Tasks for encoder DSP

The encoder outputs four square wave at 39 kHz. (a,b,c,d) Each successive signal is phased 90 degrees later than the signal before it. These signals drive fingers on the encoder stator. A signal is picked up on the encoder rotor that is a sine wave (with lots of bumps) that has a phase proportional to the rotary angle of the rotor with respect to the stator. This signal repeats 28 times as the rotor rotates through one revolution with respect to the stator. The resulting signal is transferred from the rotor to the stator and decoded. The output of this module is an 8 bit parallel number that is the rotary position of the rotor with respect to the stator, mod 28.

1. Set Up PWMs — Complete
    a. PWM frequency = 39khz
    b. PWM2 offset = 90 degrees
    c. PWM1, 2, 3 set up with complimentary outputs
2. Set Up Input Capture of Encoder Signal
    a. Set up C1IN2- as the comparator1 input
    b. Connect a wire to pin 24 from the encoder output (PIN 64)
    c. Set up internal CVrefin of 1.85 volts.
    d. Set CREF to CVrefin for comparator 1
    e. Set up RP36 (pin 32) as C1OUT (use this for debug then can remove later)
    f. Set up PWM as sync for Input Capture
    g. Set up C1OUT as trigger for Input Capture
    h. Set up capture on positive trigger
3. Write loop code
    i. Check for data – loop until data available (time out if more than 30 useconds and set error flag then continue)
    ii. Read data from input capture
    iii. Calculate capture data – position = Delta P (this is an 8 bit number ignore the rollover)
    iv. If Delta P > DVmax Delta P =DVmax (set DVmax = 2 in .h file)
    v. If Delta P < -DVmax then Delta P = -DVmax
    vi. Position = Position + Delta P
    vii. Register C (0:7) = Position (output the position to the outside)
    viii. PWM3 = Position (output the position to the outside)
    ix. Wait until 5 useconds after trigger
    x. Read encoder input – wait until negative
    xi. Wait 5 useconds
    xii. Set up capture on positive edge
    xiii. Goto Loop

Tasks for Current Control DSP

The Current controller outputs three analog sine waves that are 120 degrees apart. The amplitudes of the sine waves are the commanded currents in the motor windings. As the motor rotates, the position of the motor, within an electrical cycle, (an 8 bit digital number that is an input to this function) increases or decreases providing an X axis for the sine waves. The current command is read from an analog input through an A/D. (Later in the project the current command will come over the CAN bus from the vehicle control unit.) The current command is multiplied by the sine wave to get the current command for each phase. This command is output though a D to A converter made up of a latch and a resister bank. These analog commands are compared with the current in the windings as measured by the current sensors. (the third phase is calculated rather than measured because the sum of the currents in the three phases has to equal zero) In hardware, the analog current command is subtracted from the measured current and this signal is passed back to the motor control to be used on controlling the FETs.

1. Set Up SPI on RP39,RPI58,RP38,RP3 (SCK, SDI, SDO,SS)
2. Set Up Register C (7:0) as inputs
3. Set up an 8 bit variable “position”
4. Set up Register B (15:8) as outputs
5. Set up an 8 bit variable “current_command”
6. Set up three sine tables (0, 120 degrees, and 240 degrees) that are 256 bytes long each
7. Set up pin 13 (AN0) as an analog input and as a the A/D input
8. Set up pins 33, 34,47 (RA4, RA9, RC13) as digital outputs
9. Write Loop Code
    a. Read Register C (7:0) => temp1
    b. Read Register C (7:0) => temp2
    c. If temp1 not = temp2 goto loop
    d. Position = temp2
    e. Current_Command_A = Sine_Map_A (position)
    f. Current_Command_A =Current_Command_A * Current_Command
    g. Register B (15:8) = Current_Command_A (ignore lower 8 bits)
    h. Bitset Register C (13)
    i. Wait 1 usec
    j. Bitclear Register C (13)
    k. Current_Command_B = Sine_Map_B (position)
    l. Current_Command_B =Current_Command_B * Current_Command
    m. Register B (15:8) = Current_Command_B
    n. Bitset Register A (9)
    o. Wait 1 usec
    p. Bitclear Register A (9)
    q. Current_Command_C = Sine_Map_C (position)
    r. Current_Command_C =Current_Command_C * Current_Command
    s. Register B (15:8) = Current_Command_C
    t. Bitset Register A (4)
    u. Wait 1 usec
    v. Bitclear Register A (4)
    w. Read A/D0 => Current Command (8 bits)
    x. Goto Loop

Tasks for Motor Control DSP

The Motor Control outputs pulse width modulated phase enables that turn on/off the FETs in the motor controller. This module reads in the position of the motor and calculates the velocity of the motor. It reads an analog current command in. It calculates three sine waves from this information that represent the appropriate voltages (percentage of the battery voltage) to apply to the phases to provide a torque proportional to the commanded input. If the current in any phase exceeds the commanded current the phase is turned off for the rest of that PWM cycle.

1. Set Up SPI on RP39,RPI58,RP38,RP3 (SCK, SDI, SDO,SS)
2. Set Up Register C (7:0) as inputs
3. Set up an 8 bit variable "position"
4. Set Up PWMs
    a. PWM frequency = 20khz
    b. PWM1, 2, 3 set up with complimentary outputs
    c. Set dead time of 0.5 u seconds
    d. Set PWM to update at the beginning of each cycle
    e. Set up the fault_current_limit to turn off the PWM on a current fault for each phase
5. Set up AN0, AN1, AN2 as analog (A/D) inputs
6. Set up the A/Ds to simultaneously sample in the center for the PWM.
7. Write Loop Code (this loop must complete in less than 20 useconds. Next step will be 50 kHz and this will require this loop to complete in less than 20 useconds)
    a. Loop until current data (A/D read) is available (Measured_Current(A,B,C))
    b. Delta_Current (A,B,C) = Current_Command (A,B,C) – Measured_Current(A,B,C)
    c. Voltage_Command(A,B,C) = Voltage_Command (A,B,C) + Current_Command(A,B,C)*K1 + Delta_Current (A,B,C) * K2
    d. IF Voltage_Command(A,B,or C) in negative then reverse PWM pins else set PWM pins positive (set the state for current limit fault also)
    e. PWM(A,B,C) = Voltage Command(A,B,C)
    f. Read Register C (7:0) => temp1
    g. Read Register C (7:0) => temp2
    h. If temp1 not = temp2 then loop -2
    i. Position = temp2 – Velocity * K7
    j. Current_Command = A/D 3 data (8 bits)
    k. Current_Command_A = Sine_Map_A (position)
    l. Current_Command_A =Current_Command_A * Current_Command
    m. Current_Command_B = Sine_Map_A (position)
    n. Current_Command_B =Current_Command_C * Current_Command
    o. Current_Command_C = Sine_Map_A (position)
    p. Current_Command_C =Current_Command_C * Current_Command
    q. If Carry from Position-Position_old then Position_Full = Position_Full+256
    r. If borrow from Position-Position_old then Position_Full = Position_Full+256
    s. Velocity_Sample = Position_Full – Position_old
    t. Velocity = Velocity*K3 + Velocity_Sample*(1-K3) (set K3 =0 .975 in .h file)
    u. Position_old=Position

v. Voltage Command = Velocity * K4
w. Voltage_Command_A = Sine_Map_A (position)
x. Voltage _Command_A = Voltage _Command_A * Voltage _Command
y. Voltage _Command_B = Sine_Map_A (position)
z. Voltage _Command_B = Voltage _Command_C * Voltage _Command
aa. Voltage _Command_C = Sine_Map_A (position)
bb. Voltage _Command_C = Voltage _Command_C * Voltage _Command
cc. Goto Loop

Tasks for Modular Battery Control DSP

The Modular Battery Control operates in one of three modes:

- Buck mode
  - This mode is used when the commanded voltage is lower than or equal to the battery voltage and the battery is sourcing current.
  - In this mode the controller turns on the Boost pass FETs, Turns off the Boost FETs, and PWMs the Buck FETs. The voltage at the output is equal to the percentage on time of the buck FETs times the battery voltage.
- Boost mode
  - This mode is used when the commanded voltage is higher than the battery voltage and the battery is sourcing current.
  - In this mode the Buck FETs are turned on, the boost pass FETs and the boost FETs are set up as complimentary outputs. The output voltage is equal to the 1/(the percentage of time the boost FET is on) times the battery voltage.
- Regen mode
  - This mode is used when the commanded voltage is larger than the battery voltage and the battery is sinking current.
  - The Buck FETs are turned on and the boost FETs and boost pass FETs are in complimentary mode. The voltage at the output is equal to 1/(the percentage on time of the boost pass FETs times the battery voltage).

1. Set Up PWMs
   a. PWM frequency = 200khz center aligned
   b. PWM 2 set up with complimentary outputs
   c. PWM1 set up with complimentary outputs
   d. Set PWM1 delayed 180 degrees from PWM2
   e. Set dead time of 0.5 u seconds
   f. Set PWM to update at the beginning of the cycle
   g. Set up the fault_current_limit to turn off the PWM on a current fault for each output
2. Set up AN1, AN12, AN13, and AN14 as analog (A/D) inputs
3. Set RA4, RA7, RG7,RG8,RF1,RB8,RC13,RB7,RC10 as digital outputs
4. Set R8 as a digital input
5. Write Loop
   a. Set key on if RA8 = 1 else set key off
   b. Set Commanded_Voltage = A/D 14
   c. Set Commanded_Current = A/D 13
   d. Set Measured_Voltage = A/D 12
   e. Set Measured_Current = A/D 1 (measure in the middle of a PWM pulse)
   f. Set Battery_Voltage = A/D 3
   g. If key off
      i. Goto Set Off Mode

h. If Measured_Voltage > Battery_Voltage and Commanded_Current is negative
    i. Set Regen mode (Set RC13 = 1, clear RB8 and RB7)
    ii. Set nBuck (RA7) = 1
    iii. Set nBoost (RF1) = 0
    iv. Set PWM2H = compliment of PWM2L
    v. Set PWM1H = compliment of PWM1L
    vi. Command = Commanded_Voltage / Battery_Voltage
    vii. If Measured_Current > Commanded_Current then Command=0
    viii. PWM2 = PWM1 = Command
    ix. Goto Loop
i. If Measured_Voltage > Battery_Voltage and Commanded Current is positive
    i. Set Boost Mode (Set RB7 = 1, clear RB8 and RC13)
    ii. Set nBuck (RA7) = 1
    iii. Set nBoost (RF1) = 0
    iv. Set PWM2H = compliment of PWM2L
    v. Set PWM1H = compliment of PWM1L
    vi. Command = Battery_Voltage / Commanded_Voltage
    vii. If Measured_Current > Commanded_Current then Command=0
    viii. PWM2 = PWM1 = Command
    ix. Goto Loop
j. If Measured Voltage <= Battery Voltage and Commanded_Current is positive
    i. Set Buck mode (Set RB8 = 1, clear RB7 and RC13)
    ii. Set nBuck (RA7) = 0
    iii. Set nBoost (RF1) = 1
    iv. Set PWM2H = 0
    v. Set PWM1H = 0
    vi. Command = Commanded_Voltage / Battery_Voltage
    vii. If Measured_Current > Commanded_Current then Command=0
    viii. PWM2 = PWM1 = Command
    ix. Goto Loop
k. Else
    i. Set off mode
    ii. Set Shutdown =1
    iii. Set PWMH1=PWML1=PWMH2=PWML2=nBoost=nBuck=0
    iv. Goto Loop