PROJECT 16

DRIVING TWO DC GEARED MOTOR WITH MD10C BY USING SKPS

Enhanced 10Amp DC Motor Driver (MD10C) is an enhanced version of the MD10B which is designed to drive high current brushed DC motor up to 10A continuously. It offers several enhancements over the MD10B such as support for both locked-antiphase and sign-magnitude PWM signal as well as using full solid state components which result in faster response time and eliminate the wear and tear of the mechanical relay. Some example of application is used to drive motor for robot arms or smaller mobile robot part. In addition, by further interface with **PS2 Controller Starter Kit (SKPS),** we can control the motors with PS2 controller through UART (*Universal Asynchronous Receiver/Transmitter)* which will establish a serial communication between PIC and SKPS.

In this project, we will learn about how to drive two DC geared motor with MD10C by using SKPS. Thus, we will treat these two motors as the left wheel motor and right wheel motor. Therefore, we will use PS2 controller to control two DC motors in various type of movements such as move forward, move backward, rotate to left, rotate to right. In addition, we will use joystick to control these two motors in different movement with constant speed and variable speed. Meanwhile, we will use PIC16F887 for SK40C with LCD 2x16 to display some important messages when certain button of PS2 controller was pressed. In addition, the sample code that used for PIC16F877A will also be attached at the end of this tutorial.

COMPONENT NEEDED

	1x ENHANCED 40 PINS PIC START-UP KIT (SK40C)
	1x LCD DISPLAY (2x16) – BLUE BACKLIGHT (DS-LCD-162A-B)
	2x ENHANCED 10A DC MOTOR DRIVER (MD10C)
	1x PS2 CONTROLLER STARTER KIT (SKPS)

	2x DC GEARED MOTOR (SPG30-300K)
	1x PS2 CONTROLLER (PS-GP-1)
	MALE-TO-MALE JUMPER WIRES (WR-JW-MM65)
	FEMALE-TO-FEMALE JUMPER WIRES (WR-JW-FF10)
	1x LIPO RECHARGEABLE BATTERY (LIP-11.1-2200)
	1x ADAPTER 12V 2A (AD-12-2)

	1x SCREW DRIVER SET (TO-SD-803)

CONNECTION

For the latest MD10C, it is compatible with both Sign-Magnitude PWM and Locked-Antiphase PWM mode. Meanwhile, the only differences between these two modes are its connection. For Sign-Magnitude PWM, we need to connect its PWM pin to PWM output port of PIC and then connect its DIR pin to digital output port of PIC. However, for Locked-Antiphase PWM mode, we need to set its DIR pin as logic HIGH and then connect its PWM pin to PWM output port of PIC.

Therefore, in the following, I will briefly explain the connection for these two modes in separate part:

MODE 1: SIGN-MAGNITUDE PWM

To operate in Sign-Magnitude PWM mode, we need to use two different control signals to control the speed and direction of the motor. Thus, in this mode, PWM pin is used to control the speed while DIR pin is used to control the direction of the DC motor. Meanwhile, to further interface with SKPS, we need to connect SKPS to SK40C for serial communication. For this, we need cross connect the RX and TX pin between SKPS and SK40C.

Please refer to the connection diagram below:

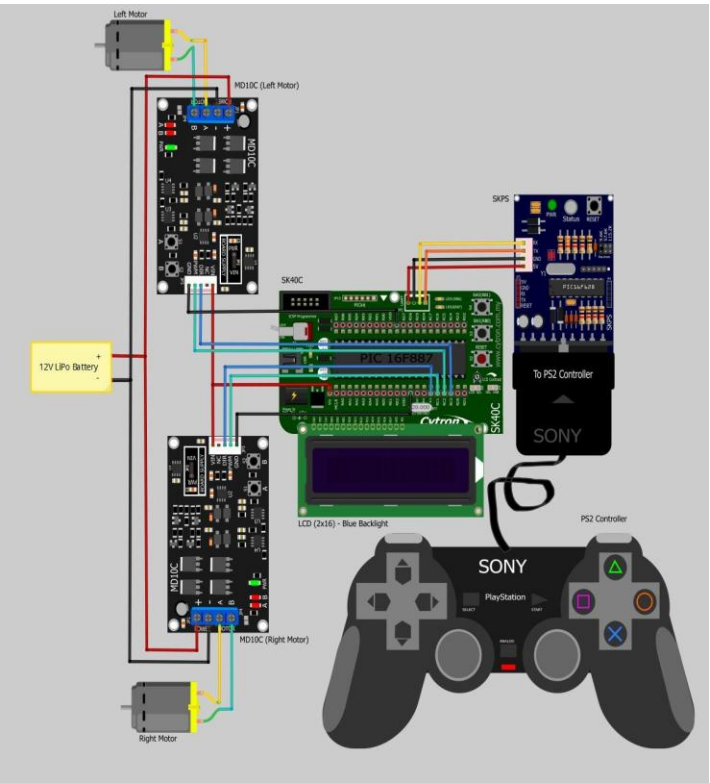

Figure 1: Full connection diagram in Fritzing for Sign-magnitude PWM mode

The following tables show the summary of the connection for sign-magnitude mode:

MD10C Terminal pins	**Connection Descriptions**
POWER (+)	Connected to positive terminal of the 12V battery
POWER (-)	Connected to negative terminal of the 12V battery
MOTOR Output A	Connected to motor terminal A
MOTOR Output B	Connected to motor terminal B

Table 1: Connection for MD10C terminal pins

*Note: Please ensure that the polarity of the 12V battery is correctly connected to MD10C to prevent permanent damage.

SK40C (PIC16F887)	**MD10C (Left motor)**	**MD10C (Right motor)**	**SKPS**
VDD (+5V)	-	-	5V
GND	GND	GND	GND
Vin (12V supply from power adapter)	VIN	VIN	-
RC0	-	DIR	-
RC1	-	PWM	-
RC2	PWM	-	-
RC3	DIR	-	-
RC6 (TX)	-	-	RX
RC7 (RX)	-	-	TX

Table 2: Connection between SK40C-MD10C-SKPS

*Note: The field that mark with '-' is means that no connection needed.

MODE 2: LOCKED-ANTIPHASE PWM

To operate in Locked-Antiphase PWM mode, we need only single control signal to control both the speed and direction of the motor. Thus, in this mode, PWM pin is always connected to logic high while the DIR pin is connected to the PWM signal. Therefore, when the PWM signal has 50% duty cycle, the motor will stop. Meanwhile, if the PWM has less than 50% duty cycle, the motor will turn in one direction and turn in another direction if the PWM signal has more than 50% duty cycle. Furthermore, to further interface with SKPS, we need also connect SKPS to SK40C for serial communication as for Sign-Magnitude mode.

Please refer to the connection diagram below:

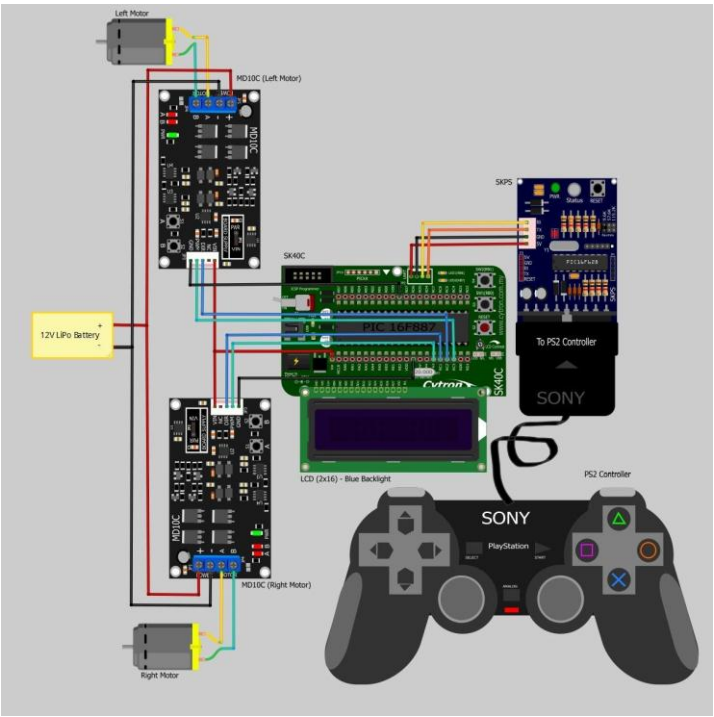

Figure 2: Full connection diagram in Fritzing for Locked-Antiphase PWM mode

The following tables show the summary of the connection for locked-antiphase mode:

MD10C Terminal pins	**Connection Descriptions**
POWER (+)	Connected to positive terminal of the 12V battery
POWER (-)	Connected to negative terminal of the 12V battery
MOTOR Output A	Connected to motor terminal A
MOTOR Output B	Connected to motor terminal B

Table 3: Connection for MD10C terminal pins

*Note: Please ensure that the polarity of the 12V battery is correctly connected to MD10C to prevent permanent damage.

SK40C (PIC16F887)	**MD10C (Left motor)**	**MD10C (Right motor)**	**SKPS**
VDD (+5V)	-	-	5V
GND	GND	GND	GND
Vin (12V supply from power adapter)	VIN	VIN	-
RC0	-	PWM	-
RC1	-	DIR	-
RC2	DIR	-	-
RC3	PWM	-	-
RC6 (TX)	-	-	RX
RC7 (RX)	-	-	TX

Table 4: Connection between SK40C-MD10C-SKPS

*Notes: The field that mark with '-' is means that no connection needed.

ADDITIONAL INFORMATION

Some additional information about MD10C have been discussed inside Project 10: Driving DC brush motor with MD10C. Please refer it from there. For further information, please refer to MD10C User Manual.

Apart from that, some information about SKPS has been discussed inside Project 15: Interface with SKPS. Please refer it from there. For further information, please refer to SKPS User Manual.

CODE OVERVIEW

MODE 1: SIGN-MAGNITUDE PWM

For Sign-Magnitude PWM, the basic ideas of programming are to control the two signals to control the speed and direction. Referring to the connection description above, we connect PWM of MD10C to RC1 for right motor and RC2 for left motor so that we can control the speed by setting the duty cycle of the PWM because RC1 and RC2 are the ports that can function as PWM. Meanwhile, to control the direction of motor, we need to connect the DIR of MD10C to digital ports such as RC0 and RC3. Then, we control the direction by set it logic HIGH or LOW for different direction. The following shows some descriptions of the sample code for PIC16F887:

1. Firstly, we need to include the important library and write the configuration words for PIC16F887. In the SK40C tutorial, it uses HI-TECH C Compiler V9.70, so I just write the configuration words for this version in the sample code as shown below:

```
__CONFIG(HS &           // High Speed Crystal.
         WDTDIS &       // Disable Watchdog Timer.
         PWRTEN &       // Enable Power Up Timer.
         BORDIS &       // Disable Brown Out Reset.
         MCLREN &       // MCLR function is enabled
         LVPDIS);       // Disable Low Voltage Programming.
```

2. Secondly, we need to define the ports name and also the SKPS protocol in the "system.h" file as shown below:

(i) Definition of oscillator frequency, UART, LCD and SK40C elements.

```
// Oscillator Frequency.
#define _XTAL_FREQ      20000000

// UART baud rate
#define UART_BAUD   9600

// LCD pins
#define LCD_RS      RB4
#define LCD_EN      RB5
#define LCD_data    PORTD

//  SK40C switches
#define SW1         RB0
#define SW2         RB1

//  MD10C ports
#define MD10C_R_DIR RC0
#define SPEEDR      CCPR2L
#define SPEEDL      CCPR1L
#define MD10C_L_DIR RC3

//  SK40C LED
#define LED1        RB6
#define LED2        RB7
```

(ii) Definition of SKPS protocols

```
//  SKPS protocol
#define p_select        0
#define p_joyl          1
#define p_joyr          2
#define p_start         3
#define p_up            4
#define p_right         5
#define p_down          6
#define p_left          7
#define p_l2            8
#define p_r2            9
#define p_l1            10
#define p_r1            11
#define p_triangle      12
#define p_circle        13
#define p_cross         14
#define p_square        15
#define p_joy_lx        16
#define p_joy_ly        17
#define p_joy_rx        18
#define p_joy_ry        19
#define p_joy_lu        20
#define p_joy_ld        21
#define p_joy_ll        22
#define p_joy_lr        23
#define p_joy_ru        24
#define p_joy_rd        25
#define p_joy_rl        26
#define p_joy_rr        27
#define p_con_status    28
#define p_motor1        29
#define p_motor2        30
```

3. Thirdly, before we write the main program, we write first the important library file for LCD, UART, PWM and SKPS. Therefore, we just need to include it inside the main program so that we can easily call for it once we want to use it in the main program.

*Note: Please download the completed sample code from the attachment on this project webpage to view these codes.

4. After that, we are ready to write the main program. In the sample code, it will start the program by initialize the microcontroller. Here is the sample code:

```
unsigned char up_v, down_v, left_v, right_v;

//set I/O input output
TRISB = 0b00000011;
TRISC = 0b10000000;
TRISD = 0b00000000;

ANSEL = 0;
ANSELH = 0;

LCD_init();
PWM_init();
UART_init();

LED1=0;
LED2=0;
```

5. After that, we start to write main function to start the program with displaying the start-up message and then ask the user to press “START” to continue. Here is the sample code:

```
//display startup message
LCD_clr_home();
LCD_goto(2);
LCD_putstring("MD10C Demo");
LCD_goto(20);
LCD_putstring("SK40C PR16");
delay(100000);
LCD_clr_home();
LCD_putstring("Press Start");
LCD_goto(20);
LCD_putstring("To begin.");
delay(50000);
while(SKPS(p_start)==1) continue;
LCD_clr_home();
```

6. Next, we write the program to run the motor in different type of movement. In the sample code, it has two parts which are control with constant speed and variable speed.

(a) Here is the sample code to drive two motors in constant speed with 4 types of movement (forward, backward, rotate to left, rotate to right) depend on motor terminal connection:

```
if(SKPS(p_up)==0)
{
    LCD_goto(20);
    LCD_putstring("Move Forward!    ");
    MD10C_L_DIR=0;
    MD10C_R_DIR=0;
    SPEEDL=220;
    SPEEDR=220;
}
else if(SKPS(p_down)==0)
{
    LCD_goto(20);
    LCD_putstring("Move Backward!   ");
    MD10C_L_DIR=1;
    MD10C_R_DIR=1;
    SPEEDL=220;
    SPEEDR=220;
}
else if(SKPS(p_left)==0)
{
    LCD_goto(20);
    LCD_putstring("Rotate to left! ");
    MD10C_L_DIR=1;
    MD10C_R_DIR=0;
    SPEEDL=220;
    SPEEDR=220;
}
else if(SKPS(p_right)==0)
{
    LCD_goto(20);
    LCD_putstring("Rotate to right!");
    MD10C_L_DIR=0;
    MD10C_R_DIR=1;
    SPEEDL=220;
    SPEEDR=220;
}
```

(b) Here is the sample code to drive two motors in variable speed with 4 types of movement in Sign-Magnitude mode:

i. Move forward (depend on motor terminal connection) by joystick

```
else if(up_v>0)
{
    LCD_goto(20);
    LCD_putstring("Move Forward!    ");
    MD10C_L_DIR=0;
    MD10C_R_DIR=0;
    if(left_v>0)
    {
        if(up_v>left_v)SPEEDL=up_v-left_v+140;
        else SPEEDL=140;
        SPEEDR=up_v+140;
    }
    else if(right_v>0)
    {
        if(up_v>right_v)SPEEDR=up_v-right_v+140;
        else SPEEDR=140;
        SPEEDL=up_v+140;
    }
    else
    {
        SPEEDL=up_v+140;
        SPEEDR=up_v+140;
    }
}
```

ii. Move backward (depend on motor terminal connection) by joystick

```
else if(down_v>0)
{
    LCD_goto(20);
    LCD_putstring("Move Backward!   ");
    MD10C_L_DIR=1;
    MD10C_R_DIR=1;
    if(left_v>0)
    {
        if(down_v>left_v)SPEEDR=down_v-left_v+140;
        else SPEEDR=140;
        SPEEDL=down_v+140;
    }
    else if(right_v>0)
    {
        if(down_v>right_v)SPEEDL=down_v-right_v+140;
        else SPEEDL=140;
        SPEEDR=down_v+140;
    }
    else
    {
        SPEEDL=down_v+140;
        SPEEDR=down_v+140;
    }
}
```

iii. Rotate to left (depend on motor terminal connection) by joystick

```
else if(left_v>0)
{
    LCD_goto(20);
    LCD_putstring("Rotate to left! ");
    MD10C_L_DIR=1;
    MD10C_R_DIR=0;
    SPEEDL=left_v+120;
    SPEEDR=left_v+120;
}
```

iv. Rotate to right (depend on motor terminal connection) by joystick

```
else if(right_v>0)
{
    LCD_goto(20);
    LCD_putstring("Rotate to left! ");
    MD10C_L_DIR=0;
    MD10C_R_DIR=1;
    SPEEDL=right_v+120;
    SPEEDR=right_v+120;
}
```

(c) Here is the sample code to stop the motor:

```
else
{
    LCD_goto(0);
    LCD_putstring("Motor State:    ");
    LCD_goto(20);
    LCD_putstring("Stop!           ");
    SPEEDL=0;
    SPEEDR=0;
}
```

MODE 2: LOCKED-ANTIPHASE PWM

For Locked-Antiphase PWM, the basic ideas of programming is to control only single signal to control both the speed and direction. Referring to the connection description above, we connect DIR of MD10C to RC1 for right motor and RC2 for left motor so that we can control both the speed and direction by changing the duty cycle of PWM. Thus, the motor will stop at 50% duty cycle, run in one direction if less than 50% duty cycle and run in another direction if more than 50% duty cycle. Meanwhile, the PWM of both MD10C is always set as logic HIGH by programming. The sample code is similar to Sign-Magnitude mode, but only with some modification on pin name definition and movement algorithm as shown in the following descriptions:

1. Firstly, we follow Step 1 as for Sign-Magnitude mode to write the device configuration words.

2. Secondly, we need to modify the code of the definition of pin name from Sign-Magnitude PWM in certain segment as shown below. After that, we just follow the remaining code as for Sign-Magnitude mode.

```
//  MD10C ports
#define MD10C_R_DIR CCPR2L
#define SPEEDR      RC0
#define SPEEDL      RC3
#define MD10C_L_DIR CCPR1L
```

3. Then, we continue to write the same code by following the Step 3 to Step 5 as for Sign-Magnitude mode.

4. Next, we will write the program to run the motor in different movement in this mode. In the sample code, it also has two parts which are control with constant speed and variable speed.

(a) Here is the sample code to drive two motors in constant speed with 4 types of movement (forward, backward, rotate to left, rotate to right) depend on motor terminal connection in Locked-Antiphase mode:

```
SPEEDL=1;
SPEEDR=1;
if(SKPS(p_up)==0)
{
    LCD_goto(20);
    LCD_putstring("Move Forward!   ");
    MD10C_L_DIR=30;
    MD10C_R_DIR=30;
}
else if(SKPS(p_down)==0)
{
    LCD_goto(20);
    LCD_putstring("Move Backward!  ");
    MD10C_L_DIR=220;
    MD10C_R_DIR=220;
}
else if(SKPS(p_left)==0)
{
    LCD_goto(20);
    LCD_putstring("Rotate to left! ");
    MD10C_L_DIR=220;
    MD10C_R_DIR=30;
}
else if(SKPS(p_right)==0)
{
    LCD_goto(20);
    LCD_putstring("Rotate to right!");
    MD10C_L_DIR=30;
    MD10C_R_DIR=220;
}
```

(b) Here is the sample code to drive two motors in variable speed with 4 types of movement in Locked-Antiphase mode:

i. Move forward (depend on motor terminal connection) by joystick:

```
else if(up_v>0)
{
    LCD_goto(20);
    LCD_putstring("Move Forward!    ");
    MD10C_L_DIR=0;
    MD10C_R_DIR=0;
    if(left_v>0)
    {
        if(up_v>left_v)MD10C_L_DIR=255-(up_v-left_v+140);
        else MD10C_L_DIR=115;
        MD10C_R_DIR=255-(up_v+140);
    }
    else if(right_v>0)
    {
        if(up_v>right_v)MD10C_R_DIR=255-(up_v-right_v+140);
        else MD10C_R_DIR=115;
        MD10C_L_DIR=255-(up_v+140);
    }
    else
    {
        MD10C_L_DIR=255-(up_v+140);
        MD10C_R_DIR=255-(up_v+140);
    }
}
```

ii. Move backward (depend on motor terminal connection) by joystick:

```
else if(down_v>0)
{
    LCD_goto(20);
    LCD_putstring("Move Backward!    ");
    if(left_v>0)
    {
        if(down_v>left_v)MD10C_R_DIR=down_v-left_v+140;
        else MD10C_R_DIR=140;
        MD10C_L_DIR=down_v+140;
    }
    else if(right_v>0)
    {
        if(down_v>right_v)MD10C_L_DIR=down_v-right_v+140;
        else MD10C_L_DIR=140;
        MD10C_R_DIR=down_v+140;
    }
    else
    {
        MD10C_L_DIR=down_v+140;
        MD10C_R_DIR=down_v+140;
    }
}
```

iii. Rotate to left (depend on motor terminal connection) by joystick

```
else if(left_v>0)
{
    LCD_goto(20);
    LCD_putstring("Rotate to left! ");
    MD10C_L_DIR=left_v+120;
    MD10C_R_DIR=255-(left_v+120);
}
```

iv. Rotate to right (depend on motor terminal connection) by joystick

```
else if(right_v>0)
{
    LCD_goto(20);
    LCD_putstring("Rotate to left! ");
    MD10C_L_DIR=255-(right_v+120);
    MD10C_R_DIR=right_v+120;
}
```

(c) Here is the sample code to stop the motor:

```
else
{
    LCD_goto(0);
    LCD_putstring("Motor State:     ");
    LCD_goto(20);
    LCD_putstring("Stop!            ");
    MD10C_L_DIR=127;
    MD10C_R_DIR=127;
    SPEEDL=1;
    SPEEDR=1;
}
```