SPC

SMART PERIPHERAL CONTROLLER

Low Cost Motor Controller

Table Of Contents

1. INTRODUCTION

Smart Peripheral Controller / SPC LOW COST MOTOR CONTROLLER is a DC and stepper motor controller module which is compact, reliable, and compatible for robotic applications. This module can be used to control the direction and speed of 4 DC motors using the Pulse Width Modulation (PWM) method or 2 stepper motors using full-step or half-step. This module is equipped with quad full H-Bridge driver, UART TTL interface, and I^2C interface, so that it can easily be connected with other systems.

1.1. SPC LOW COST MOTOR CONTROLLER SPECIFICATION

SPC LOW COST MOTOR CONTROLLER specification is as follows:

- The module requires 4.8 – 5.4 VDC power supply.
- The motor requires 8 – 36 VDC power supply.
- Uses a A3988 motor driver IC.
- Each driver's maximum continuous current is 1.2 A.
- Can be used for unipolar or bipolar stepper motors.
- Input/Output pins are compatible with TTL and CMOS voltage level.
- Equipped with UART TTL and I^2C interface.
- Using I^2C, SPC LOW COST MOTOR CONTROLLER can be cascaded up to 8 modules.

1.2. SUGGESTED SYSTEM

Suggested system for SPC LOW COST MOTOR CONTROLLER is as follows:

Hardware:

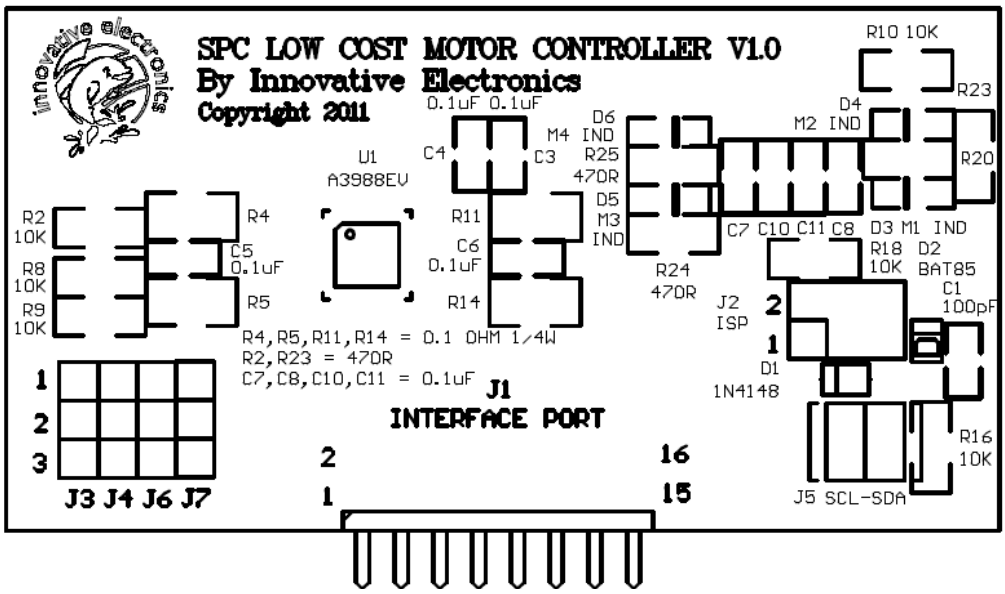
- PC™ AT™ Pentium® IBM™ Compatible with USB port.
- DT-AVR Low Cost Series.
- DVD-ROM Drive and Hard disk.

Software:

- Windows® XP Operating System.
- CodeVisionAVR©.
- Program CD/DVD contents:
  Contoh_i2c folder, contoh_uart folder, A3988.pdf, and SPC Low Cost Motor Controller Manual.pdf.

2. SPC LOW COST MOTOR CONTROLLER HARDWARE

2.1. SPC LOW COST MOTOR CONTROLLER COMPONENT LAYOUT

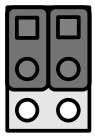
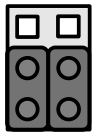


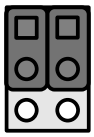
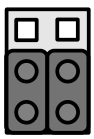
2.2. CONNECTORS AND JUMPERS CONFIGURATIONS

INTERFACE PORT (J1) connector functions as a connector for module power supply input, motor power supply input, UART TTL interface, I^2C interface, and motors.

Pin	Name	Function
1	M11	1st Output from H-Bridge **M1** pair
2	M12	2nd Output from H-Bridge **M1** pair
3	M21	1st Output from H-Bridge **M2** pair
4	M22	2nd Output from H-Bridge **M2** pair
5	M31	1st Output from H-Bridge **M1** pair
6	M32	2nd Output from H-Bridge **M3** pair
7	M41	1st Output from H-Bridge **M1** pair
8	M42	2nd Output from H-Bridge **M4** pair
9	MGND	Ground reference for motor power supply
10	VM	Connected to motor power supply (8 – 36 Volts)
11	SCL	I^2C-bus clock input
12	SDA	I^2C-bus data input / output
13	RXD	TTL serial level input to SPC module
14	TXD	TTL serial level output from SPC module
15	PGND	Ground reference for SPC module power supply
16	VIN	Connected to power supply (4.8 – 5.4 Volts)

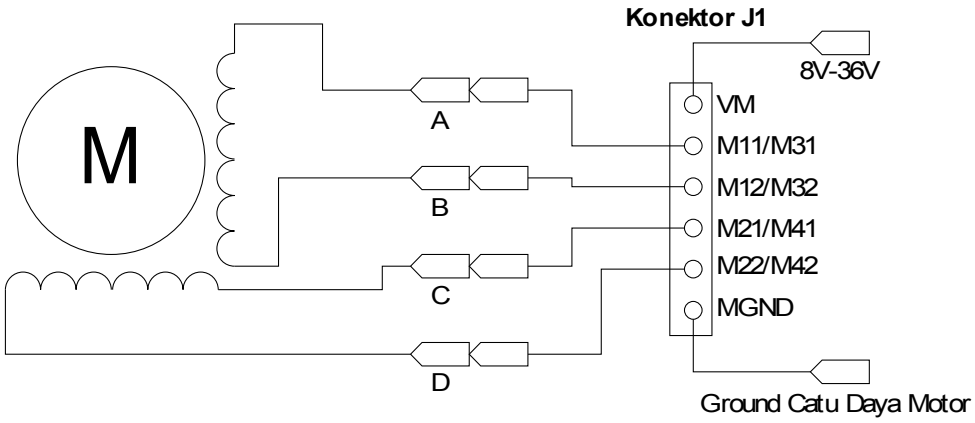
J3, J4, J6, and J7 jumpers are used to select operation mode for each H-Bridge on the SPC module.

M1 & M2 Functions	J3 & J4 Position
DC Motor Controller	 J3 J4
Stepper Motor Controller	 J3 J4

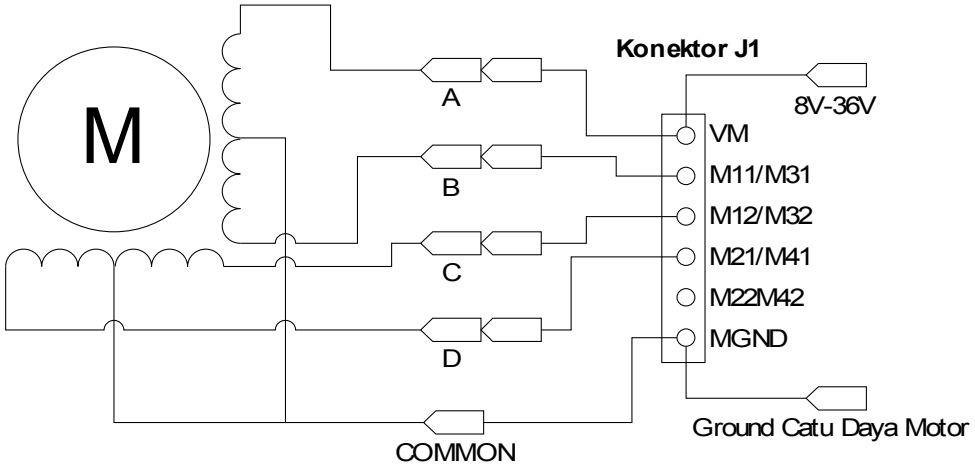
M3 & M4 Functions	J6 & J7 Position
DC Motor Controller	 J6 J7
Stepper Motor Controller	 J6 J7

Pay attention to the type of stepper motor connected to SPC LOW COST MOTOR CONTROLLER because each type has its own connection. SPC LOW COST MOTOR CONTROLLER can be utilized for 3 types of stepper motor: Bipolar, 5 cables Unipolar, and 6 cables Unipolar. The following are the connection examples for each stepper motor type:

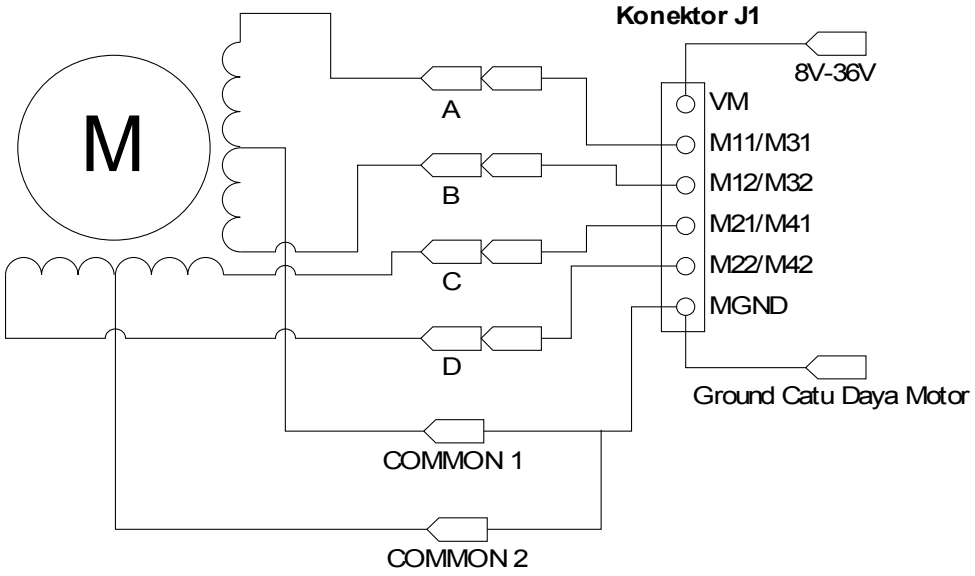
Bipolar



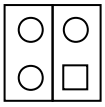
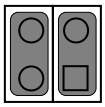
Unipolar 5 kabel



Unipolar 6 kabel



SCL-SDA (J5) jumpers are used to activate pull-up resistors for SDA and SCL on I^2C interface.

Jumper SCL-SDA J5	Function
 SCL SDA	Pull-up inactive (jumpers disconnected)
 SCL SDA	Pull up active (jumpers connected)

Important!
If more than one module is connected to I^2C-bus, then only one set of SCL-SDA (J5) jumpers needs to be connected.

I^2C address configuration can be done through UART TTL interface.

M1 IND (D3), M2 IND (D4), M3 IND (D5), and M4 IND (D6) LEDs function as motor condition indicator.

3. SPC LOW COST MOTOR CONTROLLER INTERFACE

SPC LOW COST MOTOR CONTROLLER has UART TTL and I^2C interfaces that can be used to receive commands or send data.

3.1. UART TTL INTERFACE

UART TTL communication parameters are as follows:

- 38400 bps
- 8 data bits
- 1 stop bit
- no parity bit
- no flow control

All commands sent through UART TTL interface begin with 1 byte data that contains **<command number>**, followed by (if needed) n-byte data command parameter.

If the command and parameters transmission succeeded, then SPC LOW COST MOTOR CONTROLLER will send **0x06** (Acknowledged/ACK). If the command is not recognized, the SPC LOW COST MOTOR CONTROLLER will send **0x15** (Not Acknowledged/NCK). If the command is recognized but the command parameter is incorrect, then SPC LOW COST MOTOR CONTROLLER will not send any feedback.

If the command sent is a command requesting data from SPC LOW COST MOTOR CONTROLLER module, then SPC LOW COST MOTOR CONTROLLER will send the data via TX TTL line.

A data parameter that has a range larger than 255 decimals (larger than 1 byte) will be sent in two steps. 1 byte MSB data is sent first and is followed by LSB data. For example: parameter <pulse delay> which has a range of 1 - 65535. If <pulse delay> has a value of 1500 then MSB byte will be 5 and LSB byte will be 220 ((5x256)+220=1500).

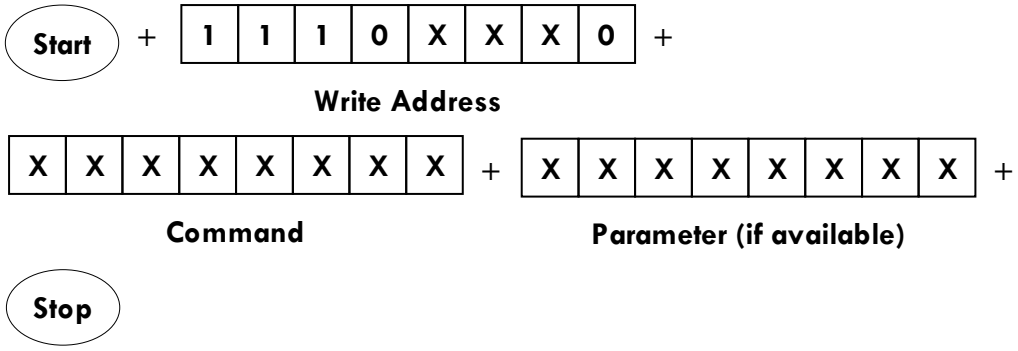
Available commands and parameters can be seen in **section 3.3**.

3.2. I²C INTERFACE

SPC LOW COST MOTOR CONTROLLER module has an I²C interface. In this interface, SPC LOW COST MOTOR CONTROLLER module acts as a slave with an address that as been determined via UART command (see **section 3.3.9**). I²C interface on SPC LOW COST MOTOR CONTROLLER module supports bit rate up to a maximum rate of 50 kHz.

All commands sent through I²C interface begin with **start condition**, followed by 1 byte of SPC LOW COST MOTOR CONTROLLER module address. After the address is sent, the master must send 1 byte data that contains **<command number>**, followed by (if needed) n-byte command parameter data. After all command parameters have been sent, the command is ended with **stop condition**.

The following is the sequence that must be done to send a command via I²C interface.

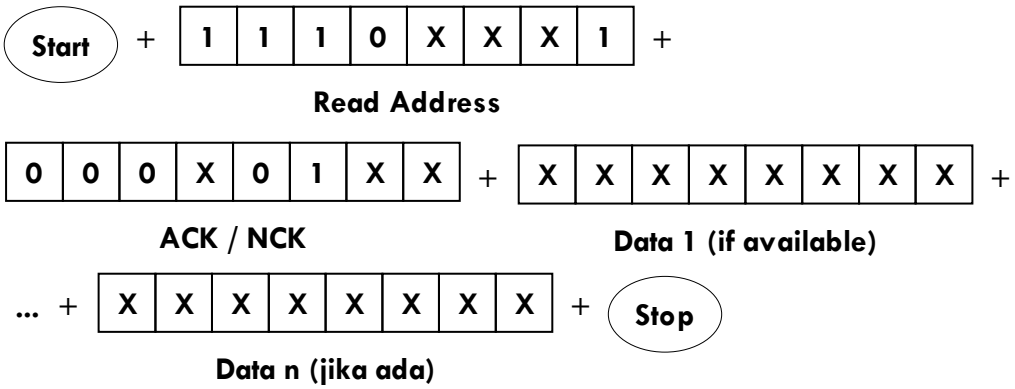


If the command and parameters transmission succeeded, then SPC LOW COST MOTOR CONTROLLER will write a hexadecimal response **0x06** (Acknowledged/ACK) in its I²C buffer. But if the command is not recognized or the command parameter is incorrect then the SPC LOW COST MOTOR

CONTROLLER will write a hexadecimal response of **0x15** (Not Acknowledged/NCK) on its I²C buffer.

Master can send a read command to read the <**ACK**/**NCK**> response. If the command sent is a command that requests data from SPC LOW COST MOTOR CONTROLLER module, then those data can be read after reading the response by using the read data command.

The following is the sequence that must be done to read response and/or data from SPC LOW COST MOTOR CONTROLLER.



A data parameter that has a range larger than 255 decimals (larger than 1 byte) will be sent in two steps. 1 byte MSB data is sent first and is followed by LSB data. For example: parameter <pulse delay> which has a range of 1 - 65535. If <pulse delay> has a value of 1500 then MSB byte will be 5 and LSB byte will be 220 ((5x256)+220=1500).

3.3. COMMAND SET

The following is a complete list of commands in UART and I²C interface:

3.3.1. DC FORWARD

Function	Controls DC motor forward rotation
Command	**0x30**
Parameter	<Motor number> 1 → DC motor connected to M1 2 → DC motor connected to M2 3 → DC motor connected to M3 4 → DC motor connected to M4 <pwm level> 0 - 255 → assigned duty cycle percentage (0 = 0%; 255 = 100%)
Response	0x06 → if command is recognized 0x15 → if command is not recognized
Delay between Command and Response	10 µs
Description	• Indicator LED light intensity for each H-Bridge (M1, M2, M3, and M4) will match the PWM value given. If the PWM value is 0 then the indicator LED will turn off. When the PWM value is 255 then indicator LED will light up with the highest intensity.

	• On forward condition, indicator LED will also blink. • On forward condition, Mn1 (n is the H-Bridge number) will produce voltage proportional to the PWM value while Mn2 will be connected with MGND. • Motor direction and PWM value **will not** be saved in EEPROM. When the SPC module is powered on, PWM values of each H-bridge is 0 (zero) and the motor will be in a stop condition (all-4 indicator LEDs will blink faintly every 2 seconds).

Example with UART interface to control the forward speed of DC motor connected to M1. If the desired duty cycle is 50% (0.5 * 255 = 128) or equal to 128 decimal or 0x80 hexadecimal:

User : 0x30 0x01 0x80
SPC : 0x06

The following is a pseudo code example, to use this command with I^2C interface (I^2C address example = 0xE0):

```
i2c_start();          // Start Condition
i2c_write(0xE0);      // Write SPC Low Cost Motor module
i2c_write(0x30);      // "DC Forward" command
i2c_write(0x01);      // Motor number
i2c_write(0x80);      // PWM value
i2c_stop();           // Stop Condition

delay_us(10);         // delay 10 us

i2c_start();          // Start Condition
i2c_write(0xE1);      // Read SPC Low Cost Motor module
temp = i2c_read(0);   // Data Acknowledgment
i2c_stop();           // Stop Condition
```

3.3.2. DC REVERSE

Function	Controls DC motor reverse rotation
Command	**0x31**
Parameter	<Motor number> 1 → DC motor connected to M1 2 → DC motor connected to M2 3 → DC motor connected to M3 4 → DC motor connected to M4 <pwm level> 0 - 255 → assigned duty cycle percentage (0 = 0%; 255 = 100%)
Response	0x06 → if command is recognized 0x15 → if command is not recognized
Delay between Command and Response	10 µs
Description	• Indicator LED light intensity for each H-Bridge (M1, M2, M3, and M4) will match the PWM value given. If the PWM value is 0 then the indicator LED will turn off. When the PWM value is 255 then indicator LED will

	light up with the highest intensity. ● On reverse condition, Mn2 (n is the H-Bridge number) will produce voltage proportional to the PWM value while Mn2 will be connected with MGND. ● Motor direction and PWM value **will not** be saved in EEPROM. When the SPC module is powered on, PWM values of each H-bridge is 0 (zero) and the motor will be in a stop condition (all-4 indicator LED will blink faintly every 2 seconds).

Example with UART interface to control the reverse speed of DC motor connected to M1. If the desired duty cycle is 20% (0.25 * 255 = 64) or equal to 64 decimal and 0x40 hexadecimal:

User : 0x31 0x01 0x40
SPC : 0x06

The following is a pseudo code example, to use this command with I^2C interface (I^2C address example = 0xE0):

```
i2c_start();          // Start Condition
i2c_write(0xE0);      // Write SPC Low Cost Motor module
i2c_write(0x31);      // "DC Reverse" command
i2c_write(0x01);      // Motor number
i2c_write(0x40);      // PWM value
i2c_stop();           // Stop Condition

delay_us(10);         // delay 10 us

i2c_start();          // Start Condition
i2c_write(0xE1);      // Read SPC Low Cost Motor module
temp = i2c_read(0);   // Data Acknowledgment
i2c_stop();           // Stop Condition
```

3.3.3. DC STOP

Function	Stops DC motor
Command	**0x32**
Parameter	<Motor number> 1 → DC motor connected to M1 2 → DC motor connected to M2 3 → DC motor connected to M3 4 → DC motor connected to M4
Response	0x06 → if command is recognized 0x15 → if command is not recognized
Delay between Command and Response	10 µs
Description	● On stop condition, Mn1 and Mn2 will be in a three state / high impedance condition.

Example with UART interface to stop DC motor connected to M1:

User : 0x32 0x01
SPC : 0x06

The following is a pseudo code example, to use this command with I^2C interface (I^2C address example = 0xE0):

```
i2c_start();          // Start Condition
i2c_write(0xE0);      // Write SPC Low Cost Motor module
i2c_write(0x32);      // "DC Stop" command
i2c_write(0x01);      // Motor number
i2c_stop();           // Stop Condition

delay_us(10);         // delay 10 us

i2c_start();          // Start Condition
i2c_write(0xE1);      // Read SPC Low Cost Motor module
temp = i2c_read(0);   // Data Acknowledgment
i2c_stop();           // Stop Condition
```

3.3.4. DC ALL STOP

Function	Stops all DC or stepper motors simultaneously
Command	**0x33**
Parameter	-
Response	0x06 → if command is recognized 0x15 → if command is not recognized
Delay between Command and Response	10 μs
Description	• This command will cause all H-bridges to be in a stop condition. • If **All** H-Bridges are in stop condition, then indicator LED of each H-Bridge will blink faintly every 2 seconds.

Example with UART interface to stop all DC or stepper motors connected to M1, M2, M3, and M4 simultaneously:

User : 0x33
SPC : 0x06

The following is a pseudo code example, to use this command with I^2C interface (I^2C address example = 0xE0):

```
i2c_start();          // Start Condition
i2c_write(0xE0);      // Write SPC Low Cost Motor module
i2c_write(0x33);      // "All Stop" command
i2c_stop();           // Stop Condition

delay_us(10);         // delay 10 us

i2c_start();          // Start Condition
i2c_write(0xE1);      // Read SPC Low Cost Motor module
temp = i2c_read(0);   // Data Acknowledgment
i2c_stop();           // Stop Condition
```

3.3.5. STEPPER CONTINUOUS RUN

Function	Controls stepper motor so that it rotates continuously
Command	**0x34**
Parameter	<Motor number> 1 → stepper motor connected to M1 and M2 2 → stepper motor connected to M3 and M4 <step type> 1 → Full-Step: motor will rotate 1 step every 1 pulse 2 → Half-Step: motor will rotate ½ step every 1 pulse <direction> 0 → motor will rotate clockwise 1 → motor will rotate counter clockwise <pulse delay> 1 - 65535 → Delay time between pulse to stepper motor. The smaller the pulse delay, the faster the stepper motor rotates
Response	0x06 → if command is recognized 0x15 → if command is not recognized
Delay between Command and Response	10 µs
Description	• If the stepper motor rotates to an opposite direction, then it means that the connection is reversed. To fix it, change the order of connection installation. • One pulse delay value represents delay time between pulse for about 1 ms.

Example with UART interface to run the stepper motor connected to M1 and M2 so that it rotates clockwise continuously, with a full-step step type, and the delay between pulse is about 100 ms (0x0064 hexadecimal):

User : 0x34 0x01 0x01 0x00 0x00 0x64
SPC : 0x06

The following is a pseudo code example, to use this command with I^2C interface (I^2C address example = 0xE0):

```
i2c_start();          // Start Condition
i2c_write(0xE0);      // Write SPC Low Cost Motor module
i2c_write(0x34);      // "Stepper Continuous Run" command
i2c_write(0x01);      // Motor number
i2c_write(0x01);      // Step type
i2c_write(0x00);      // Direction
i2c_write(0x00);      // MSB pulse delay
i2c_write(0x64);      // LSB pulse delay
i2c_stop();           // Stop Condition

delay_us(10);         // delay 10 us

i2c_start();          // Start Condition
i2c_write(0xE1);      // Read SPC Low Cost Motor module
temp = i2c_read(0);   // Data Acknowledgment
i2c_stop();           // Stop Condition
```

3.3.6. STEPPER PULSE COUNT RUN

Function	Controls stepper motor so that it rotates according to how many steps given
Command	**0x35**
Parameter	<Motor number> 1 → stepper motor connected to M1 and M2 2 → stepper motor connected to M3 and M4 <step type> 1 → Full-Step: motor will rotate 1 step every 1 pulse 2 → Half-Step: motor will rotate ½ step every 1 pulse <direction> 0 → motor will rotate clockwise 1 → motor will rotate counter clockwise <pulse delay> 1 - 65535 → Delay time between pulse to stepper motor. The smaller the pulse delay, the faster the stepper motor rotates <pulse count> 1 - 65535 → the number of pulse sent to stepper motor
Response	0x06 → if command is recognized 0x15 → if command is not recognized
Delay between Command and Response	10 µs
Description	● If the stepper motor rotates to an opposite direction, then it means that the connection is reversed. To fix it, change the order of connection installation. ● One pulse delay value represents delay time between pulse for about 1 ms. ● After the number of pulse that has been released matches the pulse count, stepper motor will automatically stop (on brake condition) while still maintaining motor torque (current is still flowing through stepper motor coils).

Example with UART interface to run the stepper motor connected to M1 and M2 so that it rotates clockwise 20 pulses (0x0014 hexadecimal), with a full-step step type, and the delay between pulses is about 1000 ms (0x03E8 hexadecimal):

User : 0x35 0x01 0x01 0x00 0x03 0xE8 0x00 0x14
SPC : 0x06

The following is a pseudo code example, to use this command with I^2C interface (I^2C address example = 0xE0):

```
i2c_start();          // Start Condition
i2c_write(0xE0);      // Write SPC Low Cost Motor module
i2c_write(0x35);      // "Stepper Pulse Count Run" command
i2c_write(0x01);      // Motor number
i2c_write(0x01);      // Step type
i2c_write(0x00);      // Direction
```

```
i2c_write(0x03);      // MSB pulse delay
i2c_write(0xE8);      // LSB pulse delay
i2c_write(0x00);      // MSB pulse count
i2c_write(0x14);      // LSB pulse count
i2c_stop();           // Stop Condition

delay_us(10);         // delay 10 us

i2c_start();          // Start Condition
i2c_write(0xE1);      // Read SPC Low Cost Motor module
temp = i2c_read(0);   // Data Acknowledgment
i2c_stop();           // Stop Condition
```

3.3.7. STEPPER BRAKE

Function	Stops the stepper motor while still maintaining motor torque (current is still flowing through stepper motor coils).
Command	**0x36**
Parameter	<Motor number> 1 → stepper motor connected to M1 and M2 2 → stepper motor connected to M3 and M4
Response	0x06 → if command is recognized 0x15 → if command is not recognized
Delay between Command and Response	10 μs
Description	• This command can be given after the Continuous Run command. • On brake condition, stepper motor will stop while still maintaining motor torque (current is still flowing through stepper motor coils). • On brake condition, indicator LED will lit up according to the last Run command.

Example with UART interface to stop the stepper motor connected to M1 and M2:

User : 0x36 0x01
SPC : 0x06

The following is a pseudo code example, to use this command with I^2C interface (I^2C address example = 0xE0):

```
i2c_start();          // Start Condition
i2c_write(0xE0);      // Write SPC Low Cost Motor module
i2c_write(0x36);      // "Stepper Brake" command
i2c_write(0x01);      // Motor number
i2c_stop();           // Stop Condition

delay_us(10);         // delay 10 us

i2c_start();          // Start Condition
i2c_write(0xE1);      // Read SPC Low Cost Motor module
temp = i2c_read(0);   // Data Acknowledgment
i2c_stop();           // Stop Condition
```

3.3.8. STEPPER STOP

Function	Stops the stepper motor (current doesn't flow through stepper motor coils)
Command	**0x37**
Parameter	<Motor number> 1 → stepper motor connected to M1 and M2 2 → stepper motor connected to M3 and M4
Response	0x06 → if command is recognized 0x15 → if command is not recognized
Delay between Command and Response	10 µs
Description	• This command can be given after Continuous Run, Pulse Count Run, or Brake command. • On stop condition, stepper motor will stop and there will be no current flowing through the motor coils. • Stop condition is the default condition when the SPC module is powered on

Example with UART interface:

User : 0x37 0x01
SPC : 0x06

The following is a pseudo code example, to use this command with I^2C interface (I^2C address example = 0xE0):

```
i2c_start();          // Start Condition
i2c_write(0xE0);      // Write SPC Low Cost Motor module
i2c_write(0x37);      // "Stepper Stop" command
i2c_write(0x01);      // Motor number
i2c_stop();           // Stop Condition

delay_us(10);         // delay 10 us

i2c_start();          // Start Condition
i2c_write(0xE1);      // Read SPC Low Cost Motor module
temp = i2c_read(0);   // Data Acknowledgment
i2c_stop();           // Stop Condition
```

3.3.9. SET I^2C ADDRESS

Function	Changes I^2C address
Command	**0x41**
Parameter	<0xAA> <0x55> <newAddress>
Response	0x06 → if command is recognized 0x15 → if command is not recognized
Delay between Command and Response	10 µs
Description	• This command can only be performed via UART communication line. • SPC module will use the new I^2C address after going through power off sequence.

	• The allowed I²C address <newAddress> can be seen in the table below. • If the new address given is incorrect, then the I²C address will not be changed (the previous address will be used). • The default I²C address is 0xE0. • I²C address data will be saved in EEPROM so it won't be erased when it's powered off.

I²C Address	
I²C Write Address	**I²C Read Address**
0xE0	0xE1
0xE2	0xE3
0xE4	0xE5
0xE6	0xE7
0xE8	0xE9
0xEA	0xEB
0xEC	0xED
0xEE	0xEF

Example with UART interface to change the I²C address from 0xE0 to 0xE2:

User : 0x41 0xAA 0x55 0xE2
SPC : 0x06

3.3.10. READ I²C ADDRESS

Function	Reads the current I²C address
Command	**0x42**
Parameter	-
Response	<I²CAddress> → if command is recognized 0x15 → if command is not recognized
Delay between Command and Response	10 µs
Description	• This command can only be performed via UART communication line. • SPC module's I²C address can also be seen through the number of blinks on the indicator LED when the module is powered on. • If the I²C address is 0xE0 then the indicator LED will blink once. If the I²C address is 0xE2 then the indicator LED will blink twice. If the I²C address is 0xE4 then the indicator LED will blink 3 times, and so on until I²C address 0xEE at which the indicator LED will blink 8 times.

Example with UART interface:

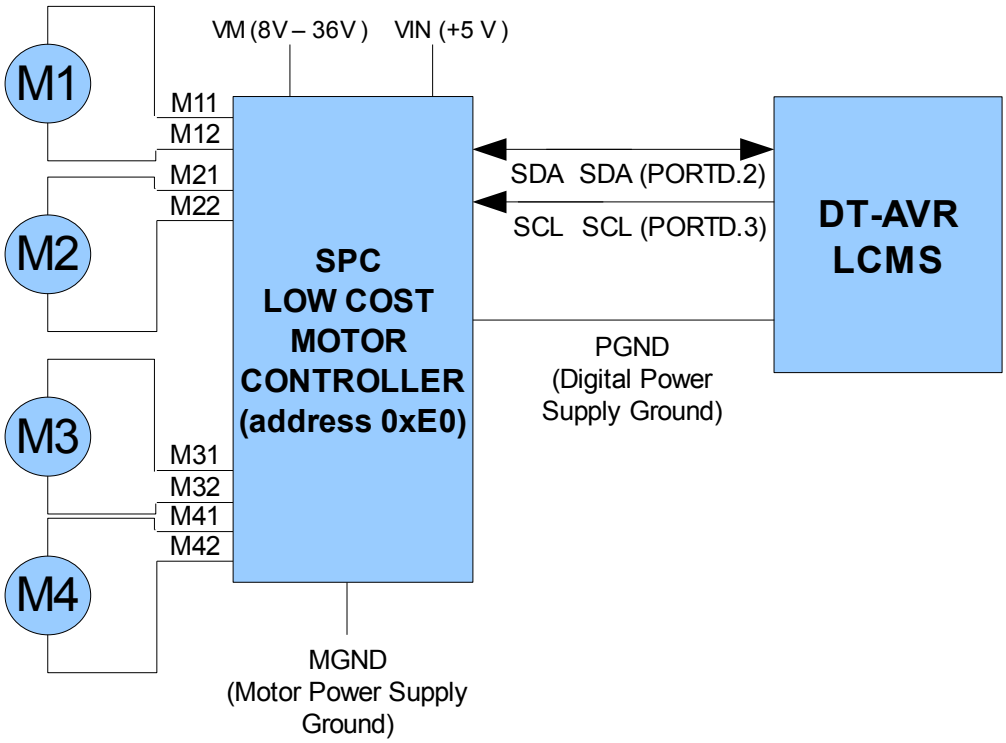
User : 0x42
SPC Module : <I2CAddress>

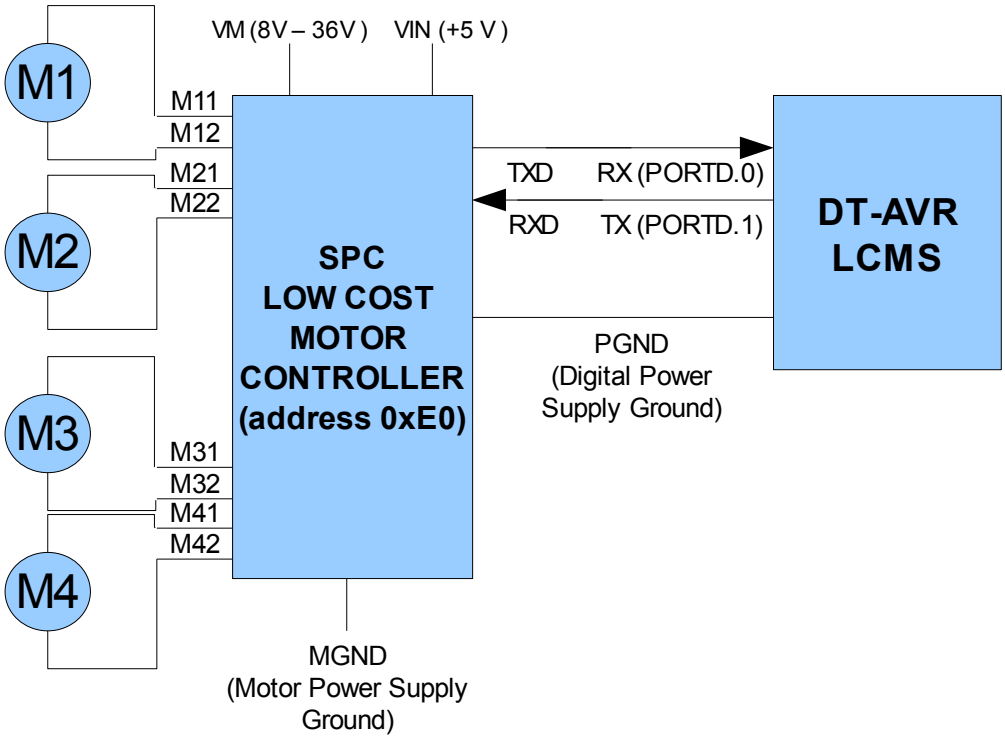
4. TESTING PROCEDURE

1. Connect the 5 Volts power supply to VIN and 9 - 12 Volts to VM SPC LOW COST MOTOR CONTROLLER module.
2. Indicator LED will blink according to I^2C address.
3. Send "DC Forward" command to motor 1 (M1) with PWM value of 255 via UART TTL interface.
4. Indicator LED M1 will blink. When the voltage between pin M11 and M12 is measured, the result will be close to the motor power supply voltage given on the VM pin.
5. Repeat step 3 and 4 for motor 2 (M2), motor 3 (M3), and motor 4 (M4).

5. APPLICATION AND PROGRAM EXAMPLE

As an application example, SPC LOW COST MOTOR CONTROLLER is used to run 4 DC motors with I^2C or UART interface. DT-AVR Low Cost Micro System (LCMS) module with ATmega8535 microcontroller is used as master. The following are the connections between the modules:





As an example program for the above application, there are two programs named contoh_i2c.c and contoh_uart.c (included in the CD/DVD) written using CodeVisionAVR 1.25.2 evaluation.

In the program, DT-AVR LCMS will send "DC Forward" command for each motors with PWM value of 255 to SPC module (for example, SPC's I^2C address is 0xE0) with about 1000ms delay for each command. After all of the commands are sent, DT-AVR LCMS will wait for 3000 ms. Afterward "DC All Stop" command will be sent to SPC followed by another 3000 ms delay. Then DT-AVR LCMS will send "DC Reverse" command for each motor with PWM value of 128 to SPC module with a 1000 ms delay for each command. When all "DC Reverse" commands have been sent, DT-AVR LCMS will wait for 3000 ms. The program ends with DT-AVR LCMS sending "DC Stop" command for each motor to SPC module.

ATTACHMENT A.
SPC LOW COST MOTOR CONTROLLER Schematics