AVR1607: Brushless DC Motor (BLDC) Control in Sensor Mode using ATxmega128A1 and ATAVRMC323

Features

• BLDC motor basics
• Hardware implementation
• Code example

Description

This application note describes how to implement a control of a brushless DC motor (BLDC) in sensor mode using the Atmel® AVR® XMEGA® ATxmega128A1 Microcontroller and the ATAVRMC323 development kit.

This application note deals only with BLDC motor control application using Hall Effect position sensors to control the commutation sequence.

In this document, we will give a short description of brushless DC motor theory of operations. We will detail how to control a brushless DC motor in sensor mode, and we will also provide a short description of the ATAVRMC303 and ATAVRMC300 boards used in this application note.

Software implementation is also discussed with a software control loop using a PID filter.
# Table of Contents

1. References and Glossary ................................................................. 3  
   1.1 References ............................................................................. 3  
   1.2 Glossary .............................................................................. 3

2. Brushless DC Motor Theory of Operation ..................................... 4  
   2.1 Simplified Model of a BLDC Motor ......................................... 4  
   2.2 Six-step Commutation .............................................................. 4  
   2.3 Power Stage ........................................................................... 5  
   2.4 Hall Sensors ............................................................................. 6  
   2.5 Phase Commutations ................................................................. 6

3. ATxmega128A1 Microcontroller ..................................................... 7  
   3.1 Timer/Counters ........................................................................ 7  
   3.2 Analog Features ......................................................................... 8

4. Hardware Description ....................................................................... 8  
   4.1 Hardware Implementation .......................................................... 9  
   4.2 MC300 Configuration and Use .................................................. 10  
   4.3 MC303 Configuration and Use .................................................. 11  
   4.4 Power Supply ........................................................................... 11  
   4.5 Motor ....................................................................................... 11  
   4.6 Technical Advice ....................................................................... 11  
      4.6.1 Disconnecting the BLDC Motor ............................................ 11  
      4.6.2 Ground and PowerWirings .................................................. 12

5. Firmware .......................................................................................... 12  
   5.1 Main Flowchart ........................................................................ 12  
   5.2 INIT: Initialization Functions ...................................................... 13  
      5.2.1 clock_init() ........................................................................ 13  
      5.2.2 mc_init() ............................................................................ 13  
   5.3 Regulation Functions ................................................................. 13  
      5.3.1 mci_set_motor_speed() ....................................................... 13  
      5.3.2 mc_regulation_loop() ......................................................... 13  
   5.4 Commutation ............................................................................. 14  
      5.4.1 mc_switch_commutation() ................................................ 14

6. RS-232 Communication with Firmware ........................................... 15  
   6.1 Connecting ATAVRMC303 to use the RS-232 Interface ............ 15  
   6.2 PC Applications ........................................................................ 15  
      6.2.1 PC Terminal: RS-232 Messages and Commands ................. 15  
      6.2.2 Atmel Motor Control Center .............................................. 15

7. USB Communication ........................................................................ 16

8. Revision History ............................................................................. 17
1. References and Glossary

1.1 References

[1] ATxmega128A1 Datasheet

1.2 Glossary

[1] BLDC: Brushless DC
[3] IGBT: Insulated Gate Bipolar Transistor
2. Brushless DC Motor Theory of Operation

2.1 Simplified Model of a BLDC Motor

A simplified model of a three-phase BLDC motor stator consists of three coils arranged in three directions, U, V, and W (Figure 2-1).

A permanent magnet forms the rotor. The rotor in a BLDC motor consists of an even number of permanent magnets. The number of magnetic poles in the rotor also affects the step size and torque ripple of the motor. More poles provide smaller steps and less torque ripple. The permanent magnets go from one to five pairs of poles. In certain cases it can go up to eight pairs of poles.

Here the rotor is outlined as a bar magnet with its rotary axis at the intersection of the three axes, U, V, and W, perpendicular to the plane of these axis. The orientation/position of the permanent magnet can be controlled by driving a configuration of currents through the three coils. The bar magnet comes to position sector 1 when a current is driven from W through V, and it comes to the following orientation (sector 2) when a current is driven from W to U.

Figure 2-1. Simplified Model of a BLDC Motor

2.2 Six-step Commutation

The method for energizing the motor windings in the sensor method described in this application note is the six-step commutation. Each step, or sector, is equivalent to 60 electrical degrees. Six sectors make up 360°, or one electrical revolution.

The arrows in the winding diagram (Figure 2-2) show the direction current flows through the motor windings in each of the six sectors.

The graph (Figure 2-3) shows the voltage applied at each lead of the motor during the six sectors. Sequencing through these six sectors moves the motor one electrical revolution.

For every sector, two windings are energized and the third winding is floating (not energized).

Connecting the coils to the power and neutral bus induces the current flow. This is referred to as trapezoidal commutation or block commutation.
2.3 Power Stage

The commutation pattern is controlled with a three-phase bridge (see Figure 2-4). The three half bridges have six power switches (IGBT or MOSFET transistors) which are switched according to the defined commutation pattern.

Notes: 1. Open phase is the one shown with dotted line.
2. Hall states are detailed in Section 2.4.
2.4 Hall Sensors

For the estimation of the rotor position, the motor is equipped with three Hall sensors. These Hall sensors are placed every 120°. With these sensors, six different commutations are possible. Phase commutation will depend on these Hall sensor values.

Figure 2-5 shows the three Hall sensors signals, H1, H2, and H3, as motor turns using sensor control. Hall states are the combination result of the H1, H2, and H3 signals.

Figure 2-5. Hall States versus Motor Leads in Sensor Mode

![Hall States versus Motor Leads in Sensor Mode](image)

With a motor of n pairs of poles, the hall sensors frequency is n times faster than the motor rotation.

2.5 Phase Commutations

Power supply to the coils must be changed when Hall sensor values change. With right synchronized commutations, the torque remains nearly constant and high.

Reading Hall sensors values (HS_xxx variable) indicate which new scheme should be switched (see Table 2-1).

Table 2-1. Switch Commutation for Clockwise Rotation when Viewed from Non-driving End

<table>
<thead>
<tr>
<th>Hall Sensor Value (H1 H2 H3) = HS_xxx</th>
<th>Hall States</th>
<th>Previous Phases</th>
<th>Previous Scheme</th>
<th>Next Phases</th>
<th>Next Scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>3</td>
<td>V-W</td>
<td>T3 ; T6</td>
<td>U-W</td>
<td>T1 ; T6</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>U-W</td>
<td>T1 ; T6</td>
<td>U-V</td>
<td>T1 ; T4</td>
</tr>
<tr>
<td>101</td>
<td>5</td>
<td>U-V</td>
<td>T1 ; T4</td>
<td>W-V</td>
<td>T5 ; T4</td>
</tr>
<tr>
<td>001</td>
<td>4</td>
<td>W-V</td>
<td>T5 ; T4</td>
<td>W-U</td>
<td>T5 ; T2</td>
</tr>
<tr>
<td>011</td>
<td>6</td>
<td>W-U</td>
<td>T5 ; T2</td>
<td>V-U</td>
<td>T3 ; T2</td>
</tr>
<tr>
<td>010</td>
<td>2</td>
<td>V-U</td>
<td>T3 ; T2</td>
<td>V-W</td>
<td>T3 ; T6</td>
</tr>
</tbody>
</table>

For motors with multiple poles, the electrical rotation does not correspond to a mechanical rotation. A BLDC motor with n pair of poles uses n electrical rotation cycles for one mechanical rotation.
The strength of the magnetic field determines the force and speed of the motor. By varying the current flow through the coils, the speed and torque of the motor can be adjusted. The most common way to control the current flow is to control the average current flow through the coils. PWM (pulse width modulation) is used to adjust the average voltage and, thereby, the average current, inducing the speed. For example, the PWM frequency selected is the range from 10kHz to 200kHz according to the application (commutation losses, audible frequency...).

**Figure 2-6. PWM Scheme**

Commutation creates a rotating field.

For instance at step 5, phase U is connected to the positive DC bus voltage through T1, phase V is connected to ground through T4, and phase W is unpowered. Two flux vectors are generated by phase U and phase V. The sum of the two vectors creates the stator flux vector. Then the rotor tries to follow this stator flux.

As soon as the rotor reaches the given position, the Hall sensor’s state changes its value from “101” to “001,” and a new voltage pattern is selected and applied to the BLDC motor. Then phase U is unpowered and phase W is connected to the positive DC bus, resulting in a new stator flux vector, step 4.

Following the commutation schematic (Figure 2-5) and Table 2-1, we get six different stator flux vectors corresponding to the six commutation steps. These six steps provide one electrical revolution.

### 3. ATxmega128A1 Microcontroller

Based on the high-performance AVR 8-bit RISC architecture, the ATxmega128A1 integrates all of the basic peripherals necessary to satisfy the needs of complex algorithms.

The ATxmega128A1 has all necessary resources to provide an integrated solution to control BLDC motors in their system environments.

#### 3.1 Timer/Counters

ATxmega128A1 provides 16-bit timers/counters with:

- Four compare or capture (CC) channels in Timer/Counter 0.
- Two compare or capture (CC) channels in Timer/Counter 1.

To generate PWM frequency, this application note uses three compare channels (A, B, C) of Timer 0 and the advanced waveform extension (AWEX) feature of Timer 0. The AWEX function is available for ports C and E (port C is used in this application note).

The benefits of the AWEX feature for motor control are:
Complementary outputs from each capture channel
- Four dead-time insertions, which avoid cross conduction
- Separate high and low side dead-time setting
- Double-buffered dead time
- Event controlled fault protection
- Single-channel, multiple-output operation
- Double-buffered pattern generation

The output pairs go through a dead-time insertion (DTI) unit that enables generation of the non-inverted low side (LS) and inverted high side (HS) of the WG output with dead-time insertion between LS and HS switching. The DTI output will override the normal port value according to the port override setting.

The fault protection unit is connected to the event system. This enables any event to trigger a fault condition that will disable the AWEX output. Several event channels can be used to trigger a fault on several different conditions.

3.2 Analog Features

ATxmega128A1 integrates several analog blocks useful for motor control:
- Two eight-channel, 12-bit, 2Msps analog-to-digital converters with programmable gain options. The inputs per ADC are:
  - Eight single-ended inputs
  - 8x4 differential inputs with selectable 1/2/4/8/16/32/64x gain
- Four analog comparators with window compare function, with selectable comparison levels, and interrupts on pin change

The ATxmega128A1 includes independent positive and negative comparator inputs available for over current detection. The input selection can be achieved from:
- Pins 0, 1, 2, 3, 4, 5, 6 for positive input
- Pins 0, 1, 3, 5, 7 for negative input

In addition to the pins above, the analog comparator reference (comparison level) can be selected from both internal and external sources:
- Output from a 12-bit DAC
- Voltage scaler that can do a 64-level scaling of the internal Vcc voltage
- Bandgap voltage reference

Refer to the ATxmega128A1 datasheet for a complete description of the ATxmega128A1 microcontroller.

4. Hardware Description

This application has been developed with ATAVRMC300 and ATAVRMC303 boards, which are the two parts of the ATAVRMC323 starter kit.

The ATAVRMC300 board is the power board, which embeds the power bridge. The ATAVRMC300 is connected (see Figure 4-1) to the ATAVRMC303, which is the processor board built around the ATxmega128A1 processor.
4.1 Hardware Implementation

The block diagram of the sensor closed loop is shown in Figure 4-2.

The outputs UH, UL, VH, VL, WH, and WL of AVR303 are used to control the power bridge (see Table 2-1). As previously seen, they depend on Timer 0 and AWEX, which generates PWM signals.

An external comparator on the MC303 board provides a fault signal (Fault_overcurrent) connected to the PE4 input:
The compared inputs are the shunt_pos and AVCC/34. Shunt_pos is the voltage monitored across the 0.05Ω resistor, and equals the motor current divided by 20.

The comparator output will toggle as soon as motor current, Im, will be:

\[
\text{Im} / 20 > \text{AVCC} / 34
\]

So the limit is: \(\text{Im} > 2.94\text{A}\)

**Table 4-1. Microcontroller I/O Ports use (TQFP100 Package)**

<table>
<thead>
<tr>
<th>PORTA</th>
<th>PA0</th>
<th>95</th>
<th>U_cond_neg</th>
<th>U Reference for zero crossing (Not used)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PA1</td>
<td>96</td>
<td>V_cond_neg</td>
<td>V Reference for zero crossing (Not used)</td>
</tr>
<tr>
<td></td>
<td>PA2</td>
<td>97</td>
<td>U_conditioned</td>
<td>= U / 6 (Not used)</td>
</tr>
<tr>
<td></td>
<td>PA3</td>
<td>98</td>
<td>V_conditioned</td>
<td>= V / 6 (Not used)</td>
</tr>
<tr>
<td></td>
<td>PA4</td>
<td>99</td>
<td>W_conditioned</td>
<td>= W / 6 (Not used)</td>
</tr>
<tr>
<td></td>
<td>PA5</td>
<td>100</td>
<td>Vn_conditioned</td>
<td>Neutral voltage / 34 (Not used)</td>
</tr>
<tr>
<td></td>
<td>PA6</td>
<td>1</td>
<td>Vm</td>
<td>(Not used)</td>
</tr>
<tr>
<td></td>
<td>PA7</td>
<td>2</td>
<td>W_cond_neg</td>
<td>W Reference for zero crossing (Not used)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PORTB</th>
<th>PB0</th>
<th>5</th>
<th>Speed_ref</th>
<th>Speed potentiometer (MC303)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PB1</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PB2</td>
<td>7</td>
<td>Current_reference</td>
<td>= AVCC / 34</td>
</tr>
<tr>
<td></td>
<td>PB3</td>
<td>8</td>
<td>Shunt_pos</td>
<td>0.05Ω shunt voltage ( = motor current / 20)</td>
</tr>
<tr>
<td></td>
<td>PB4</td>
<td>9</td>
<td>Shunt_neg</td>
<td>GND of Vmotor</td>
</tr>
<tr>
<td></td>
<td>PB5</td>
<td>10</td>
<td>Shunt_U</td>
<td>= Shunt_pos (not used)</td>
</tr>
<tr>
<td></td>
<td>PB6</td>
<td>11</td>
<td>Shunt_V</td>
<td>= Shunt_pos (not used)</td>
</tr>
<tr>
<td></td>
<td>PB7</td>
<td>12</td>
<td>Shunt_W</td>
<td>= Shunt_pos (not used)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PORTC</th>
<th>PC0</th>
<th>15</th>
<th>UL</th>
<th>Drives T1 power transistor of MC300</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PC1</td>
<td>16</td>
<td>UH</td>
<td>Drives T2 power transistor of MC300</td>
</tr>
<tr>
<td></td>
<td>PC2</td>
<td>17</td>
<td>VL</td>
<td>Drives T3 power transistor of MC300</td>
</tr>
<tr>
<td></td>
<td>PC3</td>
<td>18</td>
<td>VH</td>
<td>Drives T4 power transistor of MC300</td>
</tr>
<tr>
<td></td>
<td>PC4</td>
<td>19</td>
<td>WL</td>
<td>Drives T5 power transistor of MC300</td>
</tr>
<tr>
<td></td>
<td>PC5</td>
<td>20</td>
<td>WH</td>
<td>Drives T6 power transistor of MC300</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PORTE</th>
<th>PE0</th>
<th></th>
<th>H1</th>
<th>Hall sensor 1 signal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PE1</td>
<td></td>
<td>H2</td>
<td>Hall sensor 2 signal</td>
</tr>
<tr>
<td></td>
<td>PE2</td>
<td></td>
<td>H3</td>
<td>Hall sensor 3 signal</td>
</tr>
<tr>
<td></td>
<td>PE4</td>
<td></td>
<td>Fault_overcurrent</td>
<td>External comparator (MC303) output</td>
</tr>
<tr>
<td></td>
<td>PE5</td>
<td></td>
<td>Fault_IPM</td>
<td>(Not used)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PORTH</th>
<th>PH0</th>
<th></th>
<th>ZC_U</th>
<th>Zero crossing external comparator output: sensorless mode (Not used)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PH1</td>
<td></td>
<td>ZC_V</td>
<td>Zero crossing external comparator output: sensorless mode (Not used)</td>
</tr>
<tr>
<td></td>
<td>PH2</td>
<td></td>
<td>ZC_W</td>
<td>Zero crossing external comparator output: sensorless mode (Not used)</td>
</tr>
</tbody>
</table>

### 4.2 MC300 Configuration and Use

The power board must be supplied with a 12V, 2A, DC power supply.
Table 4-2. ATAVRMC300 jumper settings

<table>
<thead>
<tr>
<th>Jumper</th>
<th>Position</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>J1(VHa)</td>
<td>Pin 1 and 2 shorted</td>
<td>VHa = +5V</td>
</tr>
<tr>
<td>J2(VCC)</td>
<td>connected</td>
<td>VCC = +3.3V</td>
</tr>
</tbody>
</table>

4.3 MC303 Configuration and Use

The jumpers’ configurations of the MC303 processor board are:

Table 4-3. ATAVRMC303 jumper settings

<table>
<thead>
<tr>
<th>Jumper</th>
<th>Position</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>J15</td>
<td>Open</td>
<td>Shunt_neg (Jumpers to be removed to enable JTAG programming)</td>
</tr>
<tr>
<td>J16</td>
<td>Open</td>
<td>Shunt_U (Jumpers to be removed to enable JTAG programming)</td>
</tr>
<tr>
<td>J17</td>
<td>Open</td>
<td>Shunt_V (Jumpers to be removed to enable JTAG programming)</td>
</tr>
<tr>
<td>J18</td>
<td>Open</td>
<td>Shunt_W (Jumpers to be removed to enable JTAG programming)</td>
</tr>
</tbody>
</table>

4.4 Power Supply

This firmware example has been configured according to a power supply Vm = 12V.

This power-supply must be able to provide up to 4A output current.

4.5 Motor

The BLDC motor provided inside the MC323 and MC300 Motor Control Kit has the following characteristics:

Manufacturer: TECMOTION
Number of phases: 3
Number of poles: Eight (four pairs)
Rated voltage: 24V
Rated speed: 4000rpm
Rated torque: 62.5mNm
Torque constant: 35mNm/A = k_tau
Line-to-line resistance: 1.8Ω = R
Back EMF: 3.66V/Krpm = k_e
Peak current: 5.4A
Maximum speed: 6900rpm

4.6 Technical Advice

4.6.1 Disconnecting the BLDC Motor

The BLDC motor must not be disconnected while it is running or while its coils carry current. One should only disconnect a BLDC motor when the PWM duty cycle is 0% and the rotor is at rest so that no current is driven through the coils. Be careful when stopping the power supply or PWM: a BLDC motor with a high moment of inertia is able to run for a relatively long time.
4.6.2 Ground and PowerWirings
When designing a board for BLDC motor control one has to take care of the ground wiring and power wiring. The power supply of the processor and additional signal conditioning components (e.g. additional fast comparators, operational amplifiers) has to be decoupled from the motor power supply. The ground connection has to be of low resistance and low inductance to prevent against voltage drop and noise due to high currents. A ground plane within a multilayer PCB is recommended for proper operation.

5. Firmware
The source file directory embeds HTML documentation that can be opened through the readme.html file.
The application to ATxmega128A1 is detailed in the following sections.

5.1 Main Flowchart
The firmware main flowchart is described in Figure 5-1.

Figure 5-1. Main Flowchart

The tasks are scheduled thanks to the g_tick produced each 250µs with Timer 1.
5.2 INIT: Initialization Functions
The initialization functions are described in the following sections.

5.2.1 clock_init()
- Oscillator: 32MHz RC oscillator and OSC oscillator
- PLL: clock source = 32MHz and factor = 16
- Prescalers: PSADIV = 1 and PSB and PSCDIV = 2:
  means \(\text{clk}_{\text{PER4}} = 128\text{MHz}, \text{clk}_{\text{PER2}} = 64\text{MHz}, \text{and} \text{clk}_{\text{PER}} = 32\text{MHz}\)

5.2.2 mc_init()
- Port C: Pins 0 to 5 are in output mode and clear (DIRSET and OUTCLR = 1), and are the outputs connected to the transistor power bridge
- Port E: The input pull ups are activated to connect the Hall sensor signals
- External interrupts are defined on Port E pins 0/1/2 (Hall signals)
- Timer 0 configuration is:
  - Clock prescaler / 4 (8MHz)
  - PWM_Init (255) configures a PWM frequency = 15686kHz
- Timer 1 configuration is:
  - No clock prescaler and period = 8000 produces a g_tick each 250\(\mu\)s
- ADCB configuration is:
  - Calibration/offset
  - Signed conversion mode and 12-bit resolution
  - ADC prescaler to a sample rate of CPUFREQ / 16. Allow time for storing data
  - Set reference voltage to \(V_{CC} - 0.6\text{V}\)
  - Set channel 0 to have single-ended input and gain = 1
  - Set input to the channels in ADCB to be pin 0
  - Enable ADCB with free-running mode

5.3 Regulation Functions
The main loop functions are described in the following sections.

5.3.1 mci_set_motor_speed ()
This function updates the speed set point according to the potentiometer adjustment or the speed command received on serial transmission.

\(mc\_get\_potentiometer\_value()\) returns mc_potentiometer_value

5.3.2 mc_regulation_loop() :
The duty_cycle variable controls the PWM generator. This variable is the result of the following functions:

in Open loop mode: \(duty\_cycle = mc\_get\_motor\_speed()\)
in Speed loop mode: \(duty\_cycle = mc\_control\_speed(mc\_get\_motor\_speed())\)
in Current loop mode: \(duty\_cycle = mc\_control\_current(mc\_get\_potentiometer\_value())\)
5.4 Commutation

The phase commutation has to be achieved according to the three Hall sensors. Commutation between steps is achieved when a rising or falling edge occurs on one of the three Hall sensor signals. Hall sensor signals H1/H2/H3 are connected to PE0/PE1/PE2, which are configured as interrupt sources. (Another solution could be using internal comparators to generate interrupts.)

Port Interrupt 0 mask is defined as:

```c
sfrb PORTE_INT0MASK = 0x068A
```

Are also defined:

```c
#define PORTE_INT0MASK PORTE.INT0MASK
PORTE.INT0MASK = (PIN0_bm | PIN1_bm | PIN2_bm);
```

This means that PORTE_INT0_vect (HALL-A()) is executed if any of the three Hall sensor signals are changing. This interrupt vector executes:

- the transistor commutation according to the HALL_SENSOR_VALUE:
  - HALL_SENSOR_VALUE is the value of the three Hall sensor bits = PORTE.IN & 0x07
  - The commutation is achieved by the function: `mc_switch_commutation(HALL_SENSOR_VALUE)`
- estimation of speed on the rising edge of H1 (Hall A) sensor. This means the speed is evaluated one time per electrical cycle.

5.4.1 `mc_switch_commutation()`

This function achieves two operations:

1. First, the update of the duty cycle, thanks to function:
   ```c
   mc_duty_cycle(mc_get_Duty_Cycle())
   ```
   This is achieved with the update of the output new compare value of the timer:
   ```c
   TC_SetCompareA( &TCC0, level );
   TC_SetCompareB( &TCC0, level );
   TC_SetCompareC( &TCC0, level );
   ```
   The dead time insertion is achieved with the `ConfigDTI()` function, which configures a dead time for both sides equal to 3.

2. Secondly, a switch of the power bridge according to the next expected position and the direction of the rotation. The AWEX feature, described previously, is configured on Port C.

Ports C0 to C5 are the UH/UL/VH/VL/WH/WL outputs.

An example of Hall detection and the resulting actions is described below:

```c
case HS_001:  if (direction==CCW)  {Set_Q1Q6();}
else                      {Set_Q5Q2();}
```

According to Table 2-1, the scheme following the Hall state 001 is T5 and T2 transistors on (in clockwise direction)

`Set_Q5Q2()` instruction executes the AWEX operation:

```c
AWEX_SetOutoutOveride(AWEXC,PIN5_bm | PIN0_bm)
```

which drives pins 0 and 5 connected to gates of T5 and T2 transistors.
6. RS-232 Communication with Firmware

6.1 Connecting ATAVRMC303 to use the RS-232 Interface
Connect the PC COM port to the ATAVRMC303 RS-232 connector through a direct cable.

- The serial configuration is:
  - 38400 baud
  - Eight data bits
  - One stop bit
  - No handshake

6.2 PC Applications
The user can communicate with the firmware through an RS-232 connection with the usual PC serial communication applications (i.e. HyperTerminal) or the Atmel Motor Control Center application, which can be downloaded from Atmel the web site at http://www.atmel.com.

6.2.1 PC Terminal: RS-232 Messages and Commands
At power up, the following welcome message is received on the terminal: “ATMEL Motor Control Interface”

The following commands can be sent to the firmware:

<table>
<thead>
<tr>
<th>Command</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>ru</td>
<td>Run motor</td>
</tr>
<tr>
<td>st</td>
<td>Stop Motor</td>
</tr>
<tr>
<td>help</td>
<td>Gives help</td>
</tr>
<tr>
<td>fw</td>
<td>Set direction to Forward</td>
</tr>
<tr>
<td>bw</td>
<td>Set direction to Backward</td>
</tr>
<tr>
<td>ss</td>
<td>Set Speed (followed with speed value)</td>
</tr>
<tr>
<td>gi</td>
<td>Get ID</td>
</tr>
<tr>
<td>g0</td>
<td>Get Status 0</td>
</tr>
<tr>
<td>g1</td>
<td>Get Status 1</td>
</tr>
</tbody>
</table>

6.2.2 Atmel Motor Control Center
The Atmel Motor Control Center User Guide is available in the install directory located at C:\Program Files\Atmel\Motor Control Center\help\Overview.htm

First, communication must be achieved with the MC303: Select Settings -> Target communication -> RS-232.
Then force the COM port number to the one which has been enumerated as cdc (i.e. COM9 in Figure 6-1).
The AVR1607 Target must then be selected to get the right configuration: to select this target, execute the File > Select Target command or click the button in the toolbar. The dialog box shown in Figure 6-1 pops up.
7. **USB Communication**

Communication can be achieved from the PC to the USB connector of the MC303 board. The **AVR1014, MC303 Hardware User Guide** details the configuration to be achieved. The communication port becomes a virtual COM port, which must be forced in the RS-232 Settings window. The same tools and commands described in Chapter 6 can be used through this virtual COM port.
## 8. Revision History

<table>
<thead>
<tr>
<th>Doc. Rev.</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>8311C</td>
<td>03/2014</td>
<td>Units correction</td>
</tr>
<tr>
<td>8311B</td>
<td>08/2013</td>
<td>Errors correction and new document template</td>
</tr>
<tr>
<td>8311A</td>
<td>07/2010</td>
<td>Initial document release</td>
</tr>
</tbody>
</table>