STEPPER MOTOR SPEED AND POSITION CONTROL

Group 8:
Subash Anigandla
Hemanth Rachakonda
Bala Subramanyam Yannam
Sri Divya Krovvidi

Instructor: Dr. Jens - Peter Kaps

ECE 511 Microprocessors
Fall Semester 2011
George Mason University

ABSTRACT

A precision stepper motor controller capable of both independent and synchronized control of a multiple number of stepper motors is discussed. The controller is built around a 16-bit microprocessor to provide fast and reliable control operations. In addition, micro stepping techniques are used to achieve high resolution electronically and to suppress mechanical resonance. The controller also includes two output ports. Such a controller has wide applications in manufacturing. For instance, it can be used to control a robot having multiple degrees of freedom. The controller was tested with a simultaneous control (synchronous) of two stepper motors for precision trajectory control applications.

Stepper motors are being used more often these days for various applications because they are economical and easy to control. They have become easy to control because of the emergence of various low cost microcontrollers which can be programmed to control them. This has enabled stepper motors to be controlled with more flexibility and versatility for various applications.

The reason stepper motors are becoming more prevalent is because of their ease of interfacing with digital components. A stepper motor moves one step for every pulse given to it. This enables open loop control of the position of the stepper motor. Unlike in other AC or DC motors, stepper motors do not need closed loop systems for position control. Also, since a stepper motor requires digital control pulses, there is no need for analog to digital conversion circuitry usually required for AC and DC motors. This makes Stepper Motors very economical and easy to control.

The objective of the project is to design and develop microcontroller based Stepper Motor controller for speed and position control, that will smoothly control the rotation of a stepper motor, taking into account the physical constraints on the maximum operating speed of the motor.

This project describes the process by which a control circuit for a stepper-motor is being designed using an MSP-430 microcontroller. The main features of the control circuit are speed and position control. We intend to achieve this objective by using the following components.

Major Components used:

1. *Texas Instruments MSP-430 Microcontroller*
2. *4x4 Numeric keypad*
3. *Stepper motor driver circuit*
4. *7 segment LED display*
5. *Bipolar/Unipolar Stepper Motor*

TABLE OF CONTENTS:

1. MOTIVATION:

Positioning systems have traditionally been implemented using DC motors, AC servo motors, Synchronous motors, Stepper motors, etc. DC motors are relatively easy to control. However it has disadvantages in using such motors for positioning systems like overheating of the armature windings. Also the torque to inertia is relatively low.

For the above reasons positioning systems are now being implemented using stepper motors. Usually stepper motors were designed to provide precise positioning control within an integer number of steps. They have stable open loop operation to any step position and consequently no feedback is needed to control them.

The following are some of the applications of robotics in different fields.

- Robots in Industry
  Automobile, Laboratories, Military, Mining, etc
- Industrial applications of Robots
  Material Handling, Material transfer, spot welding, Spray coating, etc
- Robots in space
  Robots in Hazardous environments
- Medical robots
  Robotic assistant for micro surgery

Using Robotics for these applications saves lot of time and manpower.

Motivation for this project comes from our desire to build a multiple axis motor controller. This can be assembled by using a combination of Stepper motor, a keypad for input, motor driver circuit and a seven segment LED to display the speed or position respectively.

A precision stepper motor controller capable of both independent and synchronized control of 2 stepper motors is built around a 16-bit microcontroller MSP-430G2452 to provide fast and reliable control operations. The controller also includes 2 output ports. This has wide applications in manufacturing, robotics, actuators used in industrial and position of various laboratory systems, etc.

2. BLOCK DIAGRAM:

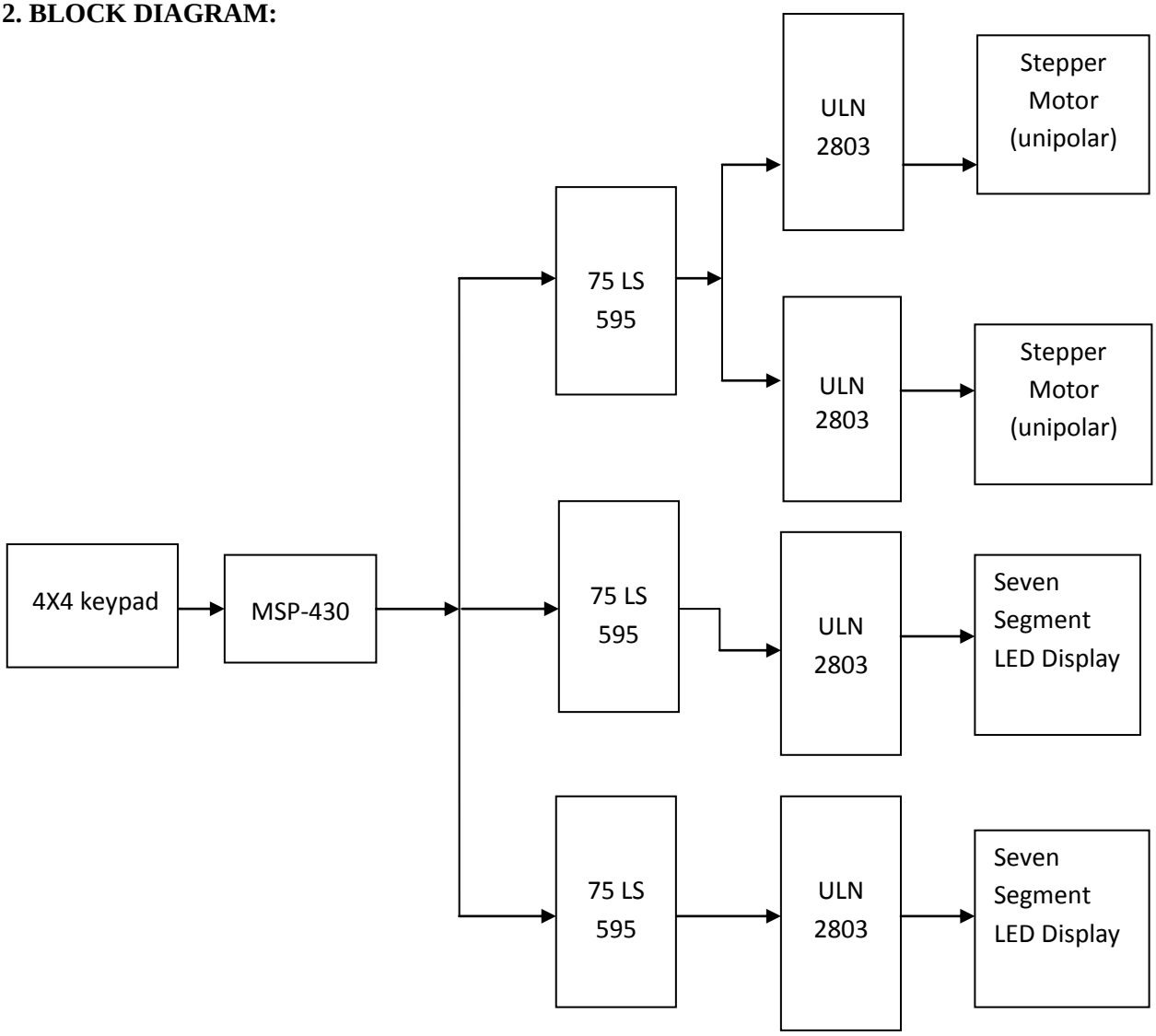


Fig: Block diagram of Stepper motor controller

The above figure represents the block diagram of a stepper motor Position and speed control. MSP 430 is used as processing unit to control the stepper motor. It provides the control output required for the speed and position control of the motor. The 4 x 4 numeric keypad is used to provide the required inputs to the MSP 430. The inputs are mode selection – speed control or position control and the actual value of the position or speed commanded. The output is an LED display. The magnitude of the speed or position is indicated using an LED display.

The MSP 430 has 2 ports, of which 1 port is interfaced directly to the numeric keypad. Out of the 8 pins, 4 pins are configured as inputs and 4 pins are configured as output pins. The 2nd port of the MSP is connected to three cascaded serial Shift Register, which are being used to expand one 8-bit port of the microcontroller to three 8-bit ports. One shift register is used for the 2 motors and the other two shift registers each are connected to the seven segment LED displays each. A Darlington array is used as an interface between the motor and the shift register.

3. MSP430 MICROCONTROLLER:

A microcontroller is an integrated circuit package (IC) that combines a small computer with clocks, voltage-measuring circuitry, and other stuff useful in controlling external devices. The microcontroller can be programmed to execute instructions. The program is written on your PC in a high-level language such as C, and then converted to code the microcontroller can understand. This code is then downloaded to the micro controller, where it remains until replaced with a new program. The MSP430G2452 family is designed for battery-powered applications and features very low power consumption from a 3V supply.

The Texas Instruments MSP430G2452 microcontroller device features a powerful 16-bit RISC CPU, 16-bit registers, with 62.5-ns Instruction Cycle Time. It has an 8KB flash memory and also has 2 IO ports. One port is connected to the 4X4 Keypad, 4 pins are configured as outputs and the remaining four are configured as inputs. The other port is connected to the stepper motor and the Seven-segment LED display through port expansion and Driver circuit.

The MSP430 is used as the main processing unit for the calculation of the pulses required to control the stepper motor. It provides the control output required for the speed and position control of the motor. The coding for this application is written in C using the IAR embedded workbench.

3. STEPPER MOTOR:

A stepper-motor is not like a normal motor that runs on direct current. It does not move continuously but rather in small increments. A stepper-motor has two coils that are connected to four inputs as shown in figure below. Each small increment that the motor turns is produced by energizing one of the two coils. The next increment is created by a energizing the other coil. If the motor needs to turn in the opposite direction, then the each state of the coils must be implemented in reverse order, that is the coil must be energized with reverse polarity.

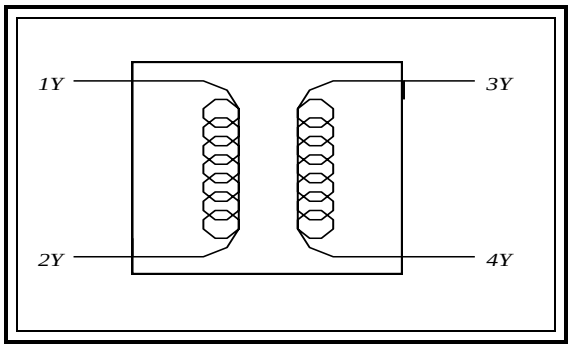


Fig: The Stepper-Motor

There are several designs of stepper motors. A unipolar stepper motor is really two motors sandwiched together. Each motor is composed of two windings. Wires connect to each of the four windings of the motor pair, so there are eight wires coming from the motor. The commons from the windings are often ganged together, which reduces the wire count to five or six instead of eight.

Hardware interfacing: As the operating voltage of the Stepper Motor is 5V and the VCC of MSP is only 3V, MSP cannot generate the required current to run both the motors. So Separate power source with a VCC of 5V is used for this. To limit the current flow between MSP and Stepper motor, ULN2803 is used as an interface. For port multiplexing 74HC595 shift register is used between ULN2803 and MSP430.

4. STEPPER MOTOR DRIVER CIRCUIT:

The stepper motor cannot be directly driven using the MCU I/O pins as the MCU cannot supply the required current to drive the Stepper Motor. Also, the Stepper Motor will cause a Back EMF in the circuit while it is accelerating or decelerating. This can cause to MCU to be damaged. Hence we use a Driver circuit which isolates the Stepper Motor circuit from the MCU circuit. The driver circuit must be able to withstand the current required by the stepper motor. For this purpose, we intend to use darlington arrays to isolate the MCU from the Stepper Motor and to provide the current required for the Stepper Motor.

5. NUMERIC KEYPAD:

The Numeric keypad used is a 4x4 matrix keypad. This keypad is used to first select the mode of operation – position or speed control and then is used to enter the required value of position or speed respectively. There will be a Start/Stop button to enable the user to stop the motor before changing modes. In this application, a 4x4 matrix keypad requiring eight Input/Output ports for interfacing with the microcontroller. Rows are connected to Peripheral Input/Output (GPIO) pins configured as output. Columns are connected to GPIO pins configured as input with interrupts. The corresponding hexadecimal value of the pressed key is sent onto the LEDs.

After debouncing is completed, a detailed scan is executed. A second fast scan is done to assure that any detection made during the first fast scan stage was valid. Then, rows are configured as inputs. When a key is pressed a high level is applied in the corresponding row.

Hardware Interfacing : Interfacing of keypad to MSP is straightforward. Port2 of MSP is exclusively used for this. Port 2, pins 0 to 3 were used as output to columns and port 2, pins 4 to 7 were used as input from rows

6. DISPLAY:

A seven segment display, as its name indicates, is composed of seven elements. Individually on or off, they can be combined to produce simplified representations of numbers. The magnitude of the speed or the value of the position is shown to the user by means of an LED display. The magnitude of speed is proportional to the number of LED's in active state. A Seven Segment LED display can also be used in place of the LED bar display without much change in the circuit.

Hardware interfacing: Interfacing of seven segment Display with MSP430 is done through 74HC595 shift register which is used for port multiplexing.

7.SOFTWARE INTERFACING:

The software interfacing is mainly segregated into three parts:

1.Motor control
2.Display
3.Keypad

Brief procedure:

1. Initiall the values are entered inputed from the keypad. The values entering will be individual speed/position of each motor and the third selection will be mode seltion i.e. the speed or position control.
2.The values are stored into the registers now and then are redirected to the LED diplay segments to display the values entered via the keypad.
3.Now based on the mode selected motor operation takes place

Motor:

The values thus entered from the keypad are stored in the registers and are now used to calculate the no: steps if the motor mode off operation is selected as position control, and if the mode of operation is selected as speed control then the delay between the steps are calculated. The values thus calculated are now sent into the respective motor sub-routine which now generate the required stepping sequence to drive the motor according to the input configurations. The function that is supposed to generate the values takes in the direction and speed as input and generates the delay that has to be inserted between pulses and the amount of steps the motor has to rotate.

Display:

The seven segment display is used as out displaying device. For the display in the coding part based on the appropriate number selection is checked using a switch function and the matched data is sent out to the display corresponding to the number. These are represented in the decimal form in the code.

Keypad:

For the keypad function we are mainly using he functions keypad(), keyeyfind(). The keypad function is useed too identify which keys are inputed on the keypad and the corresponding key value is written into the function for the further proceedings i.e. the identification of the mode and the steps that the motor has to run according to the inputed values on the keypad. This whole operation is determining which key has gone high is identified by the kypad function

8. RESULTS AND CONCLUSIONS:

The project was completed to the given specifications. Had we been a bit ahead of the schedule we could have managed to perform better. The problem faced were compatibility issues and the lack of memory. While testing the individual components code wasn't too long, hence sufficient to fit the MSP430 we ordered initially. But after integrating the code we had the memory issue. Also due to the large number of inter connections between the different components we faced problems with signal flow and also ended up burning a few boards and IC's.

No complains though, as we have learnt a lot from this experience. We learnt how to build a real time system and how to code. We can add that we did learn to manage time and team work.

APPENDIX

DIVISION OF WORK:

Bala Subramanyam Yannam & Sri Divya Krovvidi
Development of Stepper control code – Speed, position and direction control routines

Hemanth Rachakonda & Subash Anigandla
Development of MCU-Peripherial interface circuits and Integration of the system

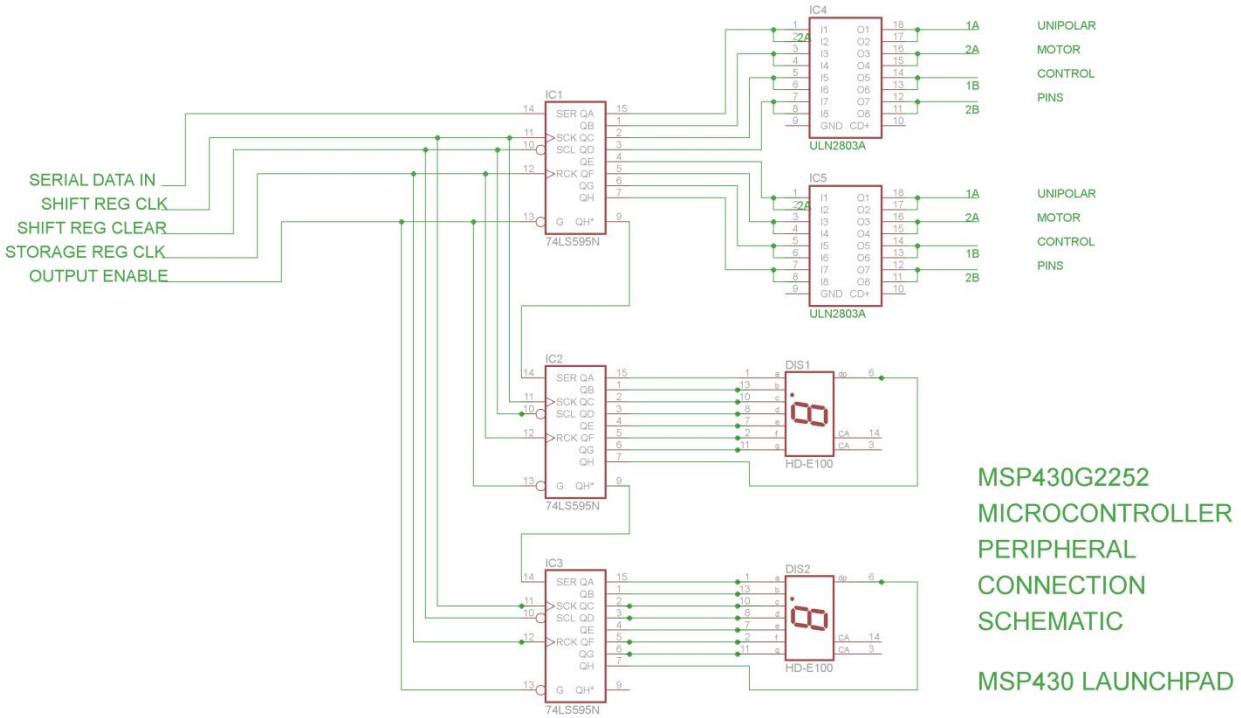
Sri Divya Krovvidi & Hemanth Rachakonda
Development of LED bar display / Seven Segment LED display control code

Subash Anigandla & Bala Subramanyam Yannam
Development of 4 x 4 Matrix Code interface code and Low Power Mode code

COMPONENTS LIST:

MSP430G2452	→ 1
74LS595N	→ 3
ULN2803A	→ 4
STEPPER MOTOR	→ 2
Seven Segment LED Displays	→ 2
EASY DRIVER BOARD	→ 1 (burnt so had to replace it with ULN 2803)
Power supply Unit	→ 1

SCHEMATIC:



MSP430G2252
MICROCONTROLLER
PERIPHERAL
CONNECTION
SCHEMATIC

MSP430 LAUNCHPAD