| PC1 | 17 | I ² C (SDA) |
| PC2 | 18 | Keypad (Row4) / JTAG (TCK) |
| PC3 | 19 | Keypad (Row3) / JTAG (TMS) |
| PC4 | 20 | Keypad (Row2) / JTAG (TDO) |
| PC5 | 21 | Keypad (Row1) / JTAG (TDI) |
| PC6 | 22 | Buzzer |
| PC7 | 23 | Relay1 |
| PD0 | 8 | RX |
| PD1 | 9 | TX |
| PD2 | 10 | IR Receiver |
| PD3 | 11 | Keypad (Col4) |
| PD4 | 12 | Analog Output (PWM) |
| PD5 | 13 | Keypad (Col3) |
| PD6 | 14 | Keypad (Col2) |
| PD7 | 15 | Keypad (Col1) |

HOW TO START?

Step1

Install USB driver for CH340.

Step2

Install **Eta Burner** Software tool.

Step3

Connect USB cable to **Eta32** kit and run **Eta Burner** which starts to search COM ports and detects kit automatically.

If there's a problem in connection, please check USB cable connection to kit and click "Re-Connect" button to try again.

Step4

Now you are ready to upload your hex code and enjoy working with **Eta32** kit. Refer to Programming Using **Eta Burner** Tool for more details about using its IDE.

If it is the first time to use **Eta32** kit, you should perform some test operations on kit before working on it. The CD included with package contains the firmware code required for testing all modules in kit. So, it's recommended to upload this test code before going to your own application firmware to ensure correct functioning.

Upon burning "**Eta32_Test.hex**" code user can test

- Keypad in matrix mode.
- 4X20 LCD.
- Outputs unit (Relays, LEDs, Buzzer and Analog output).
- 7segment display.
- UART serial operation.
- Analog input.
- Serial IIC (24C08 and DS1307).
- DC Motor Driver.

Some hardware settings before testing

1. Enable DIP switches of all modules on kit except Analog inputs and 7segment display.
2. Set keypad to matrix mode (Set KB Mode to Matrix position).
3. Plug in USB cable.
4. Turn on power switch.
5. Open **Eta Burner** soft.
6. Click "LOAD" button to load the test code included in CD (**Eta32_Test.hex**) or just click Reload Test Code button.
7. Click Write button.
8. After programming is completed the microcontroller is reset automatically.

After power on or reset operation the test sequence is

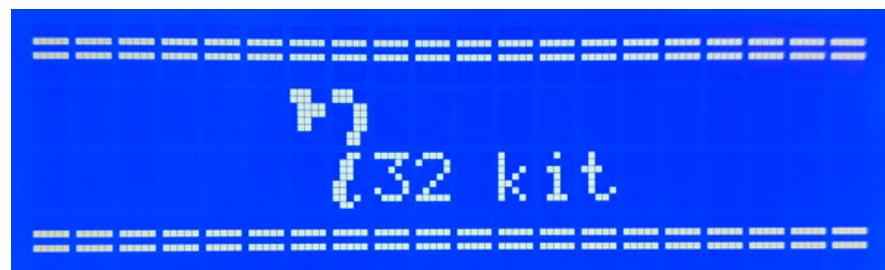
1 - Serial module transmits this message to serial port

"FARES PCB Co."
"ATMEL AVR Development kit."
"Eta32 kit."
"Arduino IDE compatible."

Transmission baud rate is "9600"

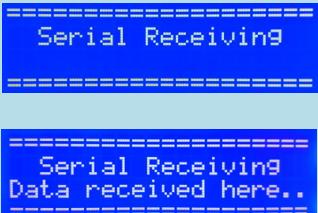
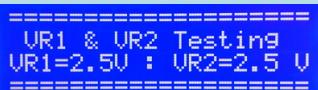
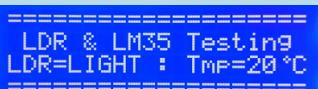
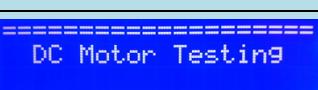
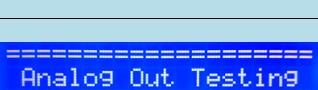
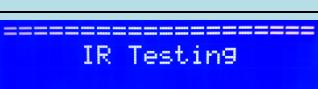
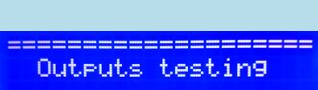
User may receive this statement by Hyper Terminal program or any other serial monitor software.

2 - LCD shows the following messages one by one.



3 – Each switch performs a test operation for a specified module on kit.
Follow the table below to test every module on kit.

| Switch | Serial port / LCD message | Test operation |
|--------|---|---|
| SW1 | "SW (1) pressed" ===== Yellow LED Test SW (1) Pressed ===== | Toggle Yellow LED |
| SW2 | "SW (2) pressed" ===== Blue LED Test SW (2) Pressed ===== | Toggle Blue LED |
| SW3 | "SW (3) pressed" ===== Green LED Test SW (3) Pressed ===== | Toggle Green LED |
| SW4 | "SW (4) pressed" ===== Red LED Test SW (4) Pressed ===== | Toggle Green LED |
| SW5 | "SW (5) pressed" ===== Relay1 Test SW (5) Pressed ===== | Toggle Relay1 |
| SW6 | "SW (6) pressed" ===== Relay2 Test SW (6) Pressed ===== | Toggle Relay2 |
| SW7 | "SW (7) pressed" ===== 7 Segment display Testing ===== | 7 Segment test Each Digit Counts up from 0 To 9 one by one. Auto reset will be applied after testing 7segment to reinitialize LCD. You can cancel testing by pressing any switch. A restart operation is initiated. |
| SW8 | "SW (8) pressed" ===== 24C08 Testing 24C08 Tested OK ===== | Serial EEPROM (24C08) Testing First 100 addresses will be checked. No data changed after testing. If damaged memory or no memory found, a test error message is shown. |

| | | |
|------|---|--|
| SW9 | "SW (9) pressed" | Real Time Clock (DS1307) testing Every time you press SW9 the RTC will be reinitialized to the same date and time. Time 12:59:00 AM Date 01/01/20 |
| |  | |
| SW10 | "SW (10) pressed" | UART Serial data receiving test Baud rate is 9600. Received characters displayed in the third line. Don't transmit characters bulky. Send characters one by one as characters displaying on LCD take some time. |
| |  | |
| SW11 | "SW (11) pressed" | VAR1 and VR2 testing Please Enable channel 1,3 only of analog inputs DIP switch. Changing the trim pots from end to end give a full range of 5V. |
| |  | |
| SW12 | "SW (12) pressed" | LDR and Temperature sensor LM35 testing Please Enable channel 2,4 only of analog inputs DIP switch. Relay1 is turned on when low light level is detected otherwise it's turned off. Use LDR Adj potentiometer to set light detecting sensitivity. |
| |  | |
| SW13 | "SW (13) pressed" | DC motor driver circuit testing Motor runs for 3 seconds in CW direction. Stop 0.5 second. Motor runs for 3 seconds in CCW direction. |
| |  | |
| SW14 | "SW (14) pressed" | Analog output testing LCD shows the analog output voltage vs the PWM duty cycle signal. PWM = 0 > 100 % Analog output = 0 > 5V Pressing SW14 increments the PWM duty cycle. |
| |  | |
| SW15 | "SW (15) pressed" | Infrared receiver testing If IR receiver detects an IR signal (38KHz) then the yellow LED is turned on. |
| |  | |
| SW16 | "SW (16) pressed" | Outputs Testing All digital outputs are tested continuously as folowing: Relay1 turned on for 400 msec then off. Relay2 turned on for 400 msec then off. Yellow LED turned on for 400 msec then off. Blue LED turned on for 400 msec then off. Green LED turned on for 400 msec then off. Red LED turned on for 400 msec then off. |
| |  | |

Special Thanks to:

Eng. / Ahmed Ibrahim Ahmed
ah.ahmed@nu.edu.eg

Copyright © 2020 by FARESPCB

For our full range of products see our website at <http://www.fares-pcb.com>
 If you have any technical questions about our products, e-mail us at
www.support@fares-pcb.com

FARESPCB co. (Head office)

164 Tahrir st,
 Bab El-Louq,
 Cairo,
 Egypt.
 Tel: +202-23904484
 Mob:+201000652977

FARESPCB co. (Branch office)

4 El-Shabrawy st,
 Road El-Farag,
 Cairo,
 Egypt.
 Tel: +202-24577118
 Mob:+201022457902

FARESPCB Co reserves the right to make changes in circuit design, software and/or specifications at any time without prior notification. For the most up-to-date information, please visit our web site at <http://www.fares-pcb.com>

Information furnished by FARESPCB is believed to be accurate and reliable. However, FARESPCB assumes no responsibility arising from the use of the specifications described.

Warranty: FARESPCB™ warrants its products against defects in materials and workmanship for a period of 30 days. If you discover a defect, we will, at our option, repair or replace your product or refund your purchase price. This warranty does not cover products that have been physically abused or misused in any way.

Distributor:

RAM Electronics
 32 El Falaky St. Bab El Louk
 Tahrir, Cairo
 Egypt.
 Tel: +202-27960551
www.ram.com.eg
Sales@ram-electronics.com

