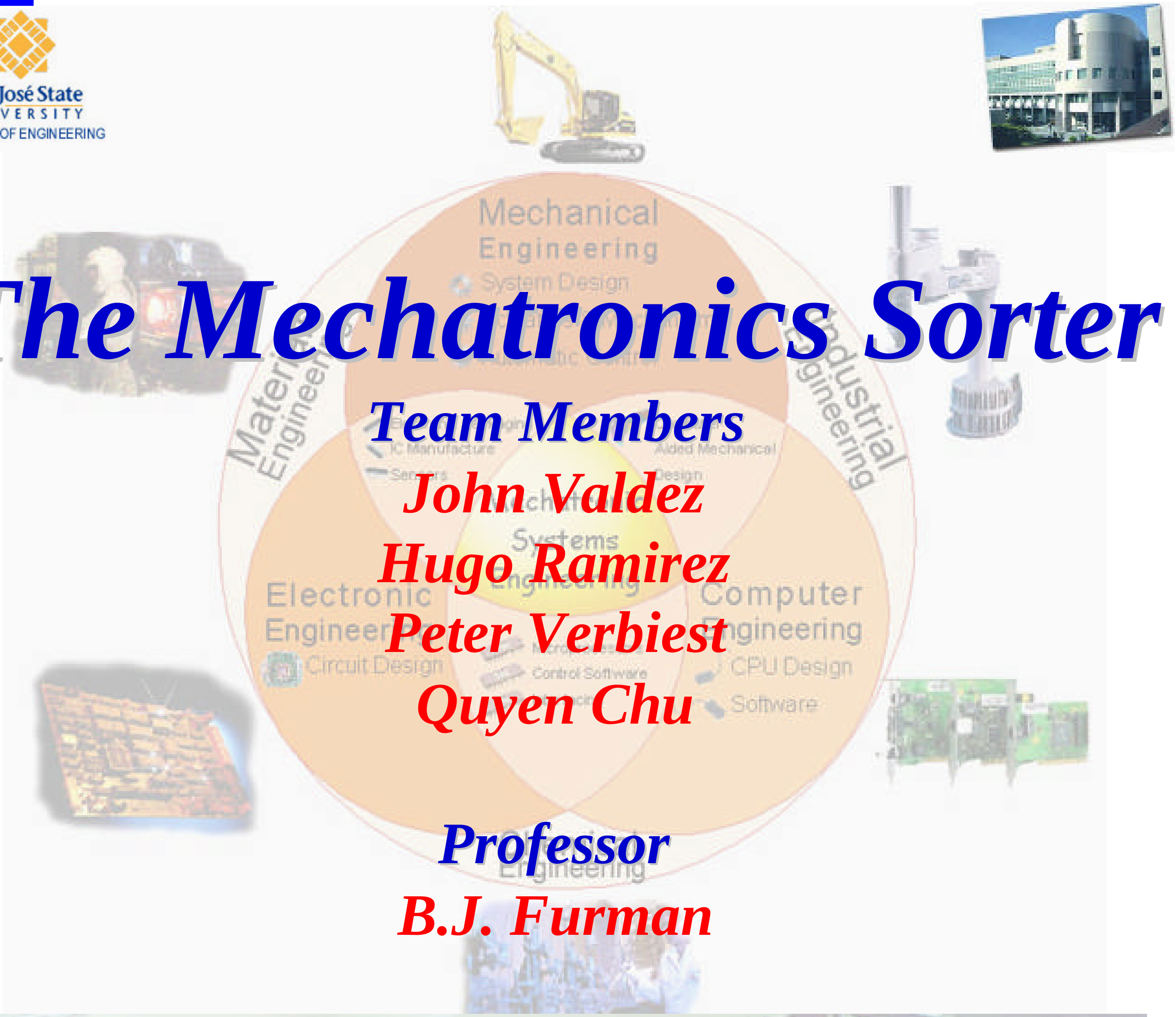
San José State
UNIVERSITY
COLLEGE OF ENGINEERING



# The Mechatronics Sorter

## Team Members

John Valdez
Hugo Ramirez
Peter Verbiest
Quyen Chu

## Professor

B.J. Furman

## Course

ME 106

## Date

12.9.99

# Table of Contents

# *Project Overview*

## Introduction

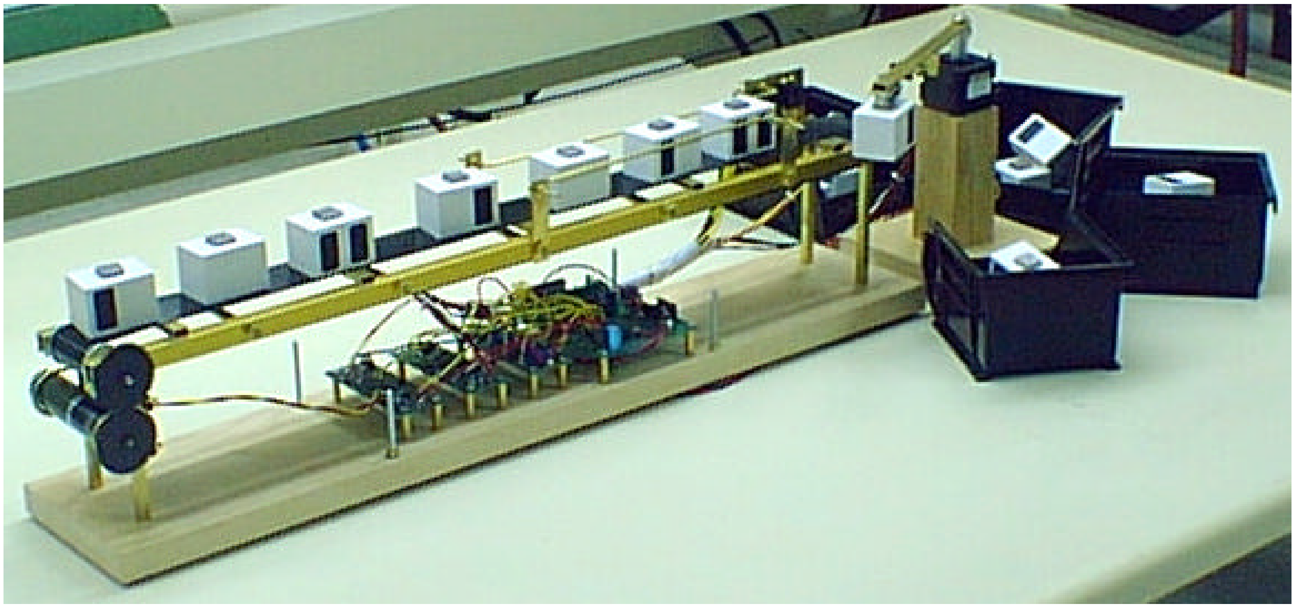
Mechanical Engineering is a versatile field that covers many aspects of everyday life. Without the many devices designed by Mechanical Engineers we would not be able to enjoy the simple pleasures of life such as going for a drive or living in an air conditioned home. Mechanical Engineering is an old discipline, nevertheless the field is still as innovative as the name suggests. Furthermore, Mechanical Engineering has now taken a whole new approach to the design of better devices embraced in the principles of MECHATRONICS. This approach demands the design of devices that do not just perform repetitive tasks but that also have the ability to respond to environmental changes. This is done with the integration of sensors, motors, microcontrollers, and other electromechanical devices used in the electronics and mechanical fields. Ultimately, MECHATRONICS allows the design of more reliable, efficient, and sophisticated systems.

## Mechatronics Applications

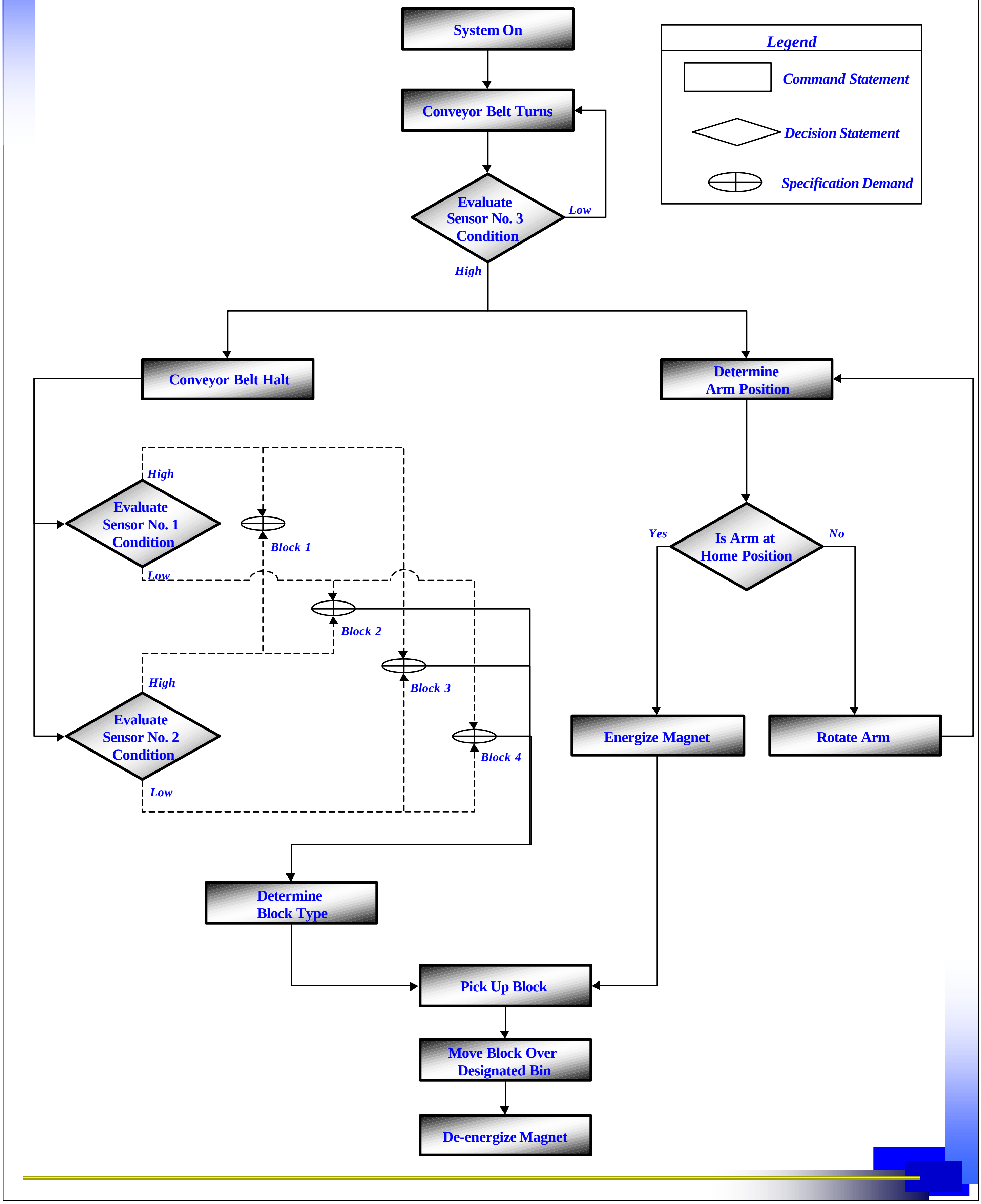
Devices designed under the principles of MECHATRONICS range from toys to satellites. The complexity of each device range from application to application, however the common element in every system is that it integrates electronic and mechanical components. An example of this is the conveyor belt system constructed in this project. The system is designed to sort small blocks according to the color combinations printed on their sides. To accomplish this task, the conveyor belt is equipped with four sensors. The first sensor is used to detect the block as it travels along the conveyor belt. Once the block is detected, a signal is sent to a microcontroller. The microcontroller used for this application is the Parallax BASIC Stamp. The signal informs the conveyor belt to stop and read the code printed on the block. The code is comprised of a combination of two colors, black and white, that is read by two light sensors. The microcontroller then deciphers the code and determines the appropriate bin for drop-off. The final sensor is used to determine the home position of the electromagnetic arm as it picks and places the blocks.

Though the principle of the system appears simple,the challenge was to integrate the design, construction, electrical circuit layout, and programming into a cohesive, working apparatus. This is just one application in which the application of MECHATRONICS has simplified by automating a repetitive task of sorting out items.

## The Mechatronics Sorter



# *Theory of Operation Flow Chart*



*Legend*

*Command Statement*

*Decision Statement*

*Specification Demand*

System On

Conveyor Belt Turns

Evaluate Sensor No. 3 Condition

*Low*

*High*

Conveyor Belt Halt

Determine Arm Position

*High*

Evaluate Sensor No. 1 Condition

*Low*

*Block 1*

*Block 2*

*High*

Evaluate Sensor No. 2 Condition

*Low*

*Block 3*

*Block 4*

*Yes*

Is Arm at Home Position

*No*

Energize Magnet

Rotate Arm

Determine Block Type

Pick Up Block

Move Block Over Designated Bin

De-energize Magnet

# Sorter Program using PBASIC Language

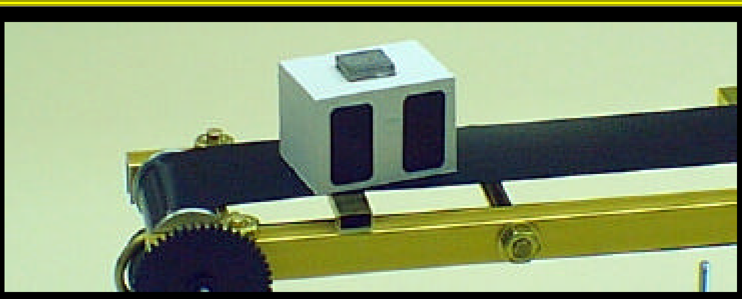
```
symbol cnvr=pin0           'Define label "cnvr"(belt switch)  for I/O pin 0.
symbol sensor1=pin3        'Define label "sensor1"  for I/O pin 1.
symbol sensor2=pin2        'Define label "sensor2" for I/0 pin 2.
symbol sensor3=pin1        'Define label "sensor3" for I/O pin 3.
symbol sensor4=pin4        'Define label "sensor4" for I/O pin 4.
symbol magswtch=pin5       'Define label "magswtch" for I/0 pin 5.
symbol stprdrctn=pin6      'Define label "stprdrctn" for I/O pin 6.
symbol stprstp=pin7        'Define label "stprstp" for I/O pin 7.

dirs=%11100001             'Define ports 1-4 as inputs and pins 5-7, 0 as outputs.
pins=%00000000             'Initialize all pins as low.

conveyorbelt:              'Loop that senses a block.
       IF sensor1=0 THEN checkstate      'Sensor1 blocked branch to checkstate.
       IF sensor1=1 THEN state           'Sensor1 not blocked branch to state.

       state:                            'Turns conveyor belt on.
              cnvr=1                     'Sends a high signal to conveyor control
                                          pin.
       goto conveyorbelt                 'Loops back into conveyorbelt loop.

checkstate:                              'Function that check if the arm is home.
              pause 91                   'wait for 91 milliseconds.
       cnvr=0                            'Turns conveyor belt  off.

       IF sensor4=1 THEN reset           'Arm not in place branch to reset.
       IF sensor4=0 THEN arm             'Arm in place branch to  arm.

arm:                       'Function, checks  for possible cases.
       IF sensor2=1 AND sensor3=1 THEN bin1    'First case branch to bin1.
       IF sensor2=1 AND sensor3=0 THEN bin2    'Second case branch to bin2.
       IF sensor2=0 AND sensor3=1 THEN bin3    'Third case branch to bin3.
       IF sensor2=0 AND sensor3=0 THEN bin4    'Fourth case branch to bin4.

              bin1:                      'White-White case move are to bin one.
                     pause 500           'wait for half a second.
                     magswtch=1          'Turns magnet on.
                     pause 500           'Wait for half a second.
                     stprdrctn=0         'Direction of motor set to turn CCW.

                     For b1=1 to 20      'Step counting loop (20 steps).
                     PULSOUT 7,25        'Pulses motor with a 25 millisecond
                     pause 25            'pulse for 25 milliseconds.
                     NEXT                'Loop back until b1=20 (20 steps).

                     goto magcntrl       'Branch to magnet control function
                                          (magcntrl).

              magcntrl:                  'Magnet control function.
                     pause 1500          'pause for one and a half seconds.
                     magswtch=0          'Turn magnet off.
                     pause 300           'Wait .3 seconds.
                     stprdrctn=0         'Direction of motor is set CCW.
```

# Sorter Program using PBASIC Language (con’t.)

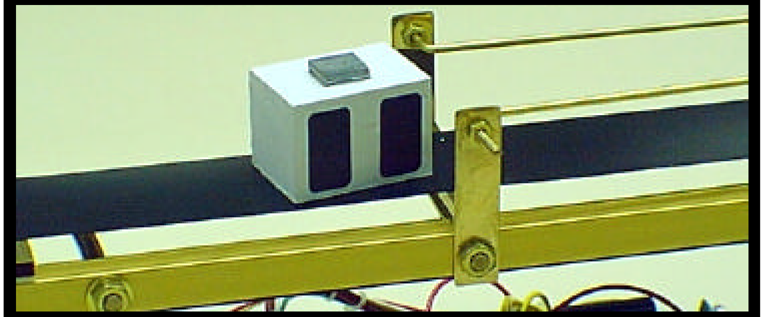
```
       goto reset          'Branch to reset function.

reset:              'Reset function checks if arm is home.
       stprdrctn=1         'Set direction of motor to CW.

       PULSOUT 7,20        'Pulses motor with a 20 millisecond
       pause 25            'pulse for 25 milliseconds.

       IF sensor4=0 THEN conveyorbelt    'Arm in place branch to
                                         converbelt.
       IF sensor4=1 THEN reset    ’Arm not in place branch to reset.

bin2:               'White-Black case.
       pause 500           'wait for half a second.
       magswtch=1          'Turn magnet on.
       pause 500           'Wait for half a second.
       stprdrctn=0         'Direction of motor is set to CCW.

       For b1=1 to 40      'Step counting loop (40 steps).
       PULSOUT 7,25        'Pulses motor with a 25 millisecond
       pause 25            'pulse for 25 milliseconds.
       NEXT                'Loop back until b1=40 (40 steps).

       goto magcntrl       'Branch to magnet control function
                            (magcntrl).
bin3:               'Black-White case.
       pause 500           'wait for half a second.
       magswtch=1          'Turn magnet on.
       pause 500           'Wait for half a second.
       stprdrctn=0         'Direction of motor is set to CCW.

       For b1=1 to 60      'Step counting loop (60 steps).
       PULSOUT 7,25        'Pulses motor with a 25 millisecond
       pause 25            'pulse for 25 milliseconds.
       NEXT                'Loop back until b1=60 (60 steps).

       goto magcntrl       'Branch to magnet control function
                            (magcntrl).

bin4:               'Black-Black case.
       pause 500           'wait for half a second.
       magswtch=1          'Turn magnet on.
       pause 500           'Wait for half a  second.
       stprdrctn=0         'Direction of motor is set to CCW.

       For b1=1 to 80      'Step counting loop (80 steps).
       PULSOUT 7,25        'Pulses motor with a 25 millisecond
       pause 25            'pulse for 25 milliseconds.
       NEXT                'Stay in loop until b1=80 (80 steps).

       goto magcntrl       'Branch to magnet cntrl loop(magcntrl).
```

***End of Program***

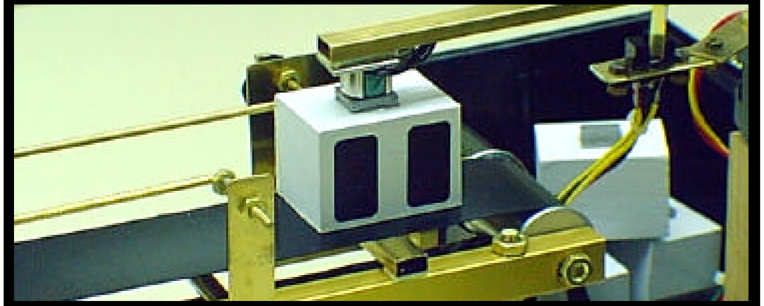
# *Step by Step Operation*



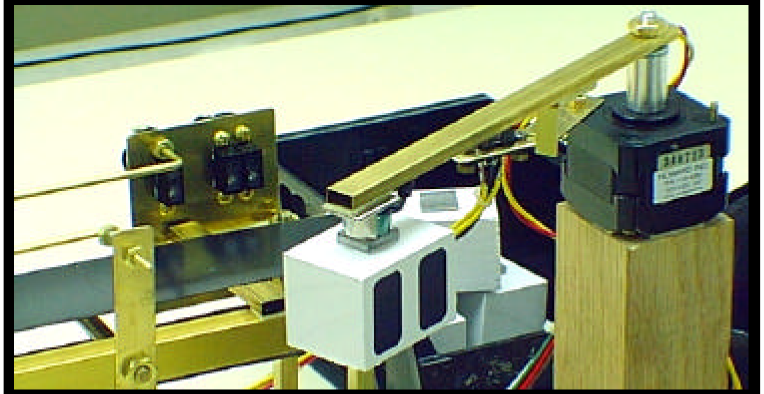
1) Block set on Conveyor



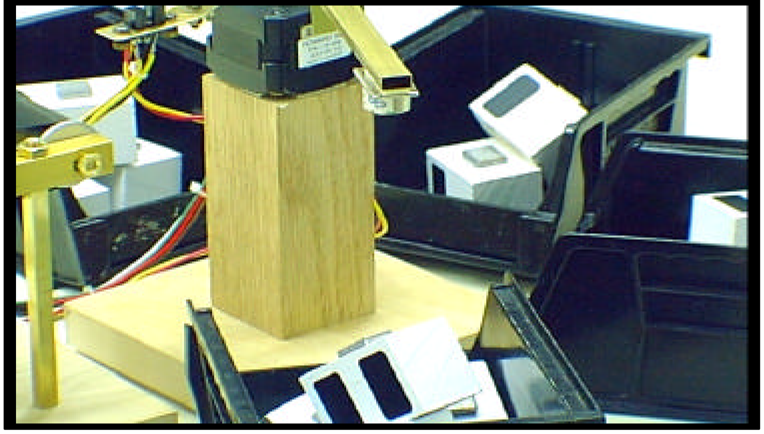
2) Block Enters Guides



3) Sensors Read Color Code & Electromagnet Energizes



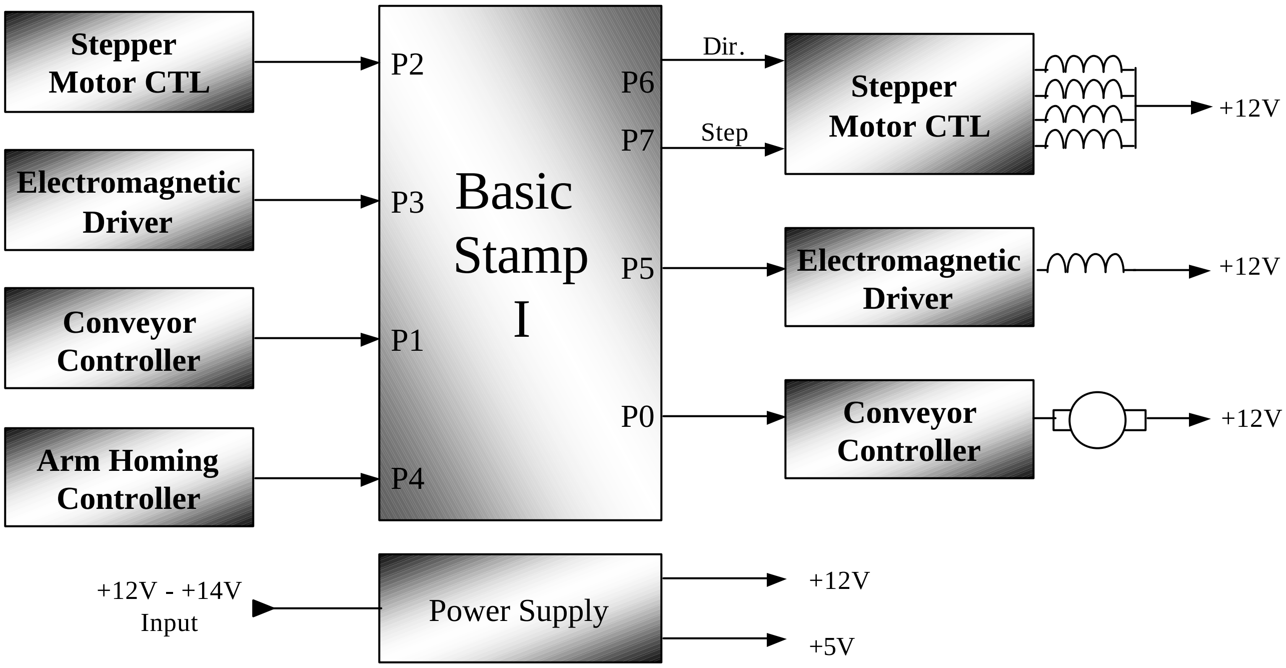
4) Arm Rotates Block to Appropriate Bin



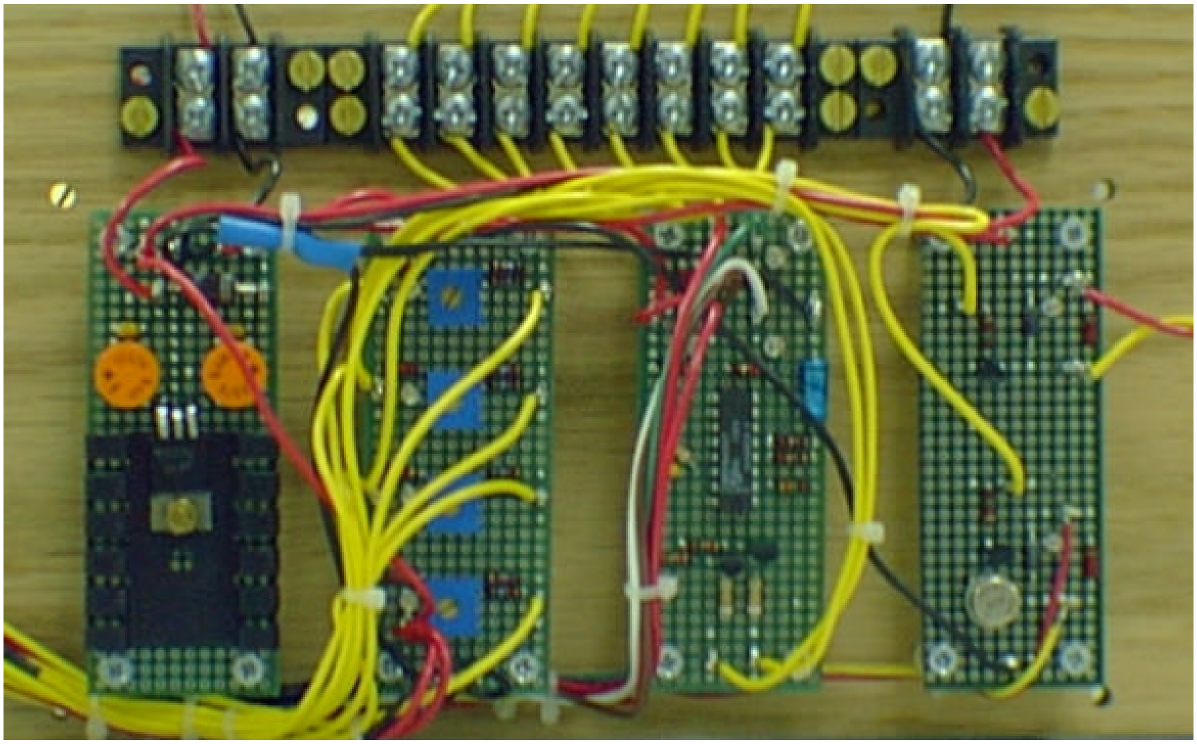
5) Electromagnet is de-energized

# *Basic Stamp I Interface to Individual Control Units*

## Block Diagram

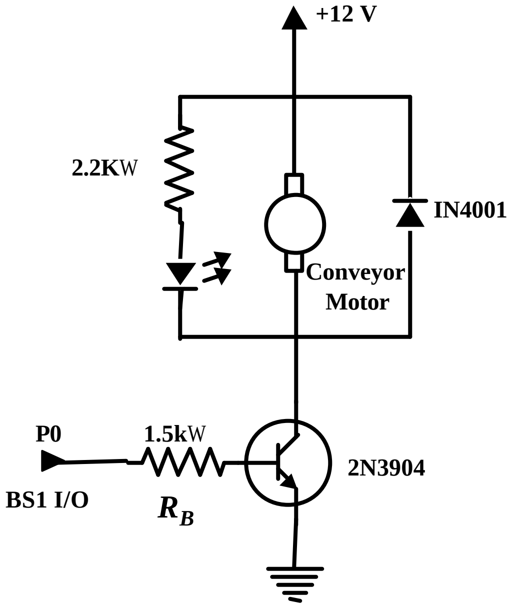


## Circuit Assembly Unit



# Conveyor Control Unit

## Schematic Diagram



*Note*

Conveyor motor draws ~ 50 mA @ 12V. Measurements performed using Fluke Digital Multimeter

*Determining $R_B$*

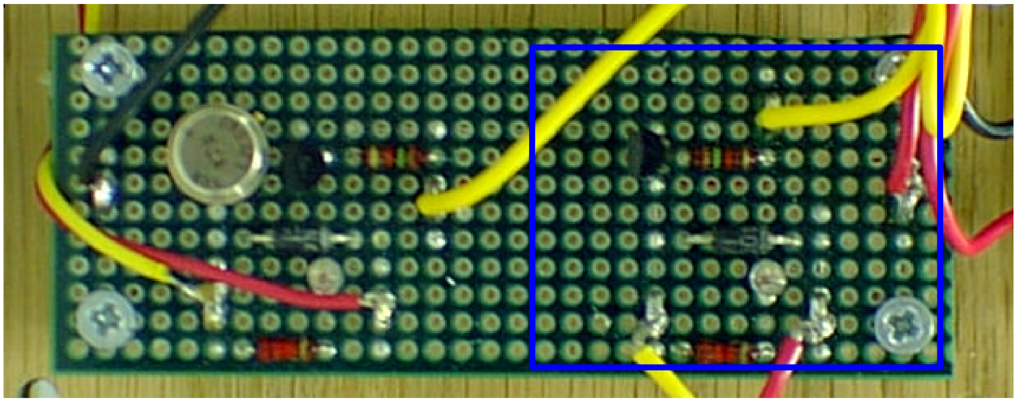
$I_C$ = 55 mA (Assume $h_{Fe} = 20$)

$$I_B = \frac{55}{20} = 2.5\ \text{mA}$$

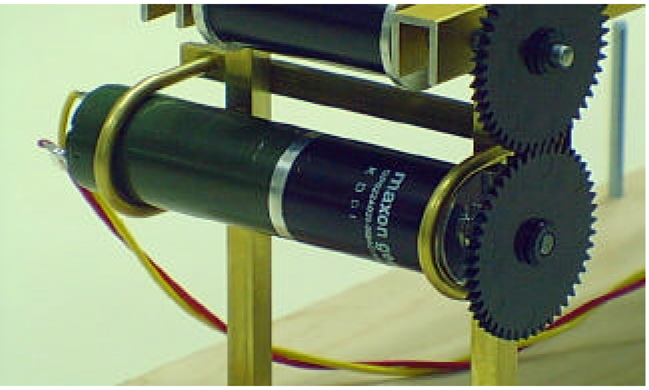
$$I_B = \frac{5 - 0.7}{R_B} = 2.5\ \text{mA}$$

$$R_B = \frac{4.3}{2.5} \text{ x } 10^3$$

## Circuit Assembly Unit



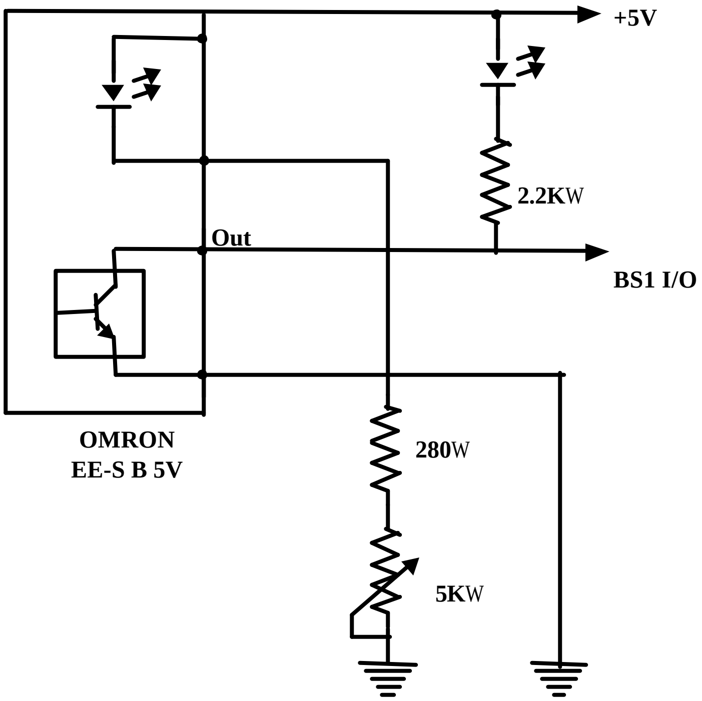
## Conveyor Motor Construction



Maxon DC motor with gearhead. This high-torque, low-speed motor is used to drive the conveyor belt.

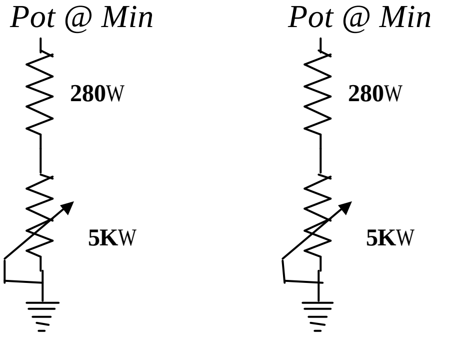
# Wooden Block Sensors and Conveyor Sensor Control Unit

## Schematic Diagram



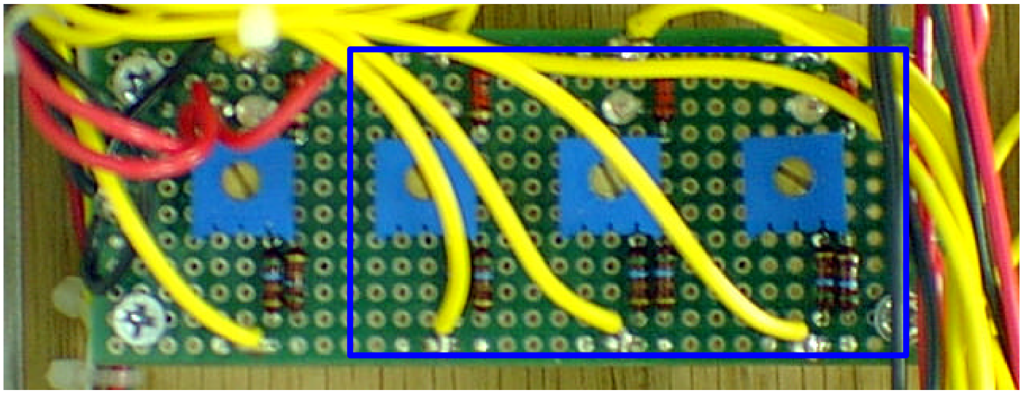
| I/O | Sensor | Function |
|---|---|---|
| P2 | No. 1 | Block Determination |
| P3 | No. 2 | Block Determination |
| P3 | No. 3 | Conveyor Motion |

*Potentiometer Sensitivity*

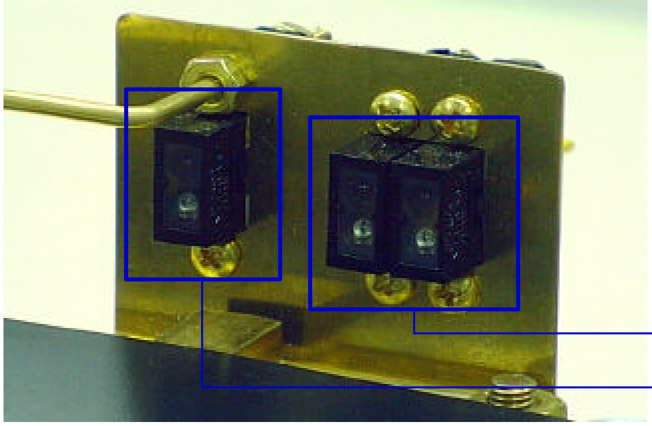


$$I_{LED} = \frac{5 - 1.6}{280} \qquad I_{LED} = \frac{5 - 1.6}{280 + 5K}$$

## Circuit Assembly Unit



## Light Sensors Construction



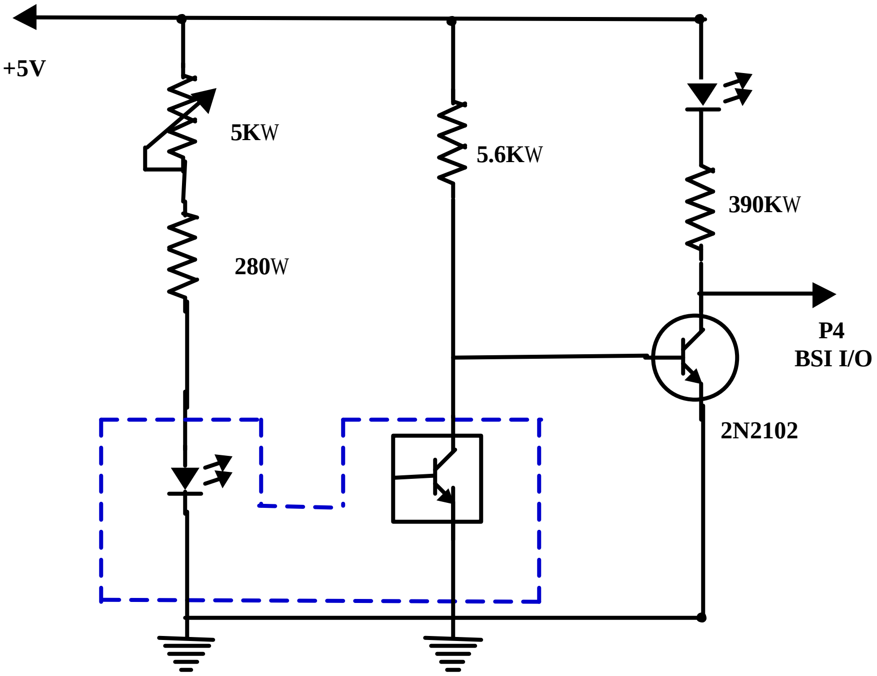
OMRON reflective opto-switches. Emit infrared light and capture reflections. Opaque objects placed near the sensor causes a logic low level. These are used to detect the presence of a block at the end of the conveyor and to read block color codes.

Block Decoder Sensors

Conveyor Motion Sensor

# *Motor Arm Detector Control Unit*

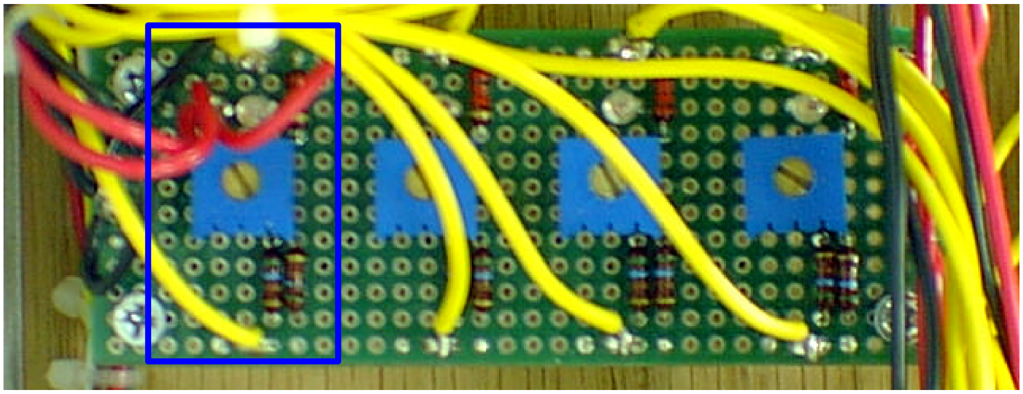
## Schematic Diagram



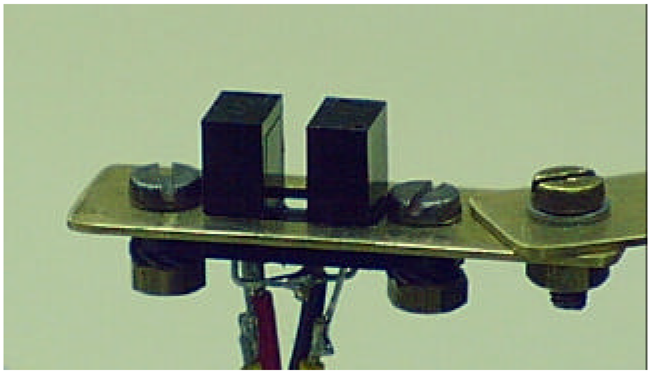
*Sensor Operation*

| *Beam Blocked* | *Beam Present* |
|---|---|
| LED = On | LED = Off |
| Output = Low | Output = High |

## Circuit Assembly Unit



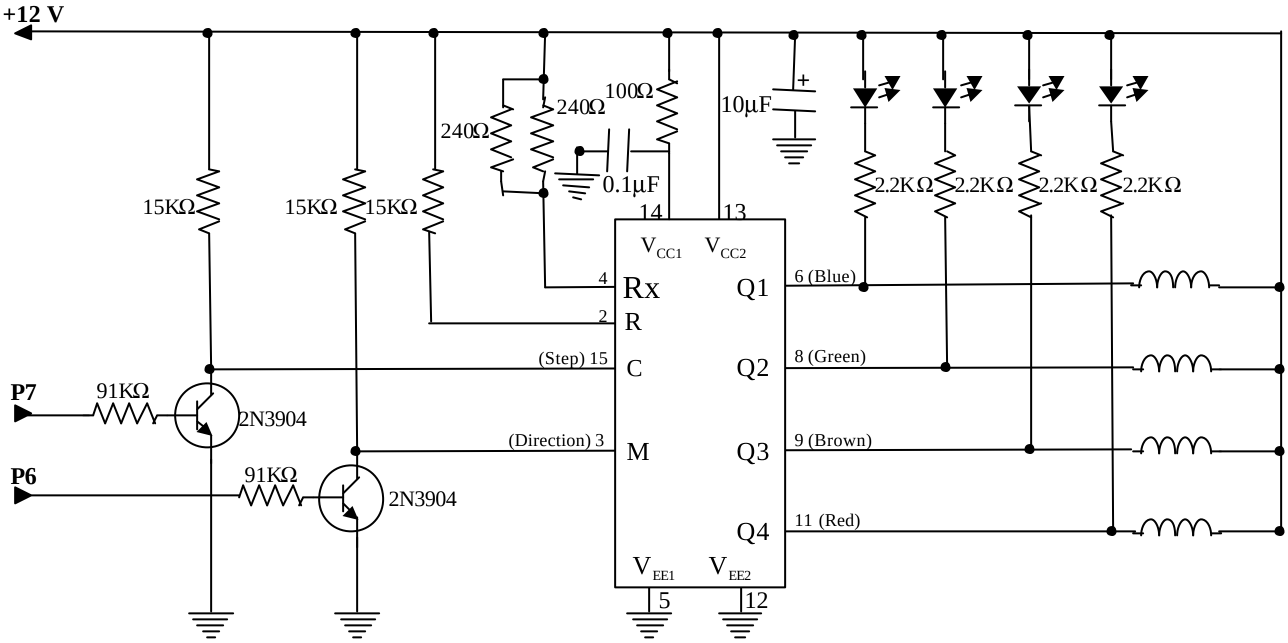
## Photo- Interrupter Construction



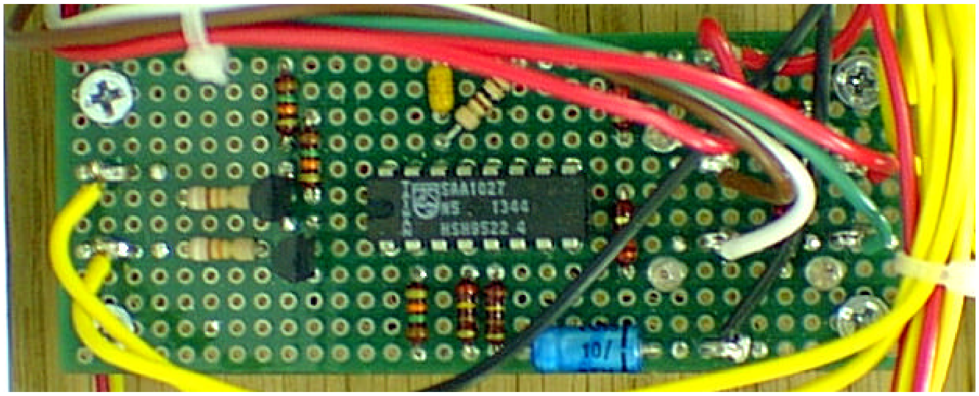
Optek photo-interrupter switch. Emits infrared from one pole and is collected at other pole. Opaque object between poles causes a logic low output. This is used in the sorter system as a means of locating a "home" position for the stepper motor.

# Stepper Motor Control Unit

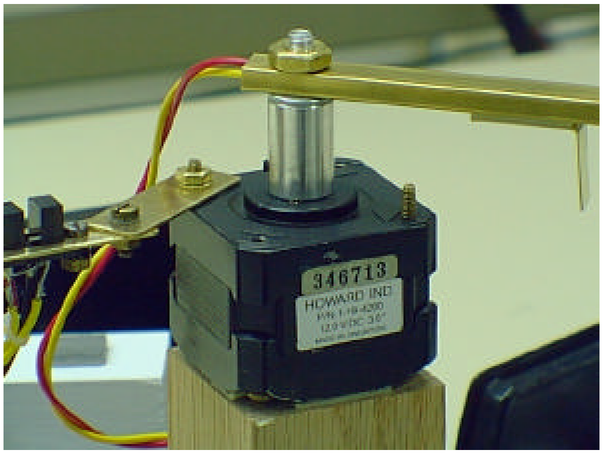
## Schematic Diagram



## Circuit Assembly Unit



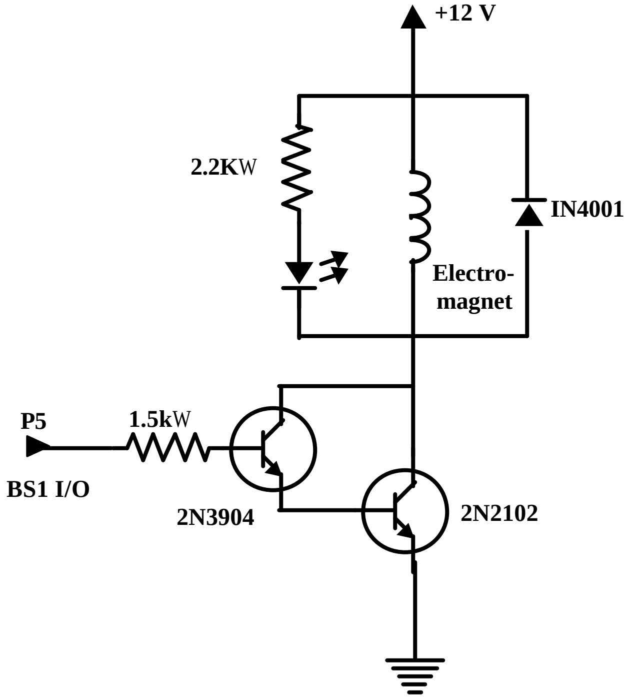
## Motor Arm Construction



Howard Industry stepper motor. Direction identification and pulses sent to the motor from a stepper motor control circuit cause the motor to rotate at a specific increment. The stepper motor is used to rotate the blocks to the appropriate bin.

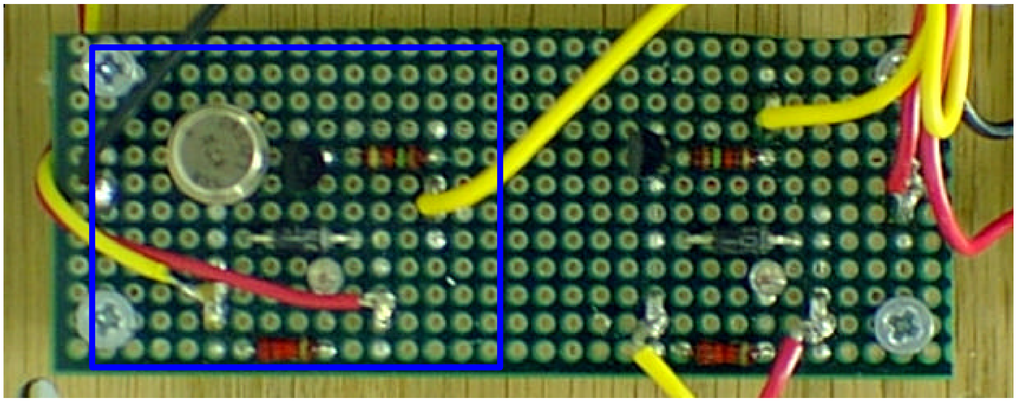
# Electromagnetic Driver

## Schematic Diagram



*Note*
Electromagnet draws ~ 500 mA @ 12V. Measurements performed using Fluke Digital Multimeter

## Circuit Assembly Unit

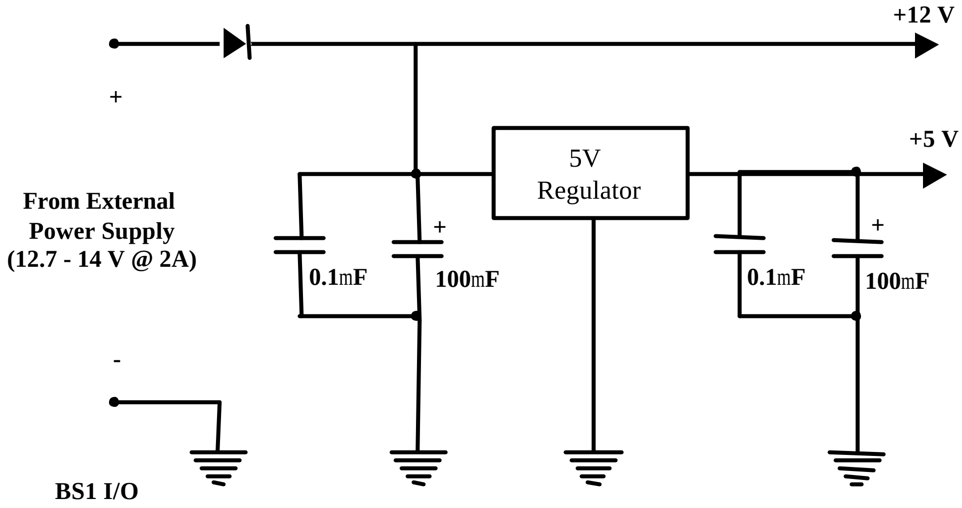


## Electromagnet Construction

Multacc 90-280 Electromagnet. Voltage applied over the coil induces a magnetic field which provides the attractive force necessary to suspend the block.

# *Power Supply*

## Schematic Diagram



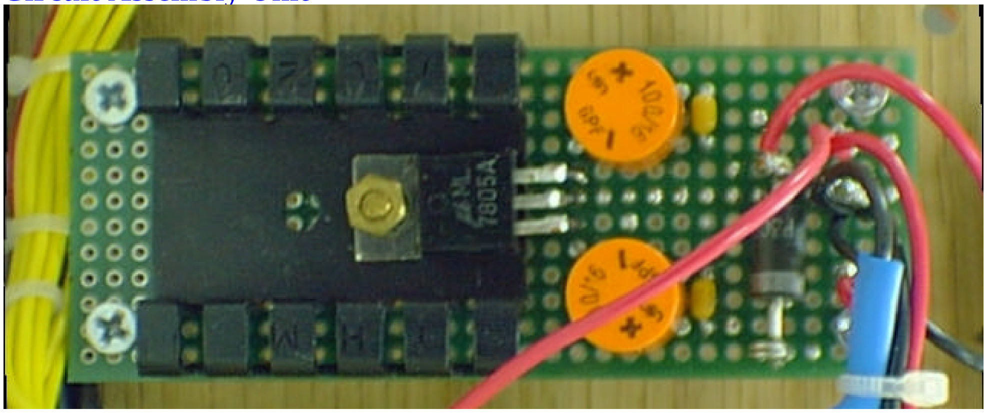
*Note*

100µF Electrolytic capacitor filters out low frequency noise and ripple.

0.1µF Ceramic capacitor filters out high frequency.

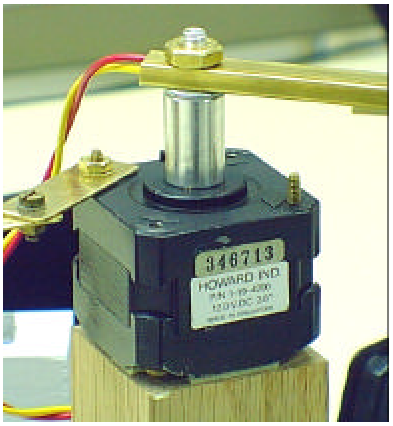
5V Regulator outsourced from Basic Stamp board.

## Circuit Assembly Unit



# Understanding the Stepper Motor

## OVERVIEW



The stepper motor used in the Mechatronics sorter apparatus was salvaged from a former SJSU students project. Since the stepper motor manufacturer was unknown, data sheets could not be obtained that would simplify the process of understanding how to make the motor work. The purpose of this document is to describe the process that was used to determine the stepper motor's type and the purpose of the five wires connected to the motor.

## PROCESS

The only information known was that given on the label and the fact that it had five wires. The label indicated that the motor was 12 VDC and that the rotational step was $3.6^{o}$ (equating to 100 steps per revolution).

*1) Determining the Stepper Motor's "Type"*

Since stepper motors can typically be classified by two types, bipolar or unipolar, the first task was to identify the type. Bipolar stepper motors have two windings and unipolar have four (see diagrams in Figure 2). Since each winding, or coil, acts as an inductor (and since inductors have resistance) an ohmmeter was used to determine that the motor was of the unipolar type. This also determined that the black wire was the ground.

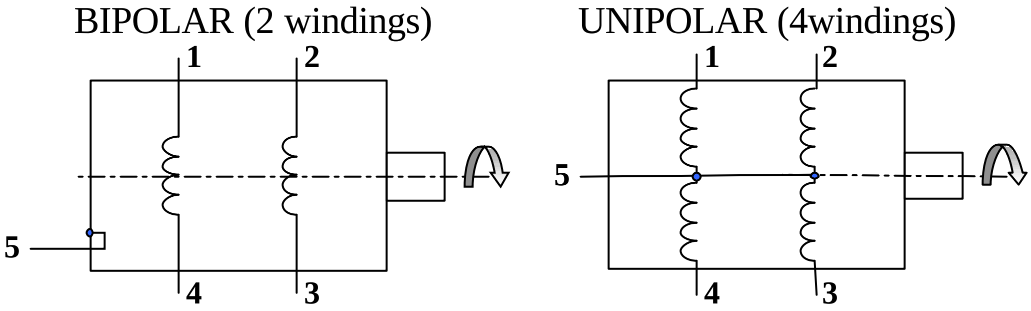


**Figure 2.** Schematic depictions of two types of stepper motors.

A speculative depiction of the stepper motor was sketched and can be seen Figure 3.

# Understanding the Stepper Motor

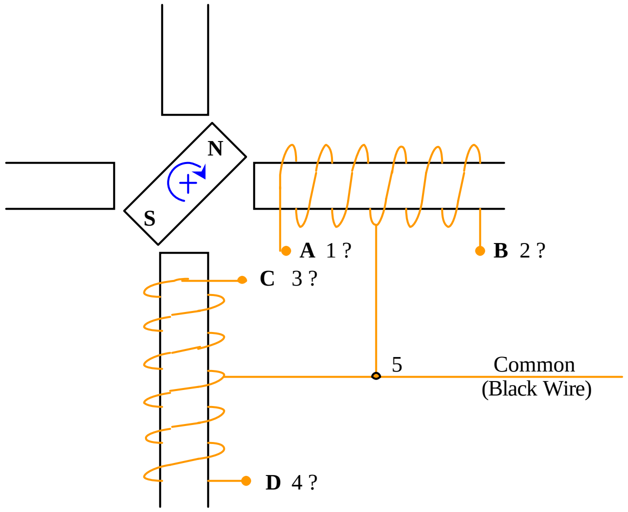


**Figure 3**. Stepper motor model.

Which winding corresponds to which wire? By manually turning the rotor, the stepper motor becomes a generator; current is induced in the coils and wires which can be read by an oscilloscope. The waveform measured between each winding must be in a well defined phase relationship relative to each other. It is known, because of the construction of a unipolar stepper motor, that:

a) A & B are wound on the same leg of the stator, and their outputs must be in phase opposition

b) C & D are wound on the same leg of the stator, and their outputs must be in phase opposition

c) A and/or B will be 90$^{o}$ or 270$^{o}$ out of phase with C and/or D.

Sketches were drawn of the induced waveforms for the different wires with ground. For example, Channel 1 of the oscilloscope  was connected to the green wire (hot) and black wire (ground). Rotating the stepper motor produced a somewhat sinusoidal waveform as seen in Figure 4 (see Green-Black). This waveform was treated as the phase reference. Performing this test for each wire allowed the determination of the wire correspondence. Plots produces for each wire are provided in Figure 4.

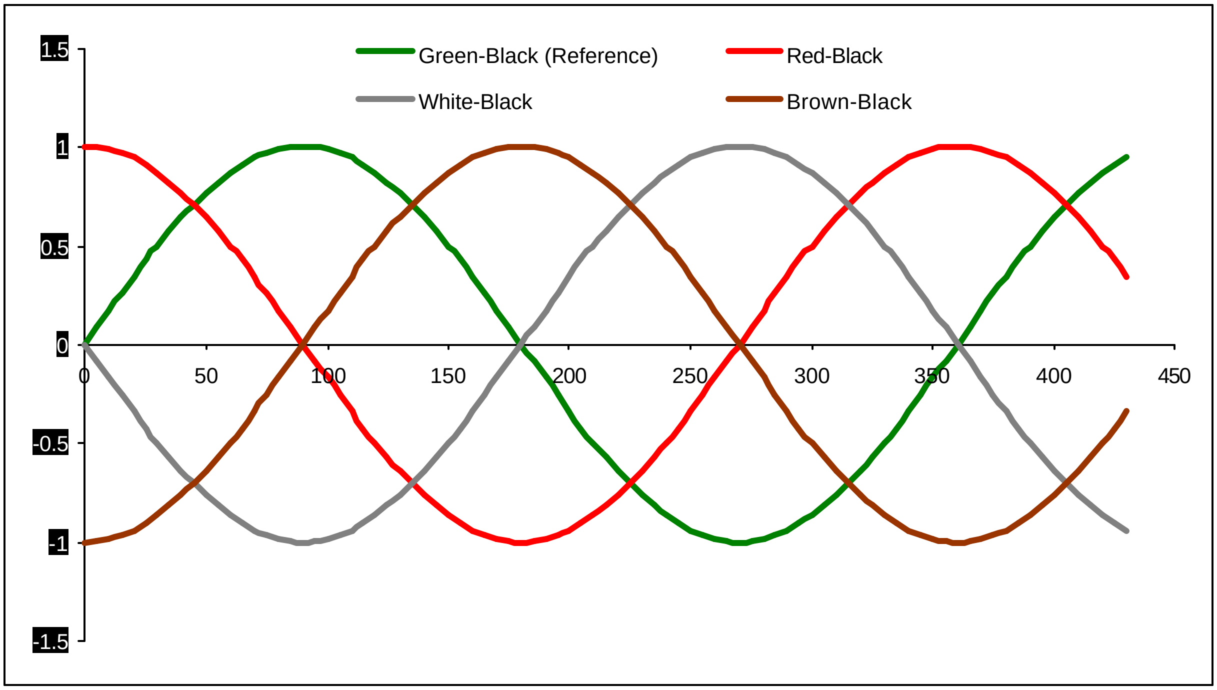
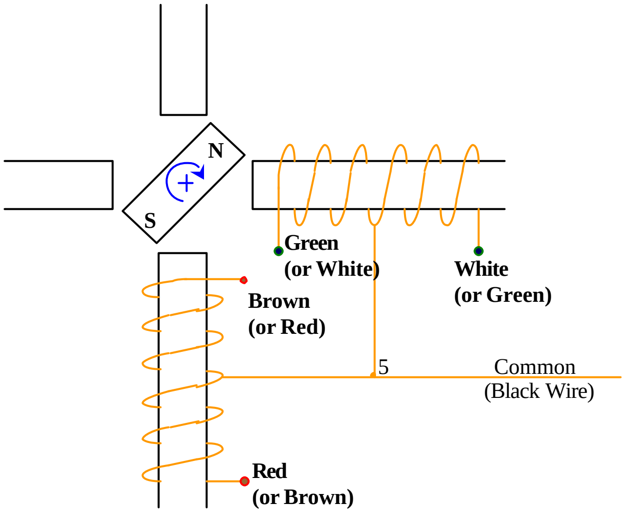


**Figure 4**. Waveforms found for each wire.

It is obvious that Green-Black and White-Black are exactly $180^{o}$ out of of phase, therefore, they must be found on the same leg. For similar reasons, Brown-Black and Red-black must be on the same leg. We can thus conclude that the motor looks as follows:



Having established which winding is which, we can safely match it up with the stepper motor IC.