UNIVERSITY OF NORTH TEXAS
DEPARTMENT OF ELECTRICAL ENGINEERING

Industrial Control Unit

Senior Design

Nicholas Tompkins
Justin Peterson
Daniel Mata
Jian Xu

Summer 2014

Table of Contents

I. Abstract

In today's modern industrial environments the need for advanced embedded control systems and real time monitoring has never been higher. This push for more advanced systems is recognized as a United States government initiative known as Smart Manufacturing (ref1). To stay competitive, we as engineers must seek to produce extended value from combining systems to play multiple roles within a manufacturing environment. We created such a system that draws multiple manufacturing and automation needs while directly interacting with existing efficiency driving techniques such as Lean Sigma. What we have produced is called an Industrial Control Unit (ICU).

The premise of our project is to join a system of operations to control a Programmable Logic Controller (PLC) and merge the features of some of the basic embedded systems and sensors to create a hybrid controller unit that can meet the needs for a factory floor. In this process we also address the evolution of the Internet of Things in terms of the growing demand for these smart devices. Once we describe the system we developed and the areas of strengths and weaknesses, we will further describe the path that we would need to take to promote this product to the next level of development.

II. Introduction

Along with this project we were given the opportunity to design a possible solution with our designed system within a local company. Labinal Power Systems is in need of a possible means to track Kanban totes throughout the factory floor. This would enable the use of "just-in-time" Lean Systems to be implemented in real-time due to a possible means of integrating our system within Labinal's Enterprise Resource Planning (ERP). Our device can do much more than RFID tracking and speaking to a database. It handles this task very well but can also take on tasks such as automation control.

III. Labinal Power Systems

Labinal Power Systems is an aircraft wiring harness manufacturer. They manufacture over 500 hand built wiring harnesses out of their Denton facility each day. The process to create these handmade harnesses is relatively slow and tedious and leaves much room for human error and also creates many opportunities to increase efficiency. One area that they would have great benefit from would be the ability to track Kanbans or totes of products that travel from station to station once an order hits the production line. This not only would generate even more detail in reporting of the flow of the work areas but it would generate the ability to see where critical orders currently are and would generate a realistic delivery date.

To generate a solution for their proposed problem they would want the ability to track totes in real-time in order to log times for events such as a Kanban changing workstations and to see how a Kanban moves through the company from start to finish.

The first thing that we thought of was the use of Radio Frequency Identification Device (RFID) tags to flag stations with timestamps of these orders through a system. This is where we begin to see an

interesting problem within the current control system designs. SPI interfaces are rare on control system devices and the only consistent solution that we had seen to merge these devices together is a RS232 port. This requires utilizing more external hardware interfacing and sacrifices the ability to have a more internal hardware control. After thinking the solution through, we decided to move forward with this problem in mind and developed an all-inclusive Industrial Control Unit that had most industrial control applications in mind.

IV. The Concept of our ICU

We chose to develop our platform based on what we considered to be the needs of an Industrial Control System. To stand apart from existing systems we needed the flexibility of a Microcontroller (MCU) to allow us to interface with sensors that are not standard to control systems. We needed to design the analog circuitry to handle the harsh Input/output (IO) of industrial environments.

Our basic ICU requirements include the following:

- Speed: A system that can handle Ethernet communications and still have enough speed to control multiple motors in near parallel
- Internet: A user friendly web-based interface accessible through a network.
- SPI interfacing: The use of SPI as a bus for sensors.
- Size: Standardizing the size of our ICU to remain comparable to that of an industrial PLC
- Support: We wanted to select a MCU from a company that could provide quick and easy customer support.

After considering these criteria, we decided to use the Tiva C Series TM4C1294 Connected Launchpad Evaluation Board from Texas Instruments (TI) since TI is an industry leader in support of their MCU customer base. This Launchpad gives us the ability to connect the IO more easily with the outside world due to the onboard Ethernet control.

The Launchpad offers additional features such as:

- 120MHz 32-bit ARM Cortex-M4 CPU with floating point
- Integrated 10/100 Ethernet MAC+PHY
- 8x 32-bit timers
- Dual 12-bit 2MSPS ADCs
- Motion control PWMs

Our ICU design must have normal 3.3V digital pins to drive signals as well as the ability to use Pulse Width Modulation (PWM). It must also have 24V IO pins separated via optical isolation to keep the controller safe. We also need some intuitive interfaces that will generate the results required of the system. One of these interfaces is located directly on the ICU as a top layer PCB with buttons, indicator lights, and an LCD screen. Our second interface is a website accessible through a network which caters to those that are off site or not at the physical station.

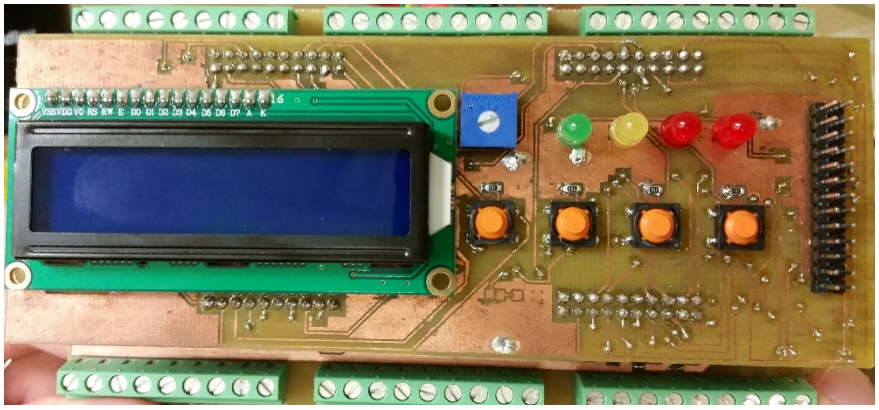

Figure 1: The top PCB of the ICU.

V. Standards in Industrial Control

A. PLCs and PACs

The best definition of a PLC or PAC is as defined by Texas Instruments:

> *"Programmable Logic Controllers (PLC) and Programmable Automation Controllers (PAC) are process and control implementations that cover everything from test labs and fabrication plants to military and medical electronics to basic data acquisition. They leverage various sensor types and feedback mechanisms to monitor and control the local environment and system/machine interactions by collecting, storing and analyzing data. Acquiring data from sensors involves precision measurement and the processing of very low values or small changes in analog voltages/currents." (Texas Instruments) (ref2)*

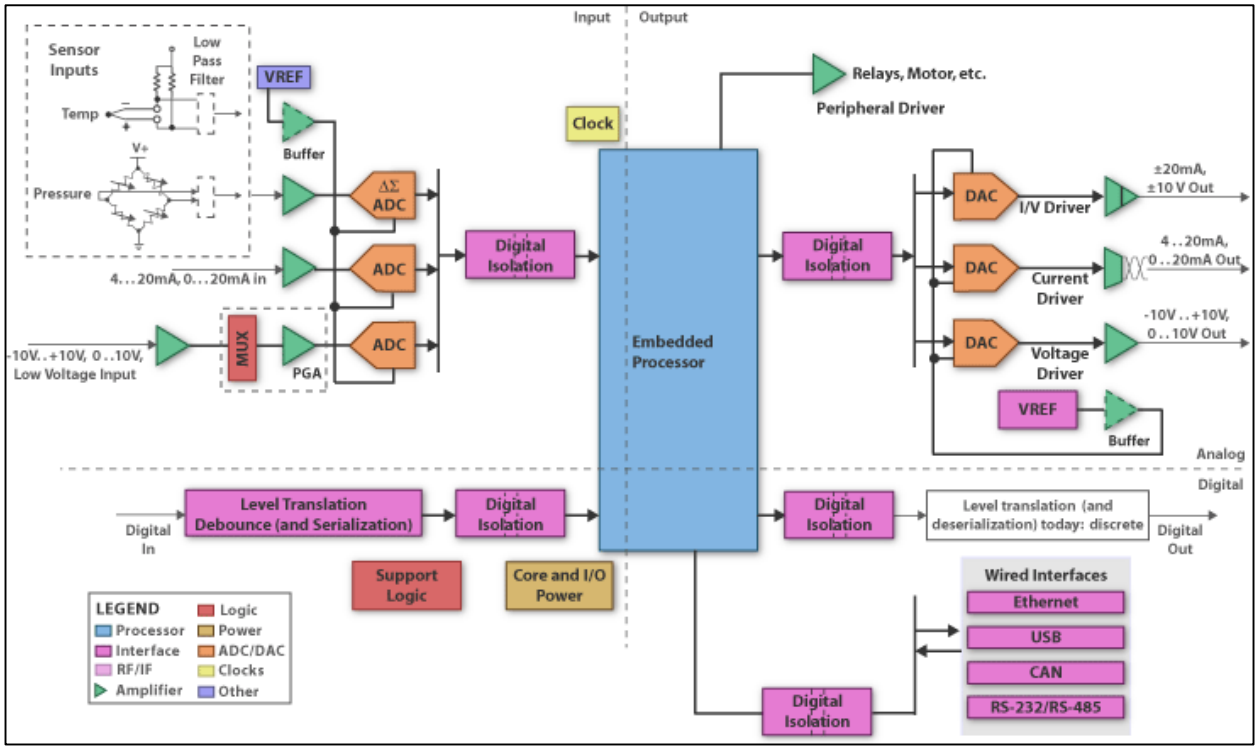

Figure 2: Block diagram of an industrial PLC from Texas Instruments.

B. IEC 61131

The International Electrotechnical Commission (IEC) creates standards for PLCs and PACs. These standards are 464 pages of very complex and involved restrictions that can only be received through purchasing a published technical report for $340. Since we neither had the money nor the time to read, understand, and consider these standards in this 10 week course, we acknowledged that our ICU will not meet industry standards and we can only consider it to be a prototype.

> *"These standards include the definition of the Sequential Function Chart (SFC) language, used to structure the internal organization of a program, and four inter-operable programming languages: Instruction List (IL), Ladder Diagram (LD), Function Block Diagram (FBD) and Structured Text (ST). Via decomposition into logical elements, modularization and modern software techniques, each program is structured, increasing its re-usability, reducing errors and increasing programming and user efficiency." (IEC 61131-3 ed3.0) (ref3)*

We have not yet developed a proper language to have this standard fully inducted due to the lack of robust libraries on the MCU for consumer use. This is why we decided to call our unit an Industrial Control Unit (ICU) instead of a PLC to set it apart from the standards associated with the PLC name. We can still

implement any algorithm required of a PLC, but we are lacking the feature rich programming languages of that of a major PLC developer.

VI. ICU Design

We divided the design of the ICU into four areas of focus. Normally each person would be assigned an area, but since each team member is proficient in most of these areas, we all worked along-side each other in all areas. We relied upon *Open Electronics OSPLC (ref6)* for reference of our 24V I/O design with some modification to the resistances. We divided the project into the following sections:

- Circuits and Hardware
- PCB Layout and Fabrication
- Embedded Coding

A. Circuits and Hardware

Optical Isolation

Optical Isolation is a feature that allows us to protect our circuitry from voltage differences. It is a switch that uses light to open and close a circuit from a remote input. The remote input circuit is completely separated from the circuit being open and closed. There is no physical connection that will transfer current between these circuits. This allows us to protect our high voltage circuits from the low voltage power in the Launchpad.

We use optical isolation in our power rails to separate the remote function that the Launchpad provides and the power that is generated through the power supply. This allows us to activate a 24V circuit to a slot on the power rail by closing the circuit through a signal which is sent from the Launchpad to the optical isolator.

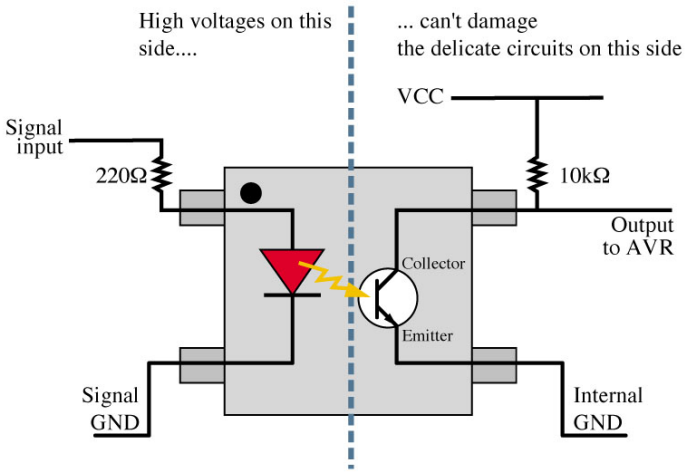

Figure 3: Optocoupler

Electro-Mechanical Relay vs. Solid State Relay

Electro-Mechanical Relays (EMR) are used in many applications where we need a small signal to switch the state of a device that is on a different voltage level. We chose to use the EMR because it has a versatile feature which allows it to handle both AC and DC within its tolerance range. Uses of the EMR are unlimited in the scope of switching the signals or current sources of devices. Another major feature is that we do not need to worry about the voltage variation between the devices that we would control with the EMR. Unfortunately, the relays get physically larger to handle higher currents.

A Solid State Relay (SSR) is a device that utilizes the features of optical isolation which encouraged us to use this unit. This means that it allows us to have digital control of an AC source. This is very important in the control of AC motors that are used in many industrial settings. The SSR has a higher current tolerance of 8 Amps while remaining in a smaller package, which makes it the better choice for our design.

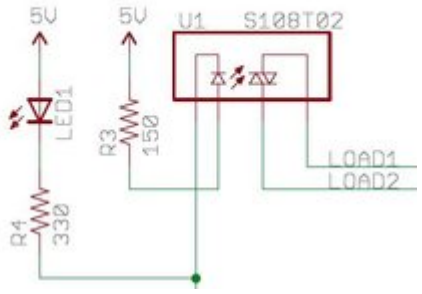

Figure 4: Solid state relay

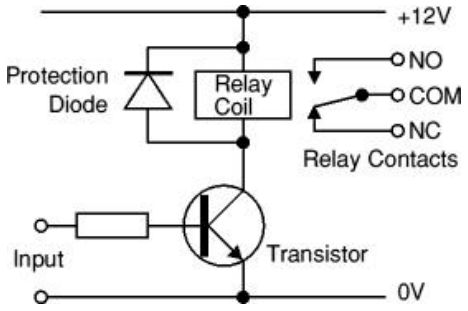

Figure 5: Electro-Mechanical relay

Digital Input Circuit

In this circuit we utilize a switching diode and zener diode duo. This enables us to have a threshold of what we consider a HIGH in the signal logic as a threshold of at least 8.1V. This gives us a range that we can consider when receiving inputs from actuators or digital industrial sensors. The optical isolator will then drive its output to the appropriate voltage 3.3 or 0 contingent on the input signal.

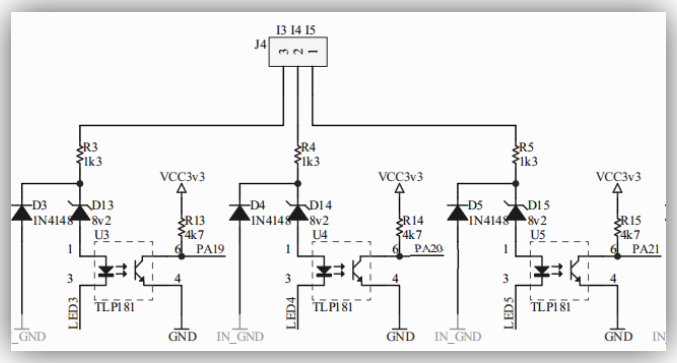

Figure 6: Circuit diagram of 24V input.

Digital Output Circuit

Utilizing optical isolation we are driving the gate of the transistor that is allowing the ground to have a connection. In doing this we are able to control the output of our supply to the actuator utilizing the diode to hold the current back making a large voltage drop appear from the output ground and the output supply. These are used to drive devices but once again we have a limit of around 1 amp that can be used from this output.

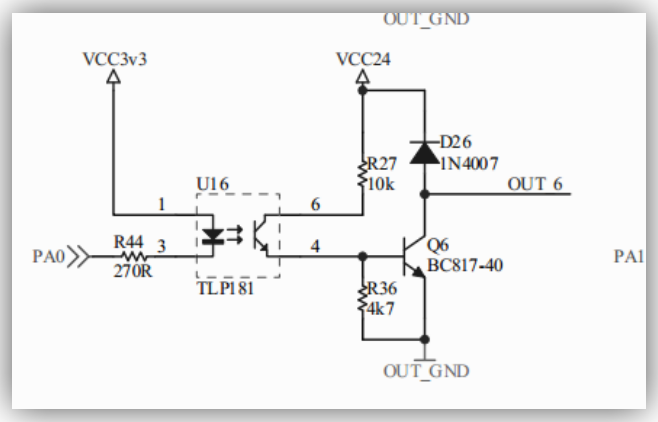

Figure 7: Circuit diagram of the 24V output.

B. PCB Layout and Fabrication

We built two PCBs per unit in order to equip our microcontroller with the additional power and interface options. These circuit boards are paramount in the design of our ICU and each PCB had to be customized to fit our design. We used a program called Eagle PCB to design a digital copy of our PCB. This program allows us to map out our design and also assists us in utilizing our space in the best way possible. The size of our PCB is dictated by the size of our microcontroller and the case we put it in.

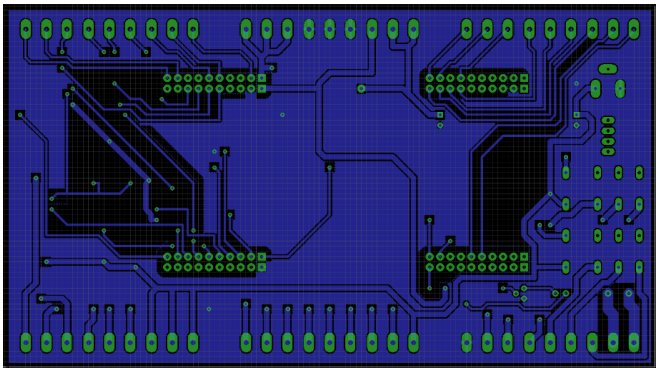

Figure 8: Bottom PCB virtual schematic example.

There are a number of options when deciding how to create a PCB. One option we considered was etching. This process uses a machine to etch the design out of the PCB through an automated process. We chose to follow a different method that takes longer, but is cheaper and more personable. Our method involves using a laser printer and a chemical to develop each PCB. The PCB design process is lengthy and takes a considerable amount of time for each PCB, but it is a more involved method that we preferred and is more visually appealing upon completion.

PCB Design Process:

1. Design digital copy of our PCB with size taken into consideration.
2. Measure the enclosure size and make sure the design fits.
3. Use a laser printer to print the digital copy of the design onto magazine paper.
4. Iron the magazine paper (ink side down) onto the PCB to transfer the toner to the copper.
5. Soak the PCB In water and brush off any paper residue.
6. Check ink lines and make touch ups as needed using a fine-tipped permanent marker.
7. Soak the PCB in a corrosive chemical which dissolves any copper not covered with ink.
8. Rinse PCB and check connectivity of each transmission line on both sides.
9. Solder components onto the PCB.
10. Check for connectivity before plugging into power supply for testing.

Bottom PCB

The bottom PCB contains the circuitry for our power rails. This circuitry includes opto-couplers for optical isolation to protect our microcontroller. We organize 6 sets of 9-output terminal blocks. Each of these terminal blocks serves a general purpose and have been organize for easy usage.

One terminal block contains 9 PWM/GPIO pins and a second block contains 9 regular GPIO pins. A third terminal block contains an organized array of all different power sources including 3.3V and 5V directly from the Launchpad, 5V (3.5A) from the power regulator and ground pins. A fourth terminal block contains six opto-coupler controlled outputs and three 24V pins. A fifth block contains solid state and mechanical relays for input and outputs. The sixth rail contains four 10 to 24V inputs. An image of the pin map can be found in the Appendix.

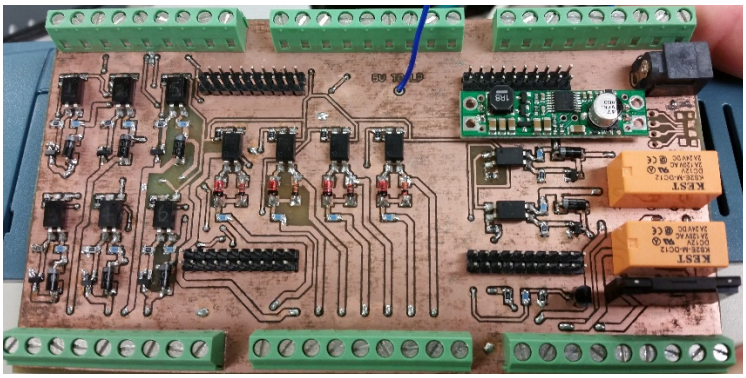

Figure 9: Actual image of bottom PCB fully assembled.

Top PCB

All of the GPIO pins as well as SPI, UART, and I2C are also connected to the top PCB because it is easier for shield extension and wiring. This PCB includes a LCD screen, some buttons and LEDs, and a potentiometer. This top PCB is created mainly for the user to interface directly with the ICU if they wish to do so. Each component is Auxiliary and can be customized to perform tasks on the UCI itself. For example, a garage door could be connected and the button on the unit could control the opening and closing of the door. We believe this is a much-needed feature since we want our design to be open source and customizable.

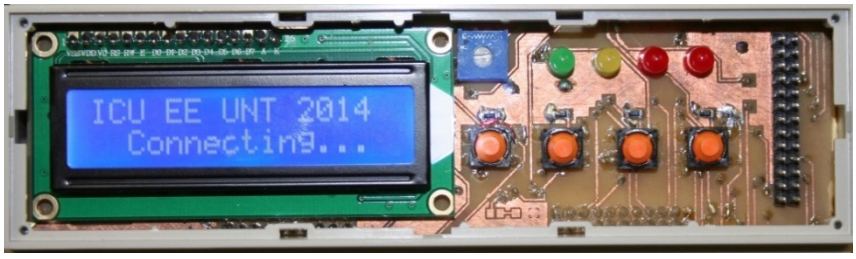

Figure 10: Actual image of top PCB completed and installed.

C. Embedded Code

We used a program called Energia as the platform for controlling the Launchpad microcontroller. Any user can use any code platform they prefer in coding the device as long as it contains the libraries to match the Launchpad we chose. We simply chose Energia because it already contained libraries and we were familiar with the coding language that it requires us to use, which is C based.

We were able to perform tasks through Energia such as embedded webserver to SQL integration, user interfaces, and motor control. We designed a number of scripts to assist us in testing and demonstrating the capabilities of our device to include a script to control servos on a robotic arm and another script for general input and output processing of sensors.

VII. Demonstration and Testing

As we have gone through the development process we felt the need to show our unit in action in a number of different examples to show its flexibility. We will demonstrate this by the Labinal RFID system, Robotic Arm, Stepper Motor Control and a Basic User I/O interface. Below is a breakdown of each of these mini-projects we developed to demonstrate the capabilities of our device.

A. Labinal RFID System

We were able to come up with a solution to Labinal's problem, explained earlier in this report, by creating a realistic representation using RFID scanners, a server, and a database. This demonstration encompasses the input/output, processing, and networking capabilities of our ICU and serves as a very thorough example of how our product is usable in solving industrial problems.

RFID scanners use radio frequencies to scan magnetic barcodes that come near them. We connected four of these scanners to an SPI bus on the Launchpad. The term "bus" means that these RFID scanners are sharing the same Master-In-Slave-Out (MISO) and Master-Out-Slave-In (MOSI) data lines with independent Slave Select pins and reset pins. This design does not allow for simultaneous reading in the literal sense of the word, but can imitate near simultaneous reading of each RFID scanner by alternating between them at incredibly fast rates. These speeds are so fast that the human eye cannot tell the difference. This processing speed decreases with the addition of more RFID scanners to the same Launchpad, though.

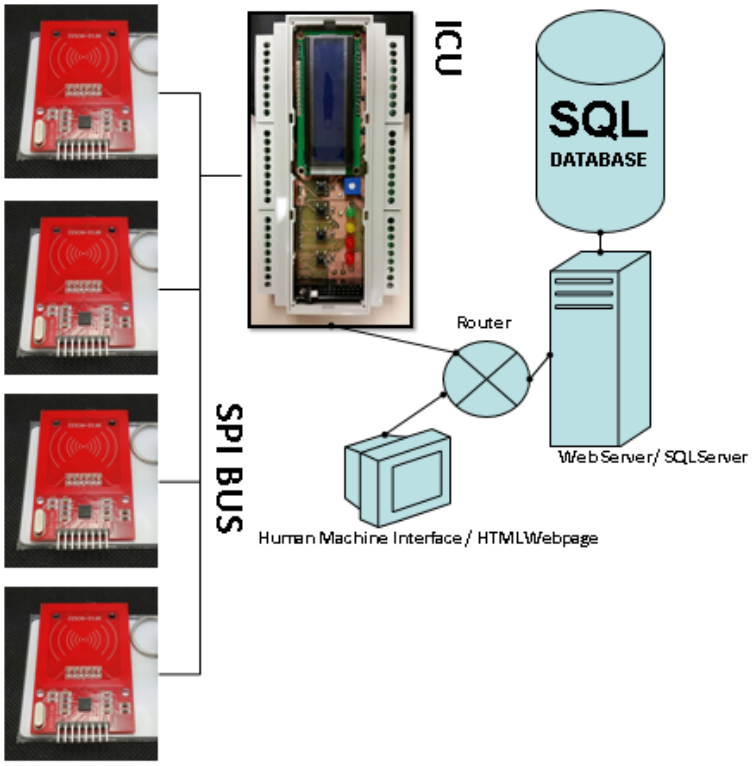

Figure 11: RFID set-up diagram

With the RFID scanners connected to the Launchpad, and the Launchpad reading the inputs, we needed a way to store the data and create an interface to view the data to replicate a corporate design. We chose to use SQL Server Database Express 2012 to store the data in real-time coupled with a combination of HTML and PHP scripts to handle the data transfer between the Launchpad and the Database as well as provide a friendly user interface.

Below is a breakdown of this demonstration. Each section discussed will cover SQL Server Database Express 2012, PHP scripts, and HTML interfacing. The code for each of these is located in at the end of this report.

1. SQL Server Database Express 2012

The database contains tables called Tracking, Marry, and Location all with a prefix of "lab" which represents the schema (or group, if you will) that they belong to. The Marry table contains an unchanging column of RFID numbers that represent the electronic barcodes for the totes. These numbers are unchanging because the company would have a certain number of totes, each with their own unique identity. The changing portion of the Marry table is the column of UPC barcodes which will be assigned to a tote upon a particular project being taken on in the company. This means that when a new harness is ordered, a tote is tasked with containing that harness, and the order number of the harness corresponds to the UPC. Now the order number is associated with that particular tote.

The Location table contains an unchanging list of all of the existing locations of RFID scanners within the company. This location table, as well as the RFID column in the Marry table, is tagged with a specific key that labels them as reference columns. The Tracking table contains the records of each tote moving through the company, as well as the data, time, location, and UPC number associated with that tote. The reference keys mean that any information that is sent to the Tracking table *must* have an existing entry in the location and marry tables. This is a preventative measure to make sure false information is not recorded in the database.

2. HTML Interface and PHP scripts

Information from the RFID scanners is processed by the Launchpad through an Energia script, and is sent to a PHP script on the web which forwards the information into the database to be recorded. This PHP script is necessary since the Launchpad cannot directly speak to the database. There are other methods to achieve this goal but we used this method since it was easier for us to understand and create a solution from in a short amount of time.

Our HTML interface includes a webpage which allows a user to scan an RFID and UPC using a barcode scanner and select a "submit" button. Upon pressing submit, the database uses the RFID to find the corresponding row in the Marry table and pastes the scanned UPC number into the second column next to that RFID. A second page allows the user to do the opposite, by scanning the RFID and UPC tags on a tote containing a finished product and removing the UPC number from the database in the Marry column. This leaves that RFID number un-associated with a UPC, which means that tote is now empty and can be assigned a new order number.

The third page on the webpage allows the user to search the tracking table in the database and pull up portions of the tables located in the database based on criteria. A

user can input an RFID number, UPC number, Location, Date range, or Time and search for any entries matching the criteria. This part was tricky, since the PHP script handles all the information hand-offs on the web-based side.

In order for the PHP script to forward information to the database, we had to structure a query (script, in database terms) within the PHP script that it would prompt the database to execute. We designed a code that would dynamically build this query based which criterion were entered in to the search boxes. This minimized the length of code and would account for any possible combination of search parameters that a user could offer.

B. Robotic Arm using Servos

We set up a robotic arm and connected 8 servos to allow for automated functionality. We also set up an HTML interface and connected it to the web so that we could control the motions of this robotic arm through any computer device connected to the network. In essence, we wanted to show that our ICU could host a website on a network while processing scripts to control the speed and range of motion of each servo simultaneously.

1. Physical Design

We used 4 large and 4 small servos and assigned each servo to a PWM pin from the Launchpad. Each servo shares the same 5V, 3.5A output from the Launchpad, but each has its own individual reset and signal pin. We had the robotic arm 3D printed based off a design we found from the web. Each servo has a max range of motion from 0 to 180 degrees, so we decided to only allow them to function within 10 to 170 degrees to prevent damage to the servos.

2. HTML Interface

This particular design has limitations due to the functional boundaries of the Energia software that we used for our Launchpad. Unfortunately, we needed to host our HTML/PHP scripts within Energia, which limited the functionality of what we could do in the HTML and PHP scripts. For example, Energia is incapable of understanding form requests written in HTML, which means we are unable to modify our webpage to include Push-and-Hold features on the buttons to allow dynamic movement based on the duration a button is pressed.

Instead, we have two buttons for each servo that allow clockwise and counter-clockwise motion for each servo. Upon pressing a button, the servo will move a predetermined number of degrees in the corresponding direction. We included three buttons to change these increments between 5, 10, and 15 degrees.

3. Servo Control Code

We also made changes to the servo's movement. Originally, if a servo was being told to stay at 10 degrees and we sent it a signal to change to 50 degrees, it would whip around

at 100% of its capable velocity to reach 50 degrees where it would stop and maintain that position. This caused our robotic arm to be very twitchy and abrupt. We implemented “For” loops with dynamic parameters in our code that would slow the movement of each servo down by injecting a delay signal in between each degree of movement.

C. Additional Demonstrations

In addition to the RFID system and the Robotic arm, we have connected a few motors and lights to our ICU to further demonstrate its capabilities. We can connect an A/C fan which we can switch on and off through a web interface if so desired. We also have a stepper motor we connected that shows a more industrial sized application in regards to voltage and power draw from a motor. These are just a few of countless electrical appliances that can be controlled through our ICU.

VIII. Ethical and Professional Issues

Some Issues that we will encounter if we were to advance this product past the prototyping stage would be that of insufficient safety certifications. When we decided not to try and adopt the name of PLC or PAC, we were in turn shielding ourselves from the strong scrutiny that comes with such products. It would be rather unethical to say that this system can perfectly mirror the quality of those products. In a professional sense we have left no opportunity for us to honestly go to market with such a device and this has be our intention from the start. We feel that in a professional environment some layer of open source hardware and software drives innovation and creates a larger pool of experience than that of a small company could afford. This is a very unique double edge sword that might bless or tarnish industries for years to come and in itself could be a whole topic in business that could span beyond the scope of any given project.

IX. Contemporary Issues

The most major issue that plagues the world of PLC’s and PAC’s is that of hacking. These devices drive safety equipment our power grinds the list is ad infinitum. In this process we fail to realize encryption due to time constraints but the threat is very real. SQL injection hacking the devices registers, imagine having your assembly plant down for a week when the competitors is up and going. The corporate environment is all about supply and demand and first to market. These issues are alive and well and we can imagine that a selling point for PLCs or PACs is that of robust encryption. This plays a larger factor into the evolutions of Smart Manufacturing as our systems get smarter and more connected they are more and more prone to infiltration.

X. Engineering Standards

The SIL requirements for hardware safety integrity are based on a probabilistic analysis of the device.

- SIL-3 RATING – Power Safety

PFD (Probability of Failure on Demand) and RRF (Risk Reduction Factor) of low demand operations

SIL	PFD	PFD (power)	RRF
1	0.1-0.01	10−1 - 10−2	10-100
2	0.01-0.001	10−2 - 10−3	100-1000
3	0.001-0.0001	10−3 - 10−4	1000-10,000
4	0.0001-0.00001	10−4 - 10−5	10,000-100,000 tion for different SILs

- IEC 6113-3 – Defines a PLC
- $340 Document that describes that a PLC should be
- IEEE 802.3 – Ethernet
- IEEE 802.15 – RFID
- Rose Helman Writing Standards

XI. Realistic Constraints

Our realistic constraints include the following:

- Speed: A system that can handle Ethernet communications and still have enough speed to control multiple motors in near parallel
- Internet: A user friendly web-based interface accessible through a network.
- SPI interfacing: The use of SPI as a bus for sensors.
- Size: Standardizing the size of our ICU to remain comparable to that of an industrial PLC
- Support: We wanted to select a MCU from a company that could provide quick and easy customer support.

XII. Conclusion

Overall, we sought out to develop an open source industrial control unit and we have been successful. We have learned how to pivot in the development phase and to utilize our group members' strengths in order to get the prototype done within our allotted time. Some of our members learned about surface mount PCB development while other learned PHP and SQL, which goes to say that as electrical engineers there is very little that we cannot learn to accomplish the prototype. We found immense value in creating something from our foundation of knowledge and we are very grateful for the chance to show the workforce our capabilities from going through this process.

XIII. References

(ref1) http://smartmanufacturing.com/what/

(ref2) http://www.ti.com/solution/programmable-logic-controller-diagram

(ref3) http://www.plcopen.org/pages/tc1_standards/iec61131-8/

(ref4) http://www.futureelectronics.com/en/optoelectronics/optocouplers.aspx

(ref5) http://www.plcopen.org/pages/tc1_standards/iec61131-8/

(ref6) http://startingelectronics.com/projects/large-open-source-PLC/

XIV. Appendix

GPIO (TOP) PIN MAP

		LP	GPIO
MOSI(1)	A9	PE_4	1
CLK(1)	A11	PB_5	2
SCL (2)		PL_1	3
TX (3)		PA_5	4
MISO(2)	A15	PD_0	5
		PN_2	6
		PN_3	7
		PM_7	8
		PM_6	9
TX (2)		PA_7	10
MOSI(2)	A14	PD_1	11
		+3.3V	12
		GND	13

GPIO	LP		
26	PE_5	A8	MISO(1)
25	PB_4	A10	CS(1)
24	PL_0		SDA (2)
23	PA_4		RX (3)
22	PE_0	A3	
21	PE_1	A2	
20	PE_2	A1	
19	PE_3	A0	
18	PM_3		
17	PH_2		
16	PH_3		
15	+5V		
14	RESET		

1602 LCD DISPLAY PIN MAP

LCD PIN	PIN NAME	LAUNCHPAD
1	GND	GND
2	VCC	+5V
3	Contrast	Potentiometer
4	RS	PP_4
5	R/W	GND
6	EN	PQ_0
7	DB0	GND
8	DB1	GND
9	DB2	GND
10	DB3	GND
11	DB4	PP_1
12	DB5	PP_0
13	DB6	PC_7
14	DB7	PC_6
15	BL+	+5V
16	BL-	GND

MAIN BOARD (BOTTOM) PIN MAP

			LP	PLC
POWER JACK (24V DC)				
			PL_4	1
			PL_5	2
			PM_5	3
			PM_4	4
RX (2)			PA_6	5
		A4	PD_7	6
CLK (2)		A12	PD_3	7
SCL (0)			PB_2	8
SDA (0)			PB_3	9

PLC	LP	CPN	CONN	INFO
54	PP_2	SSR	AC	120V
53			AC	MAX8A
52	PL_2	RELAY 2	O2	
51			O1	Closed
50			IN	
49	PL_3	RELAY 1	O2	
48			O1	Closed
47			IN	
46		GND		

Reg (3.5A)	+5V		10	
Reg (3.5A)	+5V		11	
LaunchPad		+5V	12	
	GND		13	
	GND		14	
	GND		15	
LaunchPad		+3.3V	16	
LaunchPad		+3.3V	17	
LaunchPad		+3.3V	18	

PLC	LP	CPN	CONN	INFO
45		+24V	Power supply	
44	PG_1	INPUT 4	IN-	
43			IN+	10~24V
42	PK_4	INPUT 3	IN-	
41			IN+	10~24V
40	PK_5	INPUT 2	IN-	
39			IN+	10~24V
38	PM_0	INPUT 1	IN-	
37			IN+	10~24V

CS (2)	A13	PD_2	19	
	A7	PD_4	20	
	A6	PD_5	21	
	A19	PK_3	22	
	A18	PK_2	23	
TX (4)	A17	PK_1	24	
RX (4)	A16	PK_0	25	
SCL (1)		PN_5	26	
SDA (1)		PN_4	27	

PLC	LP	CPN	CONN	INFO
36		+24V		
35		+24V		
34		+24V		
33	PM_1	OUT6	LOW	
32	PM_2	OUT5	LOW	
31	PH_0	OUT4	LOW	
30	PH_1	OUT3	LOW	
29	PK_6	OUT2	LOW	
28	PK_7	OUT1	LOW	

I2C
Serial UART
SPI
analogRead()
digitalRead(), digitalWrite()
PWM: digitalRead(), digitalWrite(), analogWrite()

XV. Code

A. BASIC IO SCRIPT

```
#include "Ethernet.h"
#include "LCD.h"

#define GREENLED     PG_0
#define YELLOWLED    PF_3
#define REDLED1      PF_2
#define REDLED2      PF_1
#define BLUELED      PP_5

#define BUTTON1      PQ_1
#define BUTTON2      PP_3
#define BUTTON3      PQ_3
#define BUTTON4      PQ_2

#define OUT1         PK_7
#define OUT2         PK_6
#define OUT3         PH_1
#define OUT4         PH_0
#define OUT5         PM_2
#define OUT6         PM_1

#define IN1          PM_0
#define IN2          PK_5
#define IN3          PK_4
#define IN4          PG_1

#define RELAY1       PL_3
#define RELAY2       PL_2
#define SSR          PP_2

void printEthernetData();
EthernetServer server(80);
String currentLine;


void setup()
{
  Serial.begin(115200);
  LCD.init(PP_4, PQ_0, PP_1, PP_0, PC_7, PC_6);
  LCD.print("PLC Control Unit",1,1);
  LCD.print(" Connecting.....",2,1);

  pinMode(GREENLED, OUTPUT);
  pinMode(YELLOWLED, OUTPUT);
  pinMode(REDLED1, OUTPUT);
  pinMode(REDLED2, OUTPUT);
```

```
  pinMode(BLUELED, OUTPUT);

  pinMode(BUTTON1, INPUT);
  pinMode(BUTTON2, INPUT);
  pinMode(BUTTON3, INPUT);
  pinMode(BUTTON4, INPUT);

  pinMode(OUT1, OUTPUT);
  pinMode(OUT2, OUTPUT);
  pinMode(OUT3, OUTPUT);
  pinMode(OUT4, OUTPUT);
  pinMode(OUT5, OUTPUT);
  pinMode(OUT6, OUTPUT);

  pinMode(IN1, INPUT);
  pinMode(IN2, INPUT);
  pinMode(IN3, INPUT);
  pinMode(IN4, INPUT);

  pinMode(RELAY1, OUTPUT);
  pinMode(RELAY2, OUTPUT);
  pinMode(SSR, OUTPUT);

  Serial.println("Connecting to Ethernet....");
  IPAddress ip = IPAddress(146,252,242,129);
  IPAddress dns = IPAddress(146,252,242,12);
  IPAddress gw = IPAddress(146,252,242,254);
  IPAddress mask = IPAddress(255,255,255,0);

  Ethernet.begin(0);
//  Ethernet.begin(0, ip, dns, gw);

  server.begin();

  printEthernetData();
  delay(1000);
  LCD.print("System is ready.",2,1);
}

void loop(){

  EthernetClient client = server.available();

  if (client) {
    boolean currentLineIsBlank = true;
    while (client.connected()) {

      if (client.available()) {
        char c = client.read();
        currentLine += c;
```

```
        if (currentLine.endsWith("GET /GREEN_LED_H"))
{digitalWrite(GREENLED, HIGH);}
        if (currentLine.endsWith("GET /GREEN_LED_L"))
{digitalWrite(GREENLED, LOW);}
        if (currentLine.endsWith("GET /YELLOW_LED_H"))
{digitalWrite(YELLOWLED, HIGH);}
        if (currentLine.endsWith("GET /YELLOW_LED_L"))
{digitalWrite(YELLOWLED, LOW);}
        if (currentLine.endsWith("GET /RED_LED1_H"))
{digitalWrite(REDLED1, HIGH);}
        if (currentLine.endsWith("GET /RED_LED1_L"))
{digitalWrite(REDLED1, LOW);}
        if (currentLine.endsWith("GET /RED_LED2_H"))
{digitalWrite(REDLED2, HIGH);}
        if (currentLine.endsWith("GET /RED_LED2_L"))
{digitalWrite(REDLED2, LOW);}

        if (currentLine.endsWith("GET /OUT1_H")) {digitalWrite(OUT1,
HIGH);}
        if (currentLine.endsWith("GET /OUT1_L")) {digitalWrite(OUT1,
LOW);}
        if (currentLine.endsWith("GET /OUT2_H")) {digitalWrite(OUT2,
HIGH);}
        if (currentLine.endsWith("GET /OUT2_L")) {digitalWrite(OUT2,
LOW);}
        if (currentLine.endsWith("GET /OUT3_H")) {digitalWrite(OUT3,
HIGH);}
        if (currentLine.endsWith("GET /OUT3_L")) {digitalWrite(OUT3,
LOW);}
        if (currentLine.endsWith("GET /OUT4_H")) {digitalWrite(OUT4,
HIGH);}
        if (currentLine.endsWith("GET /OUT4_L")) {digitalWrite(OUT4,
LOW);}
        if (currentLine.endsWith("GET /OUT5_H")) {digitalWrite(OUT5,
HIGH);}
        if (currentLine.endsWith("GET /OUT5_L")) {digitalWrite(OUT5,
LOW);}
        if (currentLine.endsWith("GET /OUT6_H")) {digitalWrite(OUT6,
HIGH);}
        if (currentLine.endsWith("GET /OUT6_L")) {digitalWrite(OUT6,
LOW);}

        if (currentLine.endsWith("GET /RELAY1_H"))
{digitalWrite(RELAY1, HIGH);}
        if (currentLine.endsWith("GET /RELAY1_L"))
{digitalWrite(RELAY1, LOW);}
        if (currentLine.endsWith("GET /RELAY2_H"))
{digitalWrite(RELAY2, HIGH);}
        if (currentLine.endsWith("GET /RELAY2_L"))
{digitalWrite(RELAY2, LOW);}
        if (currentLine.endsWith("GET /SSR_H")) {digitalWrite(SSR,
HIGH);}
```

```
        if (currentLine.endsWith("GET /SSR_L")) {digitalWrite(SSR,
LOW);}

        if (currentLine.endsWith("GET /LCD_H")){
          LCD.print("Hello World!    ",2,1);

        }
        if (currentLine.endsWith("GET /LCD_L")){
          LCD.print("Bye Bye World!  ",2,1);

        }

        if (c == '\n' && currentLineIsBlank) {
          digitalWrite(BLUELED, HIGH);
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html");
          client.println("Connection: keep-alive");
          client.println();
          if (currentLine.indexOf("ajax_switch") > -1) {

              GetSwitchState(client);
          }
          else {
            client.println("<!DOCTYPE html>");
            client.println("<html>");
            client.println("<head>");
            client.println("<title>PLC | Web</title></head><body
align=center>");
            client.println("<script>");
            client.println("function GetSwitchState() {");
            client.println("nocache = \"&nocache=\"\+ Math.random() *
1000000;");
            client.println("var request = new XMLHttpRequest();");
            client.println("request.onreadystatechange = function()
{");
            client.println("if (this.readyState == 4) {");
            client.println("if (this.status == 200) {");
            client.println("if (this.responseText != null) {");

client.println("document.getElementById(\"switch_txt\")\.innerHTML =
this.responseText;");
            client.println("}}}}");
            client.println("request.open(\"GET\", \"ajax_switch\" +
nocache, true);");
            client.println("request.send(null);");
            client.println("setTimeout('GetSwitchState()', 1000);");
            client.println("}");
            client.println("</script>");
            client.println("</head>");
            client.println("<body onload=\"GetSwitchState()\">");
            client.println("<h1 align=center><font color=\"red\">PLC
Web Control Interface</font></h1>");
```

```
            client.println("<p id=\"switch_txt\">Switch state: Not
requested...</p>");

            client.print("<font color='green'>GREEN_LED</font> <button
onclick=\"location.href='/GREEN_LED_H'\">HIGH</button>");
            client.println(" <button
onclick=\"location.href='/GREEN_LED_L'\">LOW</button><br>");
            client.print("<font color='orange'>YELLOW_LED</font>
<button onclick=\"location.href='/YELLOW_LED_H'\">HIGH</button>");
            client.println(" <button
onclick=\"location.href='/YELLOW_LED_L'\">LOW</button><br>");
            client.print("<font color='red'>RED_LED1</font> <button
onclick=\"location.href='/RED_LED1_H'\">HIGH</button>");
            client.println(" <button
onclick=\"location.href='/RED_LED1_L'\">LOW</button><br>");
            client.print("<font color='red'>RED_LED2</font> <button
onclick=\"location.href='/RED_LED2_H'\">HIGH</button>");
            client.println(" <button
onclick=\"location.href='/RED_LED2_L'\">LOW</button><br><br>");

            client.print("<font color='blue'>OUTPUT_1</font> <button
onclick=\"location.href='/OUT1_H'\">ON</button>");
            client.println(" <button
onclick=\"location.href='/OUT1_L'\">OFF</button><br>");
            client.print("<font color='blue'>OUTPUT_2</font> <button
onclick=\"location.href='/OUT2_H'\">ON</button>");
            client.println(" <button
onclick=\"location.href='/OUT2_L'\">OFF</button><br>");
            client.print("<font color='blue'>OUTPUT_3</font> <button
onclick=\"location.href='/OUT3_H'\">ON</button>");
            client.println(" <button
onclick=\"location.href='/OUT3_L'\">OFF</button><br>");
            client.print("<font color='blue'>OUTPUT_4</font> <button
onclick=\"location.href='/OUT4_H'\">ON</button>");
            client.println(" <button
onclick=\"location.href='/OUT4_L'\">OFF</button><br>");
            client.print("<font color='blue'>OUTPUT_5</font> <button
onclick=\"location.href='/OUT5_H'\">ON</button>");
            client.println(" <button
onclick=\"location.href='/OUT5_L'\">OFF</button><br>");
            client.print("<font color='blue'>OUTPUT_6</font> <button
onclick=\"location.href='/OUT6_H'\">ON</button>");
            client.println(" <button
onclick=\"location.href='/OUT6_L'\">OFF</button><br><br>");

            client.print("<font color='blue'>RELAY1</font> <button
onclick=\"location.href='/RELAY1_H'\">POS1</button>");
            client.println(" <button
onclick=\"location.href='/RELAY1_L'\">POS2</button><br>");
            client.print("<font color='blue'>RELAY2</font> <button
onclick=\"location.href='/RELAY2_H'\">POS1</button>");
```

```
            client.println(" <button
onclick=\"location.href='/RELAY2_L'\">POS2</button><br>");
            client.print("<font color='blue'>SSR</font> <button
onclick=\"location.href='/SSR_H'\">ON</button>");
            client.println(" <button
onclick=\"location.href='/SSR_L'\">OFF</button><br><br>");

            client.print("<font color='red'>LCD Display</font> <button
onclick=\"location.href='/LCD_H'\">Disp1</button>");
            client.println(" <button
onclick=\"location.href='/LCD_L'\">Disp2</button><br><br>");

            client.println("</body>");
            client.println("</html>");
          }
          //Serial.print(currentLine);
          delay(10);
          digitalWrite(BLUELED, LOW);
          currentLine = "";
          break;
        }
        if (c == '\n') {
          currentLineIsBlank = true;
        }
        else if (c != '\r') {
          currentLineIsBlank = false;
        }
      }
    }
    delay(1);
    client.stop();
  }
}


void GetSwitchState(EthernetClient cl){

  if(digitalRead(BUTTON1))cl.print("<p>BUTTON1 is HIGH</p>");
  else cl.print("<p>BUTTON1 is LOW</p>");
  if(digitalRead(BUTTON2))cl.print("<p>BUTTON2 is HIGH</p>");
  else cl.print("<p>BUTTON2 is LOW</p>");
  if(digitalRead(BUTTON3))cl.print("<p>BUTTON3 is HIGH</p>");
  else cl.print("<p>BUTTON3 is LOW</p>");
  if(digitalRead(BUTTON4))cl.print("<p>BUTTON4 is HIGH</p>");
  else cl.print("<p>BUTTON4 is LOW</p>");

  if(digitalRead(IN1))cl.print("<p>INPUT1 is LOW</p>");
  else cl.print("<p>INPUT1 is HIGH</p>");
  if(digitalRead(IN2))cl.print("<p>INPUT2 is LOW</p>");
  else cl.print("<p>INPUT2 is HIGH</p>");
  if(digitalRead(IN3))cl.print("<p>INPUT3 is LOW</p>");
  else cl.print("<p>INPUT3 is HIGH</p>");
```

```
  if(digitalRead(IN4))cl.print("<p>INPUT4 is LOW</p>");
  else cl.print("<p>INPUT4 is HIGH</p>");
}



void printEthernetData() {
  // print your IP address:
  Serial.println();
  Serial.println("IP Address Information:");
  IPAddress ip = Ethernet.localIP();
    Serial.print("IP Address:\t");
  Serial.println(ip);

  // print your MAC address:

  IPAddress subnet = Ethernet.subnetMask();
  Serial.print("NetMask:\t");
  Serial.println(subnet);

  // print your gateway address:
  IPAddress gateway = Ethernet.gatewayIP();
  Serial.print("Gateway:\t");
  Serial.println(gateway);

  // print your gateway address:
  IPAddress dns = Ethernet.dnsServerIP();
  Serial.print("DNS:\t\t");
  Serial.println(dns);

}
```

B. ROBOTIC ARM SERVO CONTROL

```
/*
  Web  Server

 A simple web server that shows the value of the analog and digital
input pins
 using a WiFi BoosterPack.

 This example is written for a network using WPA encryption. For
 WEP or WPA, change the Wifi.begin() call accordingly.

 Circuit:
 * WiFi BoosterPack

 Created: October 16, 2013 by Robert Wessels (http://energia.nu)
```

```
 Derived from example Sketch by Hans Scharler
(http://www.iamshadowlord.com)

 */

#include "Ethernet.h"
#include <Servo.h>
#include <LCD.h>

// Prototypes
void printConfig();
void printEthernetData();
void printIndex();
void printHelp();

EthernetServer server(80);
Servo myservo;  // create servo object to control a servo
Servo myservo2;
Servo myservo3;
Servo myservo4;
Servo myservo5;
Servo myservo6;
Servo myservo7;
Servo myservo8;

int LED1 = PG_0;
int LED2 = PF_3;
int LED3 = PF_2;
int LED4 = PF_1;
int LED5 = PP_5;

//int potpin = PE_4;  // analog pin used to connect the potentiometer
int val;    // variable to read the value from the analog pin
int statusConfig = 0;
int counter1 = 30;
int counter2 = 20;
int counter3 = 50;
int counter4 = 15;
int counter5 = 100;
int counter6 = 15;
int counter7 = 85;
int counter8 = 45;
int x = 5;
int pos = 0;
int uppermargin = 160;
int lowermargin = 20;

void setup() {
  //delay(2000);
  Serial.begin(115200);
  // delay(2000);
  LCD.init(PP_4, PQ_0, PP_1, PP_0, PC_7, PC_6);
```

```
  LCD.print("PLC Control Unit");
  LCD.print("  Connecting..  ",2,1);
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
  pinMode(LED4, OUTPUT);
  pinMode(LED5, OUTPUT);
//  pinMode(PUSH1, INPUT_PULLUP); // released = HIGH, pressed = LOW
//  pinMode(PUSH2, INPUT_PULLUP);
  myservo.attach(PL_4);
  myservo2.attach(PL_5);
  myservo3.attach(PM_5);
  myservo4.attach(PM_4);// attaches the servo on Port F, pin 1 (Red
LED pin) to the servo object
  myservo5.attach(PA_6);
  myservo6.attach(PD_7);
  myservo7.attach(PD_3);
  myservo8.attach(PB_2);

  myservo.write(counter1);
  myservo2.write(counter2);
  myservo3.write(counter3);
  myservo4.write(counter4);
  myservo5.write(counter5);
  myservo6.write(counter6);
  myservo7.write(counter7);
  myservo8.write(counter8);

  Serial.println("Connecting to Ethernet....");
  IPAddress ip = IPAddress(192,168,0,101);
  IPAddress dns = IPAddress(192,168,0,1);
  IPAddress gw = IPAddress(192,168,0,1);
  IPAddress mask = IPAddress(255,255,255,0);

   Ethernet.begin(0);
//     Ethernet.begin(0, ip, dns, gw);

  server.begin();

  printEthernetData();

  LCD.print(" System Ready!  ",2,1);
  digitalWrite(LED1, HIGH); delay(150); digitalWrite(LED2, HIGH);
delay(150); digitalWrite(LED3, HIGH); delay(150); digitalWrite(LED4,
HIGH); delay(150); digitalWrite(LED5, HIGH); delay(150);
  digitalWrite(LED5, LOW); delay(150); digitalWrite(LED4, LOW);
delay(150); digitalWrite(LED3, LOW); delay(150); digitalWrite(LED2,
LOW); delay(150); digitalWrite(LED1, LOW); delay(150);
}

EthernetClient client;
```

```
void loop() {
  client = server.available();


//  val = analogRead(potpin);            // reads the value of the
potentiometer (value between 0 and 4095)
//  val = map(val, 0, 100, 0, 10);     // scale it to use it with the
servo (value between 0 and 180)
//  myservo.write(val);    // sets the servo position according to the
scaled value

//  delay(15);                           // waits for the servo to get
there


  if (client) {                             // if you get a client,
    Serial.print("new client on port ");           // print a message
out the serial port
    Serial.println(client.port());
    String currentLine = "";                // make a String to hold
incoming data from the client
    while (client.connected()) {            // loop while the client's
connected


      if (client.available()) {        // if there's bytes to read
from the client,

        digitalWrite(LED1, HIGH);

        char c = client.read();             // read a byte, then
                                            // This lockup is because
the recv function is blocking.
        Serial.print(c);
        if (c == '\n') {                    // if the byte is a
newline character
          // if the current line is blank, you got two newline
characters in a row.
          // that's the end of the client HTTP request, so send a
response:
          if (currentLine.length() == 0) {
            break;
          }
          else {      // if you got a newline, then clear currentLine:
            currentLine = "";
          }
        }
        else if (c != '\r') {    // if you got anything else but a
carriage return character,
          currentLine += c;      // add it to the end of the
currentLine
        }
```

```
        if (currentLine.endsWith("GET / ")) {
          printConfig();
        }

        // Check to see if the client request was "GET /H" or "GET
/L":
//        if (currentLine.endsWith("GET /RED_LED_H"))
{digitalWrite(D1_LED, HIGH);printConfig();}
//        if (currentLine.endsWith("GET /RED_LED_L"))
{digitalWrite(D1_LED, LOW);printConfig();}
//        if (currentLine.endsWith("GET /GREEN_LED_H"))
{digitalWrite(D2_LED, HIGH);printConfig();}
//        if (currentLine.endsWith("GET /GREEN_LED_L"))
{digitalWrite(D2_LED, LOW);printConfig();}

       // Increment Control

////////////////////////////////////////////////////////////////
       if (currentLine.endsWith("GET /Inc5")){
          x = 5;
          uppermargin = 165-x;
          lowermargin = 15+x;
          printConfig();
       }
       if (currentLine.endsWith("GET /Inc10")){
          x = 10;
          uppermargin = 165-x;
          lowermargin = 15+x;
          printConfig();
       }
       if (currentLine.endsWith("GET /Inc15")){
          x = 15;
          uppermargin = 165-x;
          lowermargin = 15+x;
          printConfig();
       }


        // Servo 1

////////////////////////////////////////////////////////////////
        if (currentLine.endsWith("GET /123"))
        {
          pos = counter1 + x;
          if (counter1 > uppermargin)
           {
             for(counter1; counter1 < 165; counter1 += 1)
             {                                          // in steps
of 1 degree
               myservo.write(counter1);                 // tell servo
to go to position in variable 'pos'
```

```
              delay(25);                                  // waits 15ms
for the servo to reach the position
            }
          }
          else
          {
            for(counter1; counter1 < pos; counter1 += 1)
            {                                             // in steps
of 1 degree
              myservo.write(counter1);                    // tell
servo to go to position in variable 'pos'
              delay(25);                                  // waits
15ms for the servo to reach the position
            }
          }
          printConfig();
        }
        if (currentLine.endsWith("GET /456"))
        {
          pos = counter1 - x;
          if (counter1 < lowermargin)
          {
            for(counter1; counter1 > 15; counter1 -= 1)
            {                                             // in steps
of 1 degree
              myservo.write(counter1);                    // tell
servo to go to position in variable 'pos'
              delay(25);                                  // waits
15ms for the servo to reach the position
            }
          }
          else
          {
            for(counter1; counter1 > pos; counter1 -= 1)
            {                                             // in steps
of 1 degree
              myservo.write(counter1);                    // tell
servo to go to position in variable 'pos'
              delay(25);                                  // waits
15ms for the servo to reach the position
            }
          }
          printConfig();
        }

        // Servo 2

////////////////////////////////////////////////////////////
        if (currentLine.endsWith("GET /987"))
        {
          pos = counter2 + x;
          if (counter2 > uppermargin)
```

```
           {
             for(counter2; counter2 < 165; counter2 += 1)
             {                                          // in steps
of 1 degree
               myservo2.write(counter2);                 // tell
servo to go to position in variable 'pos'
               delay(25);                               // waits 15ms
for the servo to reach the position
             }
           }
           else
           {
             for(counter2; counter2 < pos; counter2 += 1)
             {                                          // in steps
of 1 degree
               myservo2.write(counter2);                 // tell
servo to go to position in variable 'pos'
               delay(25);                               // waits
15ms for the servo to reach the position
             }
           }
           printConfig();
        }
        if (currentLine.endsWith("GET /789"))
        {
           pos = counter2 - x;
           if (counter2 < lowermargin)
           {
             for(counter2; counter2 > 15; counter2 -= 1)
             {                                          // in steps
of 1 degree
               myservo2.write(counter2);                 // tell
servo to go to position in variable 'pos'
               delay(25);                               // waits
15ms for the servo to reach the position
             }
           }
           else
           {
             for(counter2; counter2 > pos; counter2 -= 1)
             {                                          // in steps
of 1 degree
               myservo2.write(counter2);                 // tell
servo to go to position in variable 'pos'
               delay(25);                               // waits
15ms for the servo to reach the position
             }
           }
           printConfig();
        }

        // Servo 3
```

```
//////////////////////////////////////////////////////////////
        if (currentLine.endsWith("GET /654"))
        {
          pos = counter3 + x;
          if (counter3 > uppermargin)
           {
             for(counter3; counter3 < 165; counter3 += 1)
             {                                          // in steps
of 1 degree
               myservo3.write(counter3);                 // tell
servo to go to position in variable 'pos'
               delay(25);                               // waits 15ms
for the servo to reach the position
             }
           }
           else
           {
             for(counter3; counter3 < pos; counter3 += 1)
             {                                          // in steps
of 1 degree
               myservo3.write(counter3);                 // tell
servo to go to position in variable 'pos'
               delay(25);                               // waits
15ms for the servo to reach the position
             }
           }
           printConfig();
        }
        if (currentLine.endsWith("GET /321"))
        {
           pos = counter3 - x;
           if (counter3 < lowermargin)
           {
             for(counter3; counter3 > 15; counter3 -= 1)
             {                                          // in steps
of 1 degree
               myservo3.write(counter3);                 // tell
servo to go to position in variable 'pos'
               delay(25);                               // waits
15ms for the servo to reach the position
             }
           }
           else
           {
             for(counter3; counter3 > pos; counter3 -= 1)
             {                                          // in steps
of 1 degree
               myservo3.write(counter3);                 // tell
servo to go to position in variable 'pos'
               delay(25);                               // waits
15ms for the servo to reach the position
```

```
            }
          }
          printConfig();
        }

        // Servo 4

////////////////////////////////////////////////////////////
        if (currentLine.endsWith("GET /147"))
        {
          pos = counter4 + x;
          if (counter4 > uppermargin)
           {
             for(counter4; counter4 < 165; counter4 += 1)
             {                                          // in steps
of 1 degree
               myservo4.write(counter4);                 // tell
servo to go to position in variable 'pos'
               delay(25);                               // waits 15ms
for the servo to reach the position
             }
           }
           else
           {
             for(counter4; counter4 < pos; counter4 += 1)
             {                                           // in steps
of 1 degree
               myservo4.write(counter4);                  // tell
servo to go to position in variable 'pos'
               delay(25);                                // waits
15ms for the servo to reach the position
             }
           }
          printConfig();
        }
        if (currentLine.endsWith("GET /741"))
        {
          pos = counter4 - x;
          if (counter4 < lowermargin)
          {
            for(counter4; counter4 > 15; counter4 -= 1)
            {                                           // in steps
of 1 degree
              myservo4.write(counter4);                  // tell
servo to go to position in variable 'pos'
              delay(25);                                // waits
15ms for the servo to reach the position
            }
          }
          else
          {
            for(counter4; counter4 > pos; counter4 -= 1)
```

```
            {                                                    // in steps
of 1 degree
              myservo4.write(counter4);                           // tell
servo to go to position in variable 'pos'
              delay(25);                                         // waits
15ms for the servo to reach the position
            }
          }
          printConfig();
        }

        // Servo 5

////////////////////////////////////////////////////////////////
        if (currentLine.endsWith("GET /ABC"))
        {
          pos = counter5 + x;
          if (counter5 > uppermargin)
           {
             for(counter5; counter5 < 165; counter5 += 1)
             {                                                  // in steps
of 1 degree
              myservo5.write(counter5);                           // tell
servo to go to position in variable 'pos'
              delay(25);                                        // waits 15ms
for the servo to reach the position
            }
          }
          else
          {
            for(counter5; counter5 < pos; counter5 += 1)
            {                                                    // in steps
of 1 degree
              myservo5.write(counter5);                           // tell
servo to go to position in variable 'pos'
              delay(25);                                         // waits
15ms for the servo to reach the position
            }
          }
          printConfig();
        }
        if (currentLine.endsWith("GET /CBA"))
        {
          pos = counter5 - x;
          if (counter5 < lowermargin)
          {
            for(counter5; counter5 > 15; counter5 -= 1)
            {                                                    // in steps
of 1 degree
              myservo5.write(counter5);                           // tell
servo to go to position in variable 'pos'
```

```
              delay(25);                                    // waits
15ms for the servo to reach the position
            }
          }
          else
          {
            for(counter5; counter5 > pos; counter5 -= 1)
            {                                               // in steps
of 1 degree
              myservo5.write(counter5);                      // tell
servo to go to position in variable 'pos'
              delay(25);                                    // waits
15ms for the servo to reach the position
            }
          }
          printConfig();
        }

        // Servo 6

////////////////////////////////////////////////////////////
        if (currentLine.endsWith("GET /DEF"))
        {
          pos = counter6 + x;
          if (counter6 > uppermargin)
           {
            for(counter6; counter6 < 165; counter6 += 1)
            {                                             // in steps
of 1 degree
              myservo6.write(counter6);                    // tell
servo to go to position in variable 'pos'
              delay(25);                                  // waits 15ms
for the servo to reach the position
            }
          }
          else
          {
            for(counter6; counter6 < pos; counter6 += 1)
            {                                             // in steps
of 1 degree
              myservo6.write(counter6);                    // tell
servo to go to position in variable 'pos'
              delay(25);                                  // waits
15ms for the servo to reach the position
            }
          }
          printConfig();
        }
        if (currentLine.endsWith("GET /FED"))
        {
          pos = counter6 - x;
          if (counter6 < lowermargin)
```

```
          {
            for(counter6; counter6 > 15; counter6 -= 1)
            {                                                  // in steps
of 1 degree
              myservo6.write(counter6);                         // tell
servo to go to position in variable 'pos'
              delay(25);                                       // waits
15ms for the servo to reach the position
            }
          }
          else
          {
            for(counter6; counter6 > pos; counter6 -= 1)
            {                                                  // in steps
of 1 degree
              myservo6.write(counter6);                         // tell
servo to go to position in variable 'pos'
              delay(25);                                       // waits
15ms for the servo to reach the position
            }
          }
          printConfig();
        }

        // Servo 7

////////////////////////////////////////////////////////////////
        if (currentLine.endsWith("GET /GHI"))
        {
          pos = counter7 + x;
          if (counter7 > uppermargin)
           {
            for(counter7; counter7 < 165; counter7 += 1)
            {                                                 // in steps
of 1 degree
              myservo7.write(counter7);                        // tell
servo to go to position in variable 'pos'
              delay(25);                                      // waits 15ms
for the servo to reach the position
            }
          }
          else
          {
            for(counter7; counter7 < pos; counter7 += 1)
            {                                                  // in steps
of 1 degree
              myservo7.write(counter7);                         // tell
servo to go to position in variable 'pos'
              delay(25);                                       // waits
15ms for the servo to reach the position
            }
          }
```

```
           printConfig();
         }
         if (currentLine.endsWith("GET /IHG"))
         {
           pos = counter7 - x;
           if (counter7 < lowermargin)
           {
             for(counter7; counter7 > 15; counter7 -= 1)
             {                                                  // in steps
of 1 degree
               myservo7.write(counter7);                    // tell
servo to go to position in variable 'pos'
               delay(25);                                   // waits
15ms for the servo to reach the position
             }
           }
           else
           {
             for(counter7; counter7 > pos; counter7 -= 1)
             {                                                  // in steps
of 1 degree
               myservo7.write(counter7);                    // tell
servo to go to position in variable 'pos'
               delay(25);                                   // waits
15ms for the servo to reach the position
             }
           }
           printConfig();
         }

         // Servo 8

////////////////////////////////////////////////////////////////
         if (currentLine.endsWith("GET /JKL"))
         {
           pos = counter8 + x;
           if (counter8 > uppermargin)
            {
             for(counter8; counter8 < 165; counter8 += 1)
             {                                                 // in steps
of 1 degree
               myservo8.write(counter8);                    // tell
servo to go to position in variable 'pos'
               delay(25);                                   // waits 15ms
for the servo to reach the position
             }
           }
           else
           {
             for(counter8; counter8 < pos; counter8 += 1)
             {                                                  // in steps
of 1 degree
```

```
              myservo8.write(counter8);                  // tell
servo to go to position in variable 'pos'
              delay(25);                                 // waits
15ms for the servo to reach the position
            }
          }
          printConfig();
        }
        if (currentLine.endsWith("GET /LKJ"))
        {
          pos = counter8 - x;
          if (counter8 < lowermargin)
          {
            for(counter8; counter8 > 15; counter8 -= 1)
            {                                            // in steps
of 1 degree
              myservo8.write(counter8);                  // tell
servo to go to position in variable 'pos'
              delay(25);                                 // waits
15ms for the servo to reach the position
            }
          }
          else
          {
            for(counter8; counter8 > pos; counter8 -= 1)
            {                                            // in steps
of 1 degree
              myservo8.write(counter8);                  // tell
servo to go to position in variable 'pos'
              delay(25);                                 // waits
15ms for the servo to reach the position
            }
          }
          printConfig();
        }


      }
    }
    // close the connection:
    client.stop();
    digitalWrite(LED1, LOW);
    LCD.clear();
    LCD.print(counter1,1,1);
    LCD.print(counter2,1,5);
    LCD.print(counter3,1,9);
    LCD.print(counter4,1,13);
    LCD.print(counter5,2,1);
    LCD.print(counter6,2,5);
    LCD.print(counter7,2,9);
    LCD.print(counter8,2,13);
    //Serial.println("client disonnected");
```

```
  }
}


void printConfig()
{
  // HTTP headers always start with a response code (e.g. HTTP/1.1 200
OK)
  // and a content-type so the client knows what's coming, then a
blank line:
  client.println("HTTP/1.1 200 OK");
  client.println("Content-type:text/html");
  client.println();
  client.println("<html><head><title>Stepper Motor
Control</title></head><body align=center>");
  client.println("<h1 align=center><font color=\"red\">PLC Robotic Arm
Control</font></h1>");


  // the content of the HTTP response follows the header:
  // Added: nicer buttons
//  client.print("<font color='green'>GREEN_LED</font> <button
onclick=\"location.href='/GREEN_LED_H'\">HIGH</button>");
//  client.println(" <button
onclick=\"location.href='/GREEN_LED_L'\">LOW</button><br>");
//  client.print("<font color='red'>RED_LED</font> <button
onclick=\"location.href='/RED_LED_H'\">HIGH</button>");
//  client.println(" <button
onclick=\"location.href='/RED_LED_L'\">LOW</button><br><br>");

  //client.print("<font color='green'>Increment by</font>");
//  client.println("<form action.asp/""/<font color='green'>Increment
by: </font>");
//  client.println("<input type= \"radio\" name= \"inc\" value=
\"5\">5  ");
//  client.println("<input type= \"radio\" name= \"inc\" value=
\"10\">10  ");
//  client.println("<input type= \"radio\" name= \"inc\" value=
\"15\">15 </form>");
//  //client.print("<input type= \"submit\" value=
\"Submit\"></form>");
//  client.println("<input type=\"button\" onclick=\"myFunction()\"
value=\"Increment\"></form><br><br>");
//  client.println("<script>function myFunction(){var inc =
document.forms[0].inc;var txt = /""/; var i;for(i = 0; i<inc.length;
i++){if(inc[i].checked){txt = txt + inc[i].value + /"
"/;}}}</script>");
//
  client.print("<font color='black'>Increment degrees by: </font>");
  client.print("<button onclick=\"location.href='/Inc5'\"> 5
</button>");
```

```
  client.print("<button onclick=\"location.href='/Inc10'\"> 10
</button>");
  client.print("<button onclick=\"location.href='/Inc15'\"> 15
</button><br><br>");

  client.print("<font color='blue'>Move_Servo1</font> <button
onclick=\"location.href='/123'\">Clockwise</button>");
  client.println("<button onclick=\"location.href='/456'\">Counter-
Clockwise</button><br>");

  client.print("<font color='green'>Move_Servo2</font> <button
onclick=\"location.href='/987'\">Clockwise</button>");
  client.println("<button onclick=\"location.href='/789'\">Counter-
Clockwise</button><br>");

  client.print("<font color='blue'>Move_Servo3</font> <button
onclick=\"location.href='/654'\">Clockwise</button>");
  client.println("<button onclick=\"location.href='/321'\">Counter-
Clockwise</button><br>");

  client.print("<font color='green'>Move_Servo4</font> <button
onclick=\"location.href='/147'\">Clockwise</button>");
  client.println("<button onclick=\"location.href='/741'\">Counter-
Clockwise</button><br>");

  client.print("<font color='blue'>Move_Servo5</font> <button
onclick=\"location.href='/ABC'\">Clockwise</button>");
  client.println("<button onclick=\"location.href='/CBA'\">Counter-
Clockwise</button><br>");

  client.print("<font color='green'>Move_Servo6</font> <button
onclick=\"location.href='/DEF'\">Clockwise</button>");
  client.println("<button onclick=\"location.href='/FED'\">Counter-
Clockwise</button><br>");

  client.print("<font color='blue'>Move_Servo7</font> <button
onclick=\"location.href='/GHI'\">Clockwise</button>");
  client.println("<button onclick=\"location.href='/IHG'\">Counter-
Clockwise</button><br>");

  client.print("<font color='green'>Move_Servo8</font> <button
onclick=\"location.href='/JKL'\">Clockwise</button>");
  client.println("<button onclick=\"location.href='/LKJ'\">Counter-
Clockwise</button><br>");


//  client.println("PUSH1 ");
//  if(digitalRead(PUSH1))client.print("is HIGH<br>");
//  else client.print("is LOW<br>");
//  client.println("PUSH2 ");
//  if(digitalRead(PUSH2))client.print("is HIGH<br>");
//  else client.print("is LOW<br>");
```

```
  client.println("</body></html>");
  // The HTTP response ends with another blank line:
  client.println();
  // break out of the while loop:
}


void printEthernetData() {
  // print your IP address:
  Serial.println();
  Serial.println("IP Address Information:");
  IPAddress ip = Ethernet.localIP();
    Serial.print("IP Address:\t");
  Serial.println(ip);

  // print your MAC address:

  IPAddress subnet = Ethernet.subnetMask();
  Serial.print("NetMask:\t");
  Serial.println(subnet);

  // print your gateway address:
  IPAddress gateway = Ethernet.gatewayIP();
  Serial.print("Gateway:\t");
  Serial.println(gateway);

  // print your gateway address:
  IPAddress dns = Ethernet.dnsServerIP();
  Serial.print("DNS:\t\t");
  Serial.println(dns);
```

C. RFID SETUP

```
// pin mapping for SPI(2) on the Stellaris (also tiva??) using Rei
Vilo's pinmap
// SDA  -  CS(2) -> PB_5 = pin 2 as an integer
// SCK  -  SCK(2) -> PB_4
// Mosi -  MOSI(2) -> PB_7
// Miso -  MISO(2) -> PB_6
// IRQ  - Not connected
// GND  - GND
// RST  - reset -> PF_0
// VCC  - +3.3V = pin 1

#include <Mfrc522.h>
#include <SPI.h>
#include <Ethernet.h>
#include <LCD.h>
```

```
String txData ="";

byte mac[]     = { 0x00, 0x1A, 0xB6, 0x02, 0xAE, 0x37 };
byte ip[]      = { 10,100,120,53 }; // Must be unique on local network
byte gateway[] = { 10,100,120,1 };   // I left these in for reference
in case you need them
byte subnet[]  = { 255,255,255,0 };   // I left these in for reference
in case you need them
IPAddress server(10,100,120,171);   //the server we send the data to
int statusConfig = 0;
EthernetClient client;

int chipSelectPin1 = PA_4;
int chipSelectPin2 = PE_0;
int chipSelectPin3 = PE_1;
int chipSelectPin4 = PE_2;
int NRSTPD1 = PA_5;
int NRSTPD2 = PD_0;
int NRSTPD3 = PN_2;
int NRSTPD4 = PN_3;

int LED1 = PG_0;
int LED2 = PF_3;
int LED3 = PF_2;
int LED4 = PF_1;
int LED5 = PP_5;

int counter = 0;

unsigned char serNum[5];

String cardID1 = "1";
String location1 = "";
String last_cardID1 = "2";
boolean same_cardID1 = true;
boolean empty_cardID1 = true;

String cardID2 = "1";
String location2 = "";
String last_cardID2 = "2";
boolean same_cardID2 = true;
boolean empty_cardID2 = true;

String cardID3 = "1";
String location3 = "";
String last_cardID3 = "2";
boolean same_cardID3 = true;
boolean empty_cardID3 = true;

String cardID4 = "1";
String location4 = "";
```

```
String last_cardID4 = "2";
boolean same_cardID4 = true;
boolean empty_cardID4 = true;

void setup(){

  Serial.begin(115200);
  LCD.init(PP_4, PQ_0, PP_1, PP_0, PC_7, PC_6);
  LCD.print("PLC RFID TRACKER",1,1);
  LCD.print(" Connecting.... ",2,1);
  Ethernet.begin(0); //use default setting
  Serial.print("My IP address: ");
  Serial.println(Ethernet.localIP());
  Serial.println();
  delay(1000);
  SPI.setModule(1); // using SPI module 2...

  pinMode(chipSelectPin1, OUTPUT); //moved this here from Mfrc522.cpp
  pinMode(chipSelectPin2, OUTPUT);
  pinMode(chipSelectPin3, OUTPUT);
  pinMode(chipSelectPin4, OUTPUT);

  pinMode(NRSTPD1, OUTPUT); //moved this here from Mfrc522.cpp
  pinMode(NRSTPD2, OUTPUT);
  pinMode(NRSTPD3, OUTPUT);
  pinMode(NRSTPD4, OUTPUT);

  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
  pinMode(LED4, OUTPUT);
  pinMode(LED5, OUTPUT);

  LCD.print("  System ready  ",2,1);
  digitalWrite(LED1, HIGH); delay(150); digitalWrite(LED2, HIGH);
delay(150); digitalWrite(LED3, HIGH); delay(150); digitalWrite(LED4,
HIGH); delay(150); digitalWrite(LED5, HIGH); delay(150);
  digitalWrite(LED5, LOW); delay(150); digitalWrite(LED4, LOW);
delay(150); digitalWrite(LED3, LOW); delay(150); digitalWrite(LED2,
LOW); delay(150); digitalWrite(LED1, LOW); delay(150);
  LCD.print("1:  2:  3:  4:  ",2,1);
}



void loop()
{
  for (int x=1; x<5; x++){
    switch(x){
      case 1:
        digitalWrite(chipSelectPin1, LOW);
        digitalWrite(chipSelectPin2, HIGH);
```

```
        digitalWrite(chipSelectPin3, HIGH);
        digitalWrite(chipSelectPin4, HIGH);
        digitalWrite(NRSTPD1, HIGH);
        digitalWrite(NRSTPD2, LOW);
        digitalWrite(NRSTPD3, LOW);
        digitalWrite(NRSTPD4, LOW);
        delay(100);
        getRFID(PA_4, PA_5, 1, "A", 1, 1, 1); // (chipSelectPin,
NRSTPD, case, Section, Station, Rack, Shelf)
        break;
      case 2:
        digitalWrite(chipSelectPin1, HIGH);
        digitalWrite(chipSelectPin2, LOW);
        digitalWrite(chipSelectPin3, HIGH);
        digitalWrite(chipSelectPin4, HIGH);
        digitalWrite(NRSTPD1, LOW);
        digitalWrite(NRSTPD2, HIGH);
        digitalWrite(NRSTPD3, LOW);
        digitalWrite(NRSTPD4, LOW);
        delay(100);
        getRFID(PE_0, PD_0, 2, "B", 1, 1, 2);
        break;
      case 3:
        digitalWrite(chipSelectPin1, HIGH);
        digitalWrite(chipSelectPin2, HIGH);
        digitalWrite(chipSelectPin3, LOW);
        digitalWrite(chipSelectPin4, HIGH);
        digitalWrite(NRSTPD1, LOW);
        digitalWrite(NRSTPD2, LOW);
        digitalWrite(NRSTPD3, HIGH);
        digitalWrite(NRSTPD4, LOW);
        delay(100);
        getRFID(PE_1, PN_2, 3, "C", 1, 1, 3);
        break;
      case 4:
        digitalWrite(chipSelectPin1, HIGH);
        digitalWrite(chipSelectPin2, HIGH);
        digitalWrite(chipSelectPin3, HIGH);
        digitalWrite(chipSelectPin4, LOW);
        digitalWrite(NRSTPD1, LOW);
        digitalWrite(NRSTPD2, LOW);
        digitalWrite(NRSTPD3, LOW);
        digitalWrite(NRSTPD4, HIGH);
        delay(100);
        getRFID(PE_2, PN_3, 4, "D", 1, 1, 4);
        break;

    }
    delay(10);

  }
}
```

```
void getRFID(int chipSelectPin, int NRSTPD, int case_number, char
Section[], int Station, int Rack, int Shelf) {

  Mfrc522 Mfrc522(chipSelectPin, NRSTPD);
  Mfrc522.Init();
  unsigned char status;
  unsigned char status1;
  unsigned char str[MAX_LEN];
  unsigned char RC_size;
  unsigned char blockAddr;
  String mynum = "";

  status1 = Mfrc522.Request(PICC_REQIDL, str);
  status = Mfrc522.Anticoll(str);
  memcpy(serNum, str, 5);

  switch(case_number){

    //--------------------------------------------------------------------
----------------------------------------
    case 1:
    for (int check=0; check<100; check++){

      location1 =
String(Section)+String(Station)+String(Rack)+String(Shelf);

      if (status == MI_OK){

        cardID1 =
String(serNum[0])+String(serNum[1])+String(serNum[2])+String(serNum[3]
)+String(serNum[4]);

        if (cardID1 != last_cardID1){



          Serial.print(cardID1);
          Serial.print(", ");
          Serial.println(location1);

          //delay(1000);
          digitalWrite(LED1, HIGH);
          sendPHP(cardID1, "", location1);

          last_cardID1 = cardID1;
          cardID1 = "";
          same_cardID1 = true;
        }
        else if (cardID1 == last_cardID1 && same_cardID1 == true){
```

```
          Serial.println("Same card detected!");
          LCD.print("AT",2,3);
          same_cardID1 = false;
        }

        empty_cardID1 = true;
        break;
      }
      else if (status != MI_OK && empty_cardID1 == true){

         if (status != MI_OK && status1 != MI_OK){
           counter++;
         }
        if (counter == 100){
          digitalWrite(LED1, LOW);
          sendPHP(last_cardID1, "EMPTY", location1);
          Serial.println("1 Empty!");
          LCD.print("ET",2,3);
          cardID1 = "1";
          location1 = "";
          last_cardID1 = "2";
          empty_cardID1 = false;
          counter = 0;
          break;
        }
        }

      }
      counter = 0;
      Mfrc522.Halt();
      break;



   //-------------------------------------------------------------------
-----------------------------------------
   case 2:
   for (int check=0; check<100; check++){

     location2 =
String(Section)+String(Station)+String(Rack)+String(Shelf);

     if (status == MI_OK){

        cardID2 =
String(serNum[0])+String(serNum[1])+String(serNum[2])+String(serNum[3]
)+String(serNum[4]);

        if (cardID2 != last_cardID2){
```

```
          Serial.print(cardID2);
          Serial.print(", ");
          Serial.println(location2);

          //delay(1000);
          digitalWrite(LED2, HIGH);
          sendPHP(cardID2, "", location2);

          last_cardID2 = cardID2;
          cardID2 = "";
          same_cardID2 = true;
        }
        else if (cardID2 == last_cardID2 && same_cardID2 == true){
          Serial.println("Same card detected!");
          LCD.print("AT",2,7);
          same_cardID2 = false;
        }
        empty_cardID2 = true;
        break;
      }
      else if (status != MI_OK && empty_cardID2 == true){

         if (status != MI_OK && status1 != MI_OK){
           counter++;
         }
        if (counter == 100){
          digitalWrite(LED2, LOW);
          sendPHP(last_cardID2, "EMPTY", location2);
          Serial.println("2 Empty!");
          LCD.print("ET",2,7);
          cardID2 = "1";
          location2 = "";
          last_cardID2 = "2";
          empty_cardID2 = false;
          counter = 0;
          break;
        }
        }
      }
      counter = 0;
      Mfrc522.Halt();
      break;

   //--------------------------------------------------------------------
-----------------------------------------
   case 3:
   for (int check=0; check<100; check++){

     location3 =
String(Section)+String(Station)+String(Rack)+String(Shelf);

     if (status == MI_OK){
```

```
        cardID3 =
String(serNum[0])+String(serNum[1])+String(serNum[2])+String(serNum[3]
)+String(serNum[4]);

        if (cardID3 != last_cardID3){



          Serial.print(cardID3);
          Serial.print(", ");
          Serial.println(location3);

          //delay(1000);
          digitalWrite(LED3, HIGH);
          sendPHP(cardID3, "", location3);

          last_cardID3 = cardID3;
          cardID3 = "";
          same_cardID3 = true;

        }
        else if (cardID3 == last_cardID3 && same_cardID3 == true){
          Serial.println("Same card detected!");
          LCD.print("AT",2,11);
          same_cardID3 = false;

        }

        empty_cardID3 = true;
        break;
      }

      else if (status != MI_OK && empty_cardID3 == true){

         if (status != MI_OK && status1 != MI_OK){
           counter++;
         }
        if (counter == 100){
          digitalWrite(LED3, LOW);
          sendPHP(last_cardID3, "EMPTY", location3);
          Serial.println("3 Empty!");
          LCD.print("ET",2,11);
          cardID3 = "1";
          location3 = "";
          last_cardID3 = "2";
          empty_cardID3 = false;
          counter = 0;
          break;
        }

        }
```

```
      }
      counter = 0;
      Mfrc522.Halt();
      break;


    //--------------------------------------------------------------------
----------------------------------------
   case 4:
   for (int check=0; check<100; check++){

     location4 =
String(Section)+String(Station)+String(Rack)+String(Shelf);

     if (status == MI_OK){

        cardID4 =
String(serNum[0])+String(serNum[1])+String(serNum[2])+String(serNum[3]
)+String(serNum[4]);

        if (cardID4 != last_cardID4){



          Serial.print(cardID4);
          Serial.print(", ");
          Serial.println(location4);

          //delay(1000);
          digitalWrite(LED4, HIGH);
          sendPHP(cardID4, "", location4);

          last_cardID4 = cardID4;
          cardID4 = "";
          same_cardID4 = true;

        }
        else if (cardID4 == last_cardID4 && same_cardID4 == true){
          Serial.println("Same card detected!");
          LCD.print("AT",2,15);
          same_cardID4 = false;

        }

        empty_cardID4 = true;
        break;
      }

      else if (status != MI_OK && empty_cardID4 == true){

         if (status != MI_OK && status1 != MI_OK){
           counter++;
```

```
         }
        if (counter == 100){
          digitalWrite(LED4, LOW);
          sendPHP(last_cardID4, "EMPTY", location4);
          Serial.println("4 Empty!");
          LCD.print("ET",2,15);
          cardID4 = "1";
          location4 = "";
          last_cardID4 = "2";
          empty_cardID4 = false;
          counter = 0;
          break;
        }

        }
      }
      counter = 0;
      Mfrc522.Halt();
      break;
  }
}




void sendPHP(String cardID, String status1, String location){

  digitalWrite(LED5, HIGH);

  txData = "RFID=" + String(cardID) + "&STATUS=" + String(status1) +
"&LOCATION=" + String(location);

  if (client.connect(server,80)){

    Serial.println("Connected to yourwebsite...");

    client.print("POST /submit.php HTTP/1.1\n"); //php file directory
in your server
    client.print("Host: 10.100.120.171\n");   //change it to your host
    client.println("User-Agent: energia-ethernet");
    client.print("Connection: close\n");
    client.print("Content-Type: application/x-www-form-urlencoded\n");
    client.print("Content-Length: ");
    client.print(txData.length());
    client.print("\n\n");
    client.print(txData);    // after all of the required, we send the
data
    client.stop();
    Serial.println(txData);
    txData = "";
    digitalWrite(LED5, LOW);
```

```
  }
  else{
    Serial.println("Connection Failed."); // Again the Serial is for
feedback.
    Serial.println();
//    client.stop();
//    delay(1000);
//    client.connect(server,80);
//    delay(4000);
  }
  return;
}
```