Actionare motor CC in doua directii cu reglare de viteza
Adaptare schema cu PIC16F84 si program la PIC16F819 –PIC_AP10

www.roboprog.weebly.com

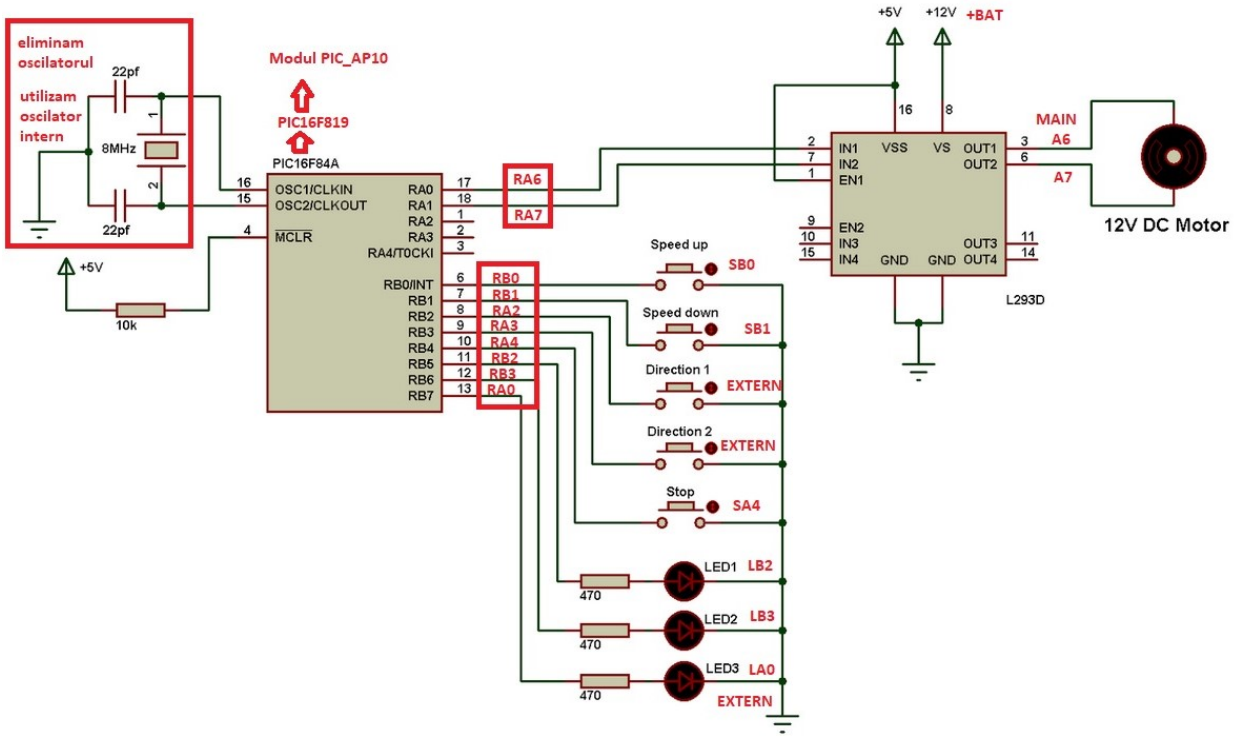
Adaptare schema

Avem urmatoarea schema pe care dorim sa o implementam. Dar fiind cu PIC16F84 dorim sa o modificam pentru a putea utiliza PIC16F819.
Pe schema am figurat cu rosu modificarile necesare (alocarea altor pini).

Oscilatorul extern este inutil pentru PIC16F819 (la 8MHz).

Iesirile pentru motor au fost alocate la RA6 si RA7.

RESET (MCLR) nu este utilizat.



Cele 5 butoane au fost alocate astfel:
-trei se gasesc deja pe modul
-doua au fost adaugate in exterior (marcate EXTERN);
-jumper-ii pentru RA2 si RA3 au fost conectati la PU.

Cele 3 LED-uri au fost alocate astfel:
-doua erau deja pe modul;
-unul a fost conectat in exterior (marcat EXTERN).

Adaptare program (in limbaj CCS C)

Programul original a fost modficat dar nu in structura sa.
Au fost realocate intrarile si iesirile.
Pentru teste, in loc de motor, se pot utiliza doua LED-uri antiparalel (desigur cu rezistor de protectie in serie).
Copiati incepind cu #use fast_io(A) in CCS C, compilati si testati.

ADAPTAT LA PIC AP10

```
// DC motor control using PIC16F84A and L293D CCS C code


//in fisierul .h
Ceea ce rezulta dupa initializarea proiectului (vedeti curs CCSC):
-oscilator intern 8MHz
-MasterClear este intrare

//in fisierul .c
#use fast_io(A)
#use fast_io(B)
#use pwm(output = pin_a6, output = pin_a7, timer = 0, frequency= 500Hz, duty = 0)

unsigned int8 i = 1;
void main() {
  port_b_pullups(TRUE);                 // Enable PORTB pull-ups
  output_a(0);                          // PORTA initial state
  set_tris_a(0b00111110);               // A7,A6,A0 iesiri; A5..A1 intrari
  output_b(0);                          // PORTB initial state
  set_tris_b(0b11110011);               // B2,B3 iesiri; restul intrari
  pwm_off();                            // Turn off all pwm outputs
  while(TRUE) {
    if(input(PIN_B0) == 0){             // If RB0 button pressed
      i++;                              // Increment i by 1 (i = i + 1)
      if(i > 99){
        i = 100;
        output_high(PIN_A0);}           // RA0 LED ON
      pwm_set_duty_percent(i * 10UL);   // Duty cycle change in tenths %
      delay_ms(100);      }             // Wait 100ms
    if(input(PIN_B1) == 0){             // If RB1 button pressed
      output_low(PIN_A0);               // RA0 LED OFF
      i--;                              // Decrement i by 1 (i = i - 1)
      if(i < 1)
        i = 1;
      pwm_set_duty_percent(i * 10UL);   // Duty cycle change in tenths %
      delay_ms(100);      }             // Wait 100ms

    if(input(PIN_A2) == 0){             // If RA2 button pressed
      if(input(PIN_B2) == 0){
        output_low(PIN_B2);             // RB2 LED OFF
        pwm_off();                      // Turn off pwm for both outputs

  output_a(0);                    // PORTA pins low
        delay_ms(100);                  // Wait 100ms
        pwm_on(PIN_A6);                 // Turn pwm on at RA6
        output_high(PIN_B2);            // RB2 LED ON
        if(i > 99)
          output_high(PIN_A0);}}
    if(input(PIN A3) == 0){             // If RA3 button pressed
      if(input(PIN B3) == 0){
        output_low(PIN_B2);             // RB2 LED OFF
        pwm_off();                      // Turn off pwm for both outputs
        output_a(0);                    // PORTA pins low
        delay ms(100);                  // Wait 100ms
        pwm on(PIN A7);                 // Turn PWM on at RA7
        output_high(PIN_B3);
        if(i > 99)
          output_high(PIN_A0);}}
    if(input(PIN_A4) == 0){             // If RA4 button pressed
      pwm_off();                        // Turn off pwm for both outputs
      output_a(0);                      // PORTA pins low
      output_b(0);}                     // PORTB pins low
    }
}
```

ORIGINAL

```
// DC motor control using PIC16F84A and L293D CCS C code

//in fisierul .h
Ceea ce rezulta dupa initializarea proiectului (vedeti curs CCSC):
-oscilator extern 8MHz
-MasterClear este doar reset la acest PIC


//in fisierul .c
#use fast_io(A)
#use fast_io(B)
#use pwm(output = pin_a0, output = pin_a1, timer = 0, frequency= 500Hz, duty = 0)

unsigned int8 i = 1;
void main() {
  port_b_pullups(TRUE);                 // Enable PORTB pull-ups
  output_a(0);                          // PORTA initial state
  set_tris_a(0b00000000);               // All PORTA pins are configured as output
s
  output_b(0);                          // PORTB initial state
  set_tris_b(0b00011111);               // Configure RB0 to RB4 as inputs
  pwm_off();                            // Turn off all pwm outputs
  while(TRUE) {
    if(input(PIN_B0) == 0){             // If RB0 button pressed
      i++;                              // Increment i by 1 (i = i + 1)
      if(i > 99){
        i = 100;
        output_high(PIN_B7);}           // RB7 LED ON
      pwm_set_duty_percent(i * 10UL);   // Duty cycle change in tenths %
      delay_ms(100);      }             // Wait 100ms
    if(input(PIN_B1) == 0){             // If RB1 button pressed
      output_low(PIN_B7);               // RB7 LED OFF
      i--;                              // Decrement i by 1 (i = i - 1)
      if(i < 1)
        i = 1;
      pwm_set_duty_percent(i * 10UL);   // Duty cycle change in tenths %
      delay_ms(100);      }             // Wait 100ms
    if(input(PIN_B2) == 0){             // If RB2 button pressed
      if(input(PIN_B5) == 0){
        output_low(PIN_B6);             // RB6 LED OFF
        pwm_off();                      // Turn off pwm for both outputs
        output_a(0);                    // PORTA pins low
        delay_ms(100);                  // Wait 100ms
        pwm_on(PIN_A0);                 // Turn pwm on at RA0
        output_high(PIN_B5);            // RB5 LED ON
        if(i > 99)
          output_high(PIN_B7);}}
    if(input(PIN_B3) == 0){             // If RB3 button pressed
      if(input(PIN_B6) == 0){
        output_low(PIN_B5);             // RB5 LED OFF
        pwm_off();                      // Turn off pwm for both outputs
        output_a(0);                    // PORTA pins low
        delay_ms(100);                  // Wait 100ms
        pwm_on(PIN_A1);                 // Turn PWM on at RA1
        output_high(PIN_B6);
        if(i > 99)
          output_high(PIN_B7);}}
    if(input(PIN_B4) == 0){             // If RB4 button pressed
      pwm_off();                        // Turn off pwm for both outputs
      output_a(0);                      // PORTA pins low
      output_b(0);}                     // PORTB pins low
    }
}
```