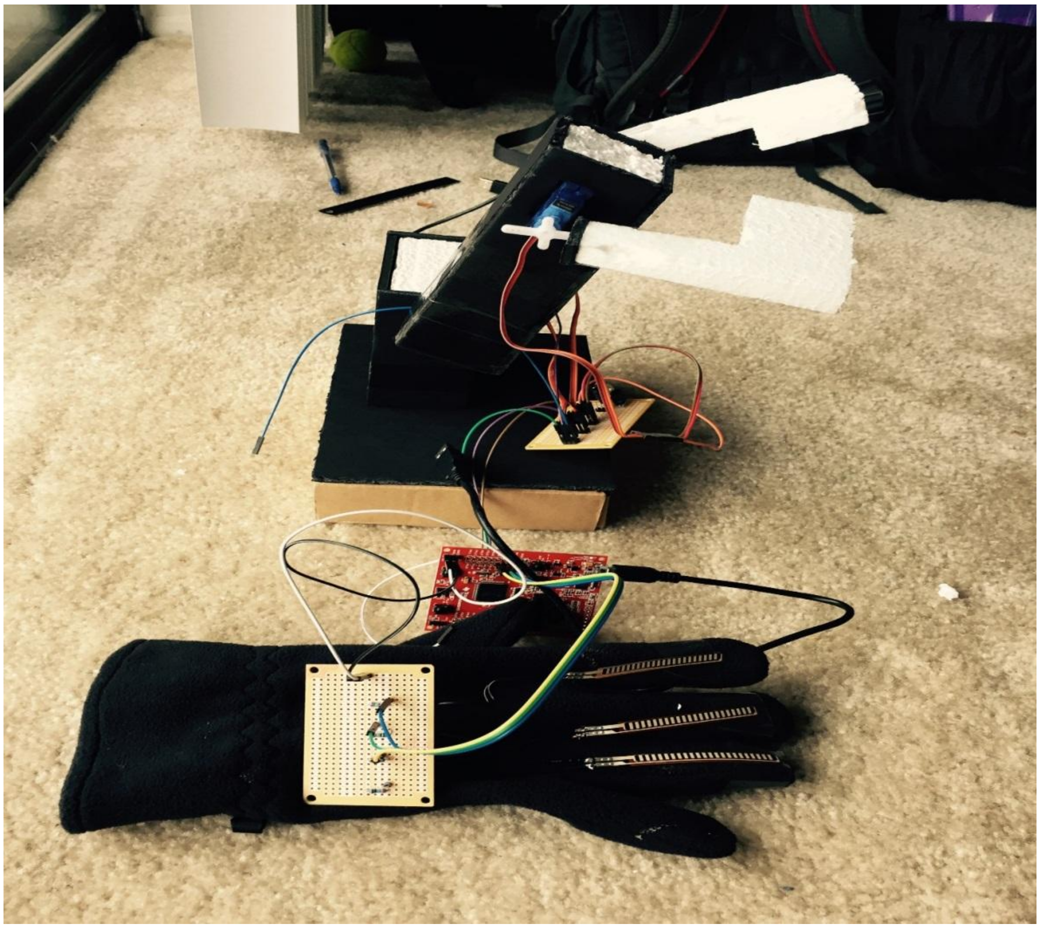
*ECE 511 Project Report*

# Robotic Claw using MSP430




**Group: 9**

**BogadiPreetham Reddy**

**Venkat Raman Sriperumbudur**

**Koushik Reddy Rokkam**

**Dibyojyoti Mukherjee**


**On December 2nd, 2014**

**Instructor: Prof. Jens-Peter Kaps**

**Abstract-**

The objective of this project is to build a robotic claw that can be controlled by flex sensors that will be put on the hand of the user. The user will be able to control the robotic claw by wearing the controller glove and perform hand gestures.

For this project we are going to use a MSP 430 Microcontroller on the glove of the user to control the Robotic Claw. The Controller Glove is fitted with Flex Resistors set up as a Voltage Divider. The Voltage Divider output goes into the MSP 430 analog input pins which send the output as PWM Signals to the respective Servos thus enabling the user to control the robotic claw with the flex sensors.

The List of Components are as follows:

1. Flex Resistors
2. Voltage Divider
3. MSP 430 Microcontroller
4. Servos for the Robot Claw
5. Power Supply

**Contents** **Page No**



**List of figures**

## 1. Introduction:-

Our project is to build a robotic claw controlled by using a glove to which are the flex sensors (variable resistors) are attached.

### 1.1 Motivation

The project was inspired by our shared interest in robotics as well as fun and interesting methods in interfacing the hardware. Despite our interest the MSP 430 microcontroller was a challenging component when we came across the hardware/software interfacing. But, the robotic arm was built and happens to work efficiently.

### 1.2 Basic Block Diagram:-

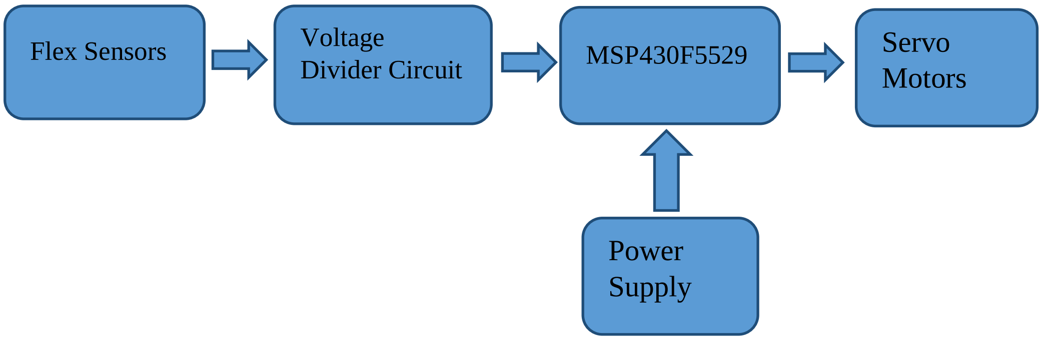



Fig A: Block Diagram

## 2. Hardware Description:-

### 2.1 Flex Sensors-

Flex Sensors are those sensors which change in resistance depending on the amount of the bend of sensor. They convert change in bend to electrical resistance the more they bend, the more is the resistance value. The flex sensor we use here is a unidirectional sensor with a length of 2.2 inch. This is interfaced with the printed circuit board (PCB) which already has a voltage divider circuit constructed on the PCB.

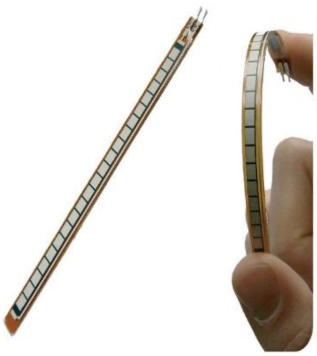


Fig B: Flex sensors

**2.2 MSP 430 Microcontroller-**

It is a mixed-signal microcontroller, family from Texas Instruments (TI). It has a 16-bit CPU which was designed for low cost and specifically low power consumption. The MSP we are using here is MSP430F5529.

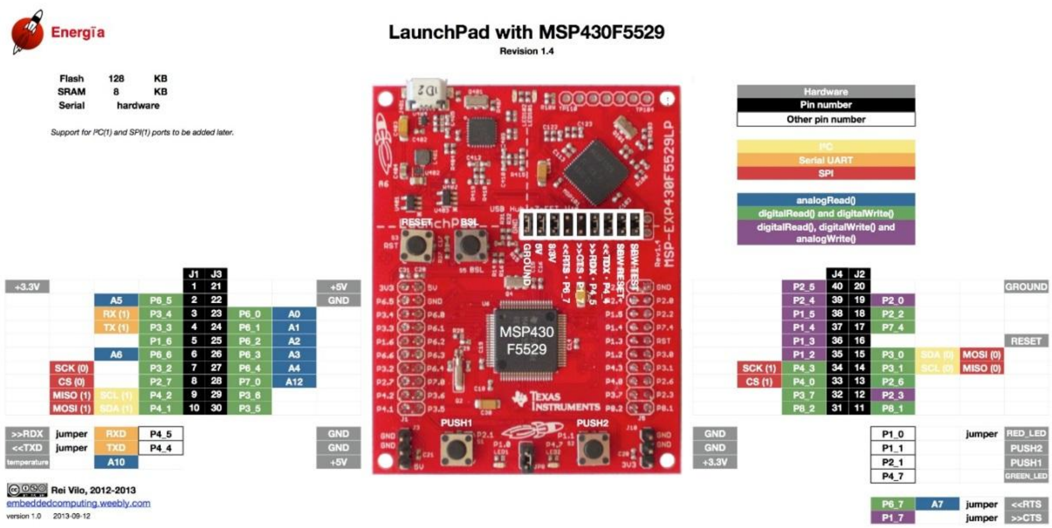



Fig C: MSP-EXP430F5529LP

The ultra-power consumption modes of the microcontroller are basically four such as the active mode, standby mode, off mode, shutdown mode. It is a 16-bit RISC architecture with fully integrated LDO having programmable regulated core supply voltage. The data sheets of the MSP will be provided at the end of the report.

**2.3 Servo Motors-**

The servo motor is a simple DC motor whose position is controlled by the MSP430. The servo motor induced is an eRC 9gm Precision Micro Servo powered with 5V whose control pins are connected to P2.5, P2.4, and P1.5 on the microcontroller.


Fig D: Servo Motor

### 2.4 Data Structures-

Array structures are used quite extensively in our code, since we have three servos to control, hence 3 resistors are used to control them, 3 commands to send to each servo.

There are two basic hardware stages in our design. The first stage is the interaction of the variable flex resistors on the glove and the second stage controls the micro-servos. In the first stage, the series of voltage dividers go from $V_{cc}$ to the GND of the MSP430. These dividers have the output into three analog input ports on the MSP430 and the detailed description of the input configuration is shown in the software description section.

In the second stage, the servos are connected in parallel to a 5V battery pack. The GPIO pins on the MSP430 emanate Pulse Width Modulation (PWM) signals through signal wires which actually control the servos.

## 3. Software Description:-

### 3.1 Routines-

*Analog Input*

We wrote a code which receives the analog signals from the flex sensors and these Analog signals (through wires) are connected to the ports P6.0, P6.1 and P6.2 on the microcontroller. The output signals to the servo motors are in the form of PWM signals. So, these PWM signals are taken form GPIO pins which are P2.5, P2.4, and P1.5 on the same Microcontroller

We have a routine that handles all the low-level logic required to read values. This was originally a large collection of groups of statements which we managed to compress into a concise loop.

*Data Conversion*

Each time we send a value from the resistor, it needs to be translated to a value sent to the servo motor. We perform this on the receiving end of things just before actually writing to the servos via Pulse Width Modulation (PWM). These turned out to be variations in the values read from different resistors, but after much experimentation, we made a few compromises and came up with a simple linear formula that works reasonably well for each resistor.

*Pulse Width Modulation (PWM)*

The period of the PWM signal for each of the servo is found roughly 20,000 cycles. The use of PWM technique is to control the angular position of the servo motor shaft which can be conveniently controlledby a microcontroller by sending a proper PWM signal to the respective servo motors.

## 4. Testing:-

Our testing strategy essentially amounted to incremental testing. We opted for bottom-up implementation, testing each new feature as it was added. For example, we started with testing of flex sensors. We tried actuating the servos if it could read values from the variable resistors. We finally incorporated and tested all the servos being used simultaneously and found all of them could move at once. Servos were found to be running on an approximated torque of 2 kg/cm. MSP was run with basic programs and found to be responding efficiently.

Since our main project was largely just a stream of data, there were not many software corner cases to check. But the real corner cases were the physical ones such as trying to move all 3 servos at once to test if we had sufficient power.

## 5. Challenges and other initiatives:-

We achieved what we proposed, with some slight simplifications. Firstly, we decided that concurrency would be largely unnecessary. In fact, for the resistors, concurrency proved not to be a plausible way to sample the resistor values, the source of which was selected via yet another register. It was only possible to read the resistors in sequence.

Furthermore, we decided not to use concurrency to implement PWM for controlling multiple servos at once. PWM actually turned out to be the most complicated piece of the design. Not only did we have to figure out all the parameters, i.e., minimum and maximum duty cycle as well as period both mathematically and experimentally, but we had to generate three PWM signals at once. We tried using concurrency, but noticed that it introduced overhead due to context switching that would reduce precision in the timing of the waves we were sending. The way we are handling the PWM now turns out to be simple and effective.

## 6. Results and Conclusions

From this project we learnt the working and programming of a microprocessor in real time, also learnt working of the servo motor and its interfacing.

One of the biggest challenges was developing code for the servo motors to run. We developed a code which responds to the analog signals from the flex sensors and sends PWM signals to the respective servo motors. Each member of the team worked hard on their respective tasks carefully which finally resulted in the successful execution of the project.

## 7. References


For documentation, we primarily consulted the MSP430 User's Guide, Data Sheet, and Development Kit User's Guide.

We also used the link for extra information on the microcontroller and other parts of our project.

- http://www.ti.com/product/msp430f5529 ~ MSP430F5529
- https://www.sparkfun.com/products/10264 ~ Flex Sensors
- http://www.jameco.com/webapp/wcs/stores/servlet/Product_10001_10001_2214601_-1~ Servo Motors
- http://www.amazon.com/gp/product/B00K38NMVI/ref=oh_aui_detailpage_o03_s00?ie=UTF8&psc=1 ~ Voltage Regulator
- http://forum.arduino.cc/index.php?topic=181222.0~ source code


## 8. Appendix

### 8.1 Task division

- Preetham: Sensors & motor testing, interfacing, final testing.
- Venkat:Hardware assembly and integration and Report Writing.
- Dibyojyoti Mukherjee: Firmware coding, Report Writing.
- Koushik Reddy: Hardware Selection, PCB designing.

### 8.2 List of the parts

- Flex sensors.
- Voltage Divider Circuit.
- MSP430 Microcontroller.
- Servo Motors.
- Power Supply.

### 8.3 Schematic diagram

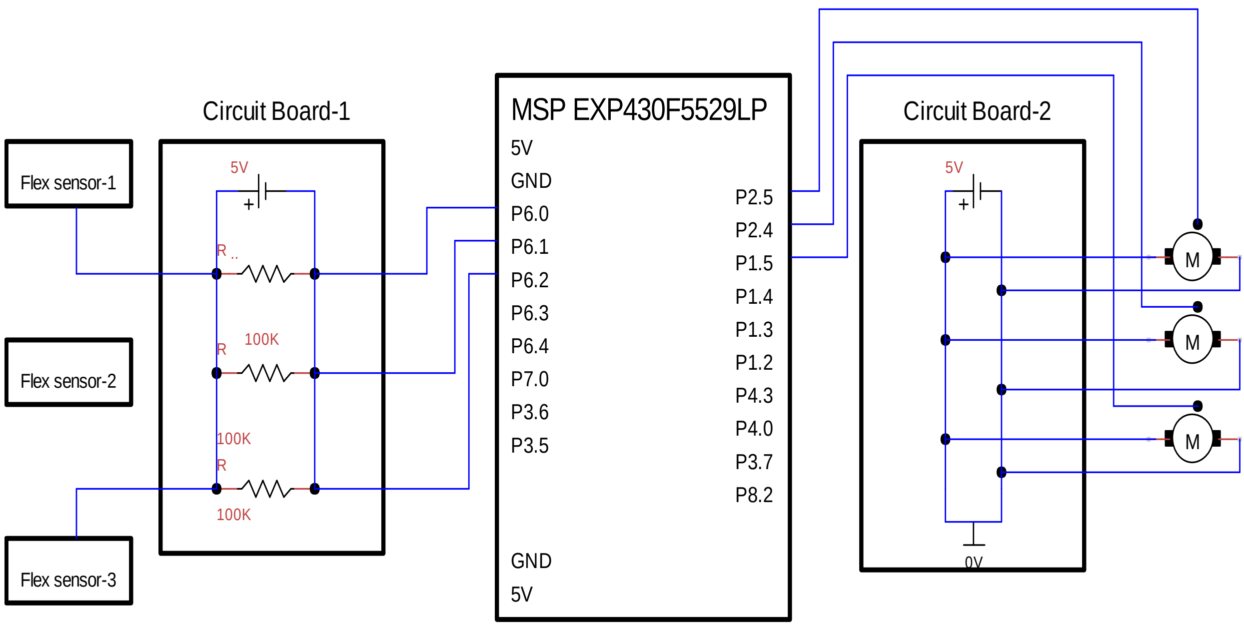



Fig E: Schematic diagram