UNIVERSIDAD DE MONTERREY
DIVISIÓN DE INGENIERÍA Y TECNOLOGÍAS
DEPARTAMENTO DE INGENIERÍA

Reporte técnico

Sistemas embebidos	Gerhard Didier De la Mora Hernández	290593
Dr. Christian Hassard	Hugo Gerardo Treviño Garza	264284
	Teresa Peña Hernández	243696
	Víctor Arizai Flores Vega	200866
	William Alexander Bejarano Pinilla	568103

San Pedro Garza García, N.L., 4 de enero de 2020

Contents

Introducción

Un sistema embebido es una máquina o entidad virtual que puede aceptar alguna entrada, analizarla y luego debe dar una salida para la cual el sistema está hecho o debe manejar lo que esté conectado a él.

En este proyecto de celda de manufactura se pretende manejar tres sistemas, los cuales, con diferentes protocolos, se comunicarán y formarán un sistema unificado.
La primera parte consiste en un brazo robótico de tres grados de libertad controlado por un MCU ATmega32u4; un segundo microcontrolador, el ATmega328P controlará el movimiento de una banda donde se transportará material; y, por último, en el panel de control una Raspberry Pi con SoC Broadcom BCM2835, el cual jugará el papel de interfaz entre el usuario y la celda de manufactura, proporcionando información útil y en caso de alguna emergencia mandarla llamar desde el panel.

El sistema consiste en una celda de manufactura a escala donde un brazo robótico servocontrolado será capaz de identificar cierto material transportado en la banda, levantarlo y depositarlo en algún lugar predefinido según las características del material. Como en cualquier sistema embebido, una interfaz sencilla e informativa es primordial, por lo que el panel de control será nuestra interfaz donde, además de mostrar información como conteo de objetos, temperatura, estatus de la máquina, etc… desde una pantalla LCD se podrán operar ciertas funciones (paro de emergencia).

Objetivos

- Uso de tres tipos de microcontroladores distintos.
- Uso de tres sensores distintos, uno en cada microcontrolador como subsistema de reconocimiento.
- Aplicación del protocolo de comunicación I2C entre el microcontrolador de la banda y el microcontrolador del brazo robótico.

- Aplicación del protocolo de comunicación serial (USB) entre el microcontrolador de la banda y el panel de control.
- Aplicación del protocolo de comunicación SPI entre el microcontrolador del brazo robótico y el panel de control.
- Implementar una interfaz de usuario a través de pantalla LCD en panel de control.
- Contador de objetos y almacenamiento en memoria de brazo robótico.
- Parada de emergencia por medio de pin.
- Uso de timers para la separación de piezas.

Avances

Banda transportadora

La estación de transporte será el inicio de la celda de manufactura. Consiste en una banda transportadora que transporta objetos de diferente tonalidad (blanco y negro) y un sensor de proximidad que es activado cuando el objeto llega al límite de la banda. Además, contará con un actuador junto con su driver y, para todo el sistema, se utilizará un microcontrolador.

Objetivo

Al iniciar el movimiento de la banda mediante actuador (motor DC), el sensor de proximidad detecta objeto y comienza un tiempo de detención del movimiento al mismo momento en que el objeto se transporta a su posición final en la banda. Al final, cuando el tiempo se cumple, la banda se detiene y se manda una señal a la siguiente estación para comenzar. Para inicializarla se realiza mediante un protocolo de comunicación entre la banda de transporte y el tablero de control, al igual que para la parada normal. Para la parada de emergencia, se activa en el tablero de control y se detiene la banda, pero mediante una interrupción. En la figura 1 se muestra la imagen de la banda transportadora.

Fig. 1 Banda transportadora

Componentes

- 1 microcontrolador Arduino UNO (ATmega328P)
- Motor DC de alto torque, 38 RMP, 3.6 kg/cm (B01-1:220. MOTORREDUCTOR)
- Driver para motor DC (L298N)
- Sensor infrarrojo de proximidad SHARP GP2Y0A21YK
- Material para banda
- Rodillo o tubo
- Soportes y estructura de la banda

Es necesario saber las características de operación de los componentes como el voltaje, la corriente y la potencia para poder hacer un diseño óptimo y funcional. En la tabla 1 se presentan dichas características de los componentes que se utilizarán.

Tabla 1. Características y consumo de operación de los componentes

Componente	Voltaje (V)	Corriente (mA)	Potencia (mW)*
Motorreductor de alto torque B01:220	5	75 – 670	3350
Controlador para motor DC L298N	5 – 35	30 – 4000	20000
MCU ATmega328P (Arduino UNO)	1.8 – 5.5	200	1100
Sensor de proximidad SHARP GP2Y0A21YK)	5	30	150

*Potencia máxima

Entradas y salidas del microcontrolador

Entradas:

- Pin A0 (salida del sensor proximidad; análogo) Pin 3 (enable del controlador de motor DC)
- Pin 4 (in1 del controlador del motor DC)
- Pin 5 (in2 del controlador del motor DC)

Salidas:

- Pin A4 del arduino uno a Pin 3 (SCL) del arduino leonardo o a pin de SCL (protocolo i2c)
- Pin A5 del arduino uno a Pin 2 (SDA) del arduino leonardo o a pin de SDA (protocolo i2c)

Diagramas de conexiones en general

La figura 2 muestra el diagrama de conexiones del circuito que se utilizará en la banda transportadora.

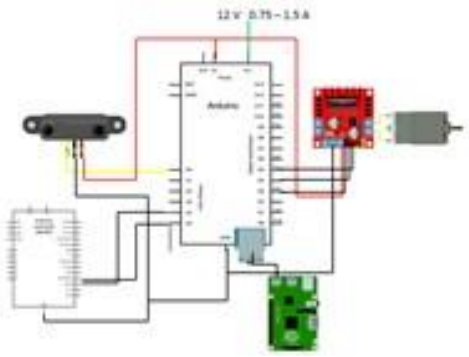


Fig. 2 Diagrama de conexiones

Sensor

El sensor que se utilizará en la banda transportadora es el SHARP GP2Y0A02YK0F, mostrado en la figura 3, el cual es un sensor óptico de distancia compuesto por un IR LED (LED Infrarrojo) con un dispositivo de detector de posición. El sensor es analógico, por lo que la salida será mediante voltaje y se tendrá que hacer una conversión para obtener el valor en distancia.

Fig. 3 Sensor SHARP GP2Y0A02YK0F

El funcionamiento consiste en que el IR LED manda una señal, la cual es reflejada con un objeto y el dispositivo de detector de posición la recibe. Se registra el tiempo que tomó en ir y regresar la señal y, en base a eso, se manda la salida de tensión analógica. La figura 4 muestra su funcionamiento gráficamente.

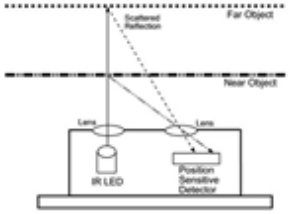


Fig. 4 Funcionamiento del sensor

La tensión de alimentación del sensor es de 4.5 a 5.5V y el consumo de corriente de 33mA. El intervalo entre mediciones es de unos 80ms, siendo el principal obstáculo del sensor la luminosidad del ambiente. La salida analógica tiene un valor de 2.5V a 20 cm y de 0.4 a 150cm. Sin embargo, como hemos mencionado, la respuesta es no lineal, por lo que es necesario interpolar el valor para obtener un nivel de precisión adecuado. El rango de medición va de 20 a 150 cm.

Para la aplicación deseada no se medirá distancia, solamente con que detecte algún objeto. Si se desea saber un valor menor a 20 se tendrán dificultes porque se registrará una misma salida, pero con distancias diferentes. La figura 5 muestra la gráfica del rango de medición.

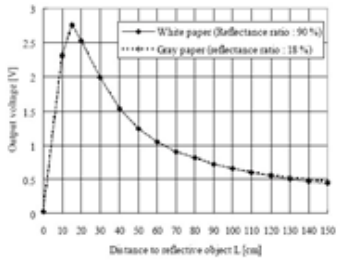


Fig. 5 Gráfica del rango de medición

En la figura 6 puede apreciarse el diagrama de conexiones o terminales del sensor y las conexiones al microcontrolador.

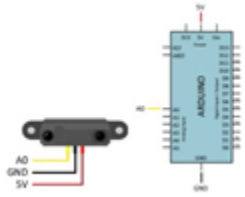


Fig. 6 Diagrama de conexiones

Otras aplicaciones del sensor puede ser en sensores de posición, fin de carrera o medir distancias.

Microcontrolador

El microcontrolador que se va a utilizar es el Arduino UNO (ATmega328P), mostrado en la figura 7, debido a su sencilla funcionalidad y que cumple con las características para implementar la estación de transporte.

Fig. 7 Arduino UNO

Un resumen de las características del dispositivo en el Arduino Uno (ATmega328P) son: AVR 8-bit MCU, RISC (Reduced Instruction Set Computing), 32K bytes de In-System Programmable Flash de los cuales 0.5 KB son usados para bootloader, 1 KB EEPROM, 2KB de RAM, una velocidad de reloj de 16 MHz, 14 pines para entradas y salidas digitales, de los cuales 6 pueden usarse como salida PWM, 6 entradas análogas y 8 canales ADC. Incluye un USART, Interfaz Serial 2-wire (I2C), puerto serial SPI. El tipo de microprocesador AVR utiliza la arquitectura Harvard, con buses separados para la memoria de datos y la memoria de programa. La figura 8 muestra dicha arquitectura.

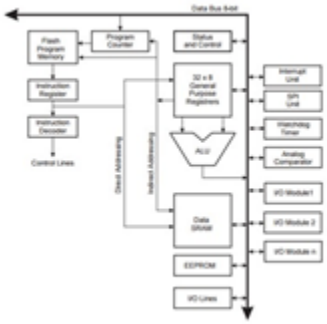


Fig. 8 Arquitectura Harvard

Protocolos de comunicación

Los microcontroladores empleados cuentan con pines y protocolos configurados que harán más sencilla la comunicación entre ellos. Para la banda de transporte se utilizarán 2 protocolos de comunicación:

- I2C (Circuito Inter-Integrado; protocolo síncrono), que será entre la banda de transporte y el manipulador robótico
- USB (Bus Universal en Serie; protocolo síncrono), que será entre la banda de transporte y el tablero de control; en este caso solamente será unidireccional, por lo que la banda de transporte sólo recibe datos del tablero de control.

Actuador

El actuador que se utilizará en el sistema, mostrado en la figura 9, es un motorreductor DC de alto torque, 38 RMP, 3.6 kg/cm, de5 V y con un máximo de 670 mA. Para poder controlar la velocidad del motor, se utilizará un driver o controlador, el L298N. El driver dual para motores (full-bridge) L298N permite controlar dos motores de corriente continua o un motor de paso bipolar de hasta 2 amperes. El módulo cuenta con componentes de protección, un regulador LM7805 para suministrar a la parte lógica 5 V. Cuenta con conectores para habilitar las

salidas del módulo. Internamente el circuito trae 4 puentes h medios (con 2 se controla un motor) y se pueden conectar motores de hasta 35V.

Un aspecto importante del controlador es que se puede controlar la dirección y la velocidad usando el PWM. Con el driver se puede usar hasta 2 motores de hasta 2 amperes por cada motor; si es un motor de menos de 4 pero más de 2 se pueden usar los puentes h conectados en paralelo: 2 conectores para motor A, 2 conectores para motor B.

Los pines de control para los motores son: enable A y enable B que se habilitan para controlar los motores y la velocidad de giro, y los 4 pines de control de los puentes H (IN1 (polarización directa) e IN2 (polarización inversa) para controlar el motor A, IN3 (polarización directa) e IN4 (polarización inversa) para controlar el motor B) para controlar el sentido de giro. Estas salidas llegan a los bordes de los motores.

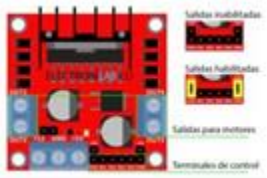


Fig. 9 Actuador

El controlador se puede alimentar de 2 maneras. Cuando el jumper de selección de 5V se encuentra activo, la entrada de alimentación puede ser de 6 a 12 V y la salida será de 5 V, ambos DC. El voltaje se puede usar en cualquier dispositivo de 5V solamente con precaución de corriente. Cuando el jumper de selección de 5V se encuentra inactivo, la alimentación del controlador varía entre 12 y 35V DC porque el regulador no se encuentra funcionando. Debido a esto, para alimentar la parte lógica del controlador, se debe de alimentar con 5VDC el otro conector que usualmente suele ser la misma que la parte de control.

Para controlar la velocidad del motor, tenemos que hacer uso de PWM. Este PWM será aplicado a los pines de activación de cada salida o pines ENA y ENB respectivamente, por lo tanto, los jumpers de selección no serán usados.

Esquema de conexión

La figura 10 muestra el esquema de conexión del actuador.

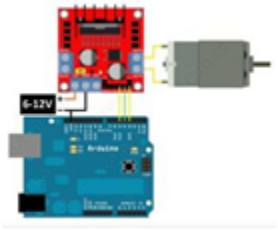


Fig. 10 Esquema de conexión

Timer/Contador en Arduino

El microprocesador ATmega328p cuenta con 2 timers/contadores de 8 bit con su preescalador y el modo de comparación, 1 timer/contador de 16 bit con su prescalador, el modo de comparación y el modo de captura, además del oscilador independiente del microprocesador.

El timer 0 y el timer 2 son timers de 8 bits, lo que significa que el valor máximo que pueden contar es 255. El timer 1 es de 16 bits, lo que significa que el valor máximo que puede almacenar es de 65535. Cuando alcanzan su valor máximo, se desborda y vuelve a 0 activándose la interrupción del timer. A 16 MHz, la interrupción en el timer de 8 bits será cada 256/16000000 (16 micros) segundos y 65536/16000000 (4 milis) segundos para el de 16 bits. El tiempo del reloj del ATmega328p es de 16MHz.

Para no tener un tiempo de desborde muy bajo, se usa el preescalador que lo que hace es “disminuir la frecuencia”, es decir, hace las operaciones con menor velocidad.

Ejemplo, si se quiere el desborde o la interrupción cada segundo (frecuencia de 1 Hz), sería la frecuencia del reloj entre el preescalador *la frecuencia del tiempo – 1 que es el valor tomando el cero. Esto daría 16000000/(1024*1) – 1 = 15624. Como es mayor a 255, se utilizaría el timer 1 de 16 bits.

Cuando el timer se desborda, el bit del registro TIMSKx se active siempre y cuando el bit del registro TOIEx del timer esté activado. Cuando ocurre esto, será llamada una subrutina en el programa ISR(TIMERx_OVF_vect).

Los registros del timer en el microcontrolador son: TCNTx – Registro del timer/Contador en donde se almacena el valor actual; OCRx – Registro de comparación de salida; ICRx – Registro de captura de entrada (sólo para el timer de 16 bit); TIMSKx – Registro del timer/Contador de interrupción de máscara (para habilitar o deshabilitar las interrupciones del timer); TIFRx – Registro del timer/Contador de la bandera de interrupción (indica si hay interrupción pendiente o activada). (x = lugar del timer 0, 1 ó 2; y después el registro A o B).

Para calcular la frecuencia o el tiempo en que se debe activar la interrupción del timer es mediante:

- Frecuencia del microprocesador (16MHz Arduino UNO)
- Máximo valor a contar (256 de 8 bit y 65536 para el timer de 16 bit)
- Dividir la frecuencia del microprocesador entre el preescalador que puede ser 8,64,256,1024.
- El resultado dividirlo entre la frecuencia deseada final y si el valor es menor al máximo valor a contar está correcto y es el valor que se carga. De lo contrario, seleccionar otro preescalador de mayor valor.

Interrupciones

Las interrupciones son acciones que se ejecutan cuando se recibe una señal específica que indica que el programa o lo que se está realizando se interrumpa para hacer otra acción de acuerdo a esa interrupción. Hay interrupciones por timer, contador, externas.

Los pines en Arduino UNO para interrupciones son: Interrupción 0 – pin 2; Interrupción 1 - pin 3. La figura 11 muestra un diagrama de conexiones para una interrupción.

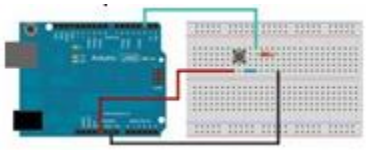


Fig. 11 Diagrama de conexiones para una interrupción

Fuente de poder para brazo robótico

Se revisó la hoja de datos de la fuente de poder OKL-T/3-W12N-C (el overview de la misma se muestra en anexos). Aunque es una unidad integrada programable que parece sencilla de usar se deben de hacer varias observaciones para el uso óptimo de la misma.

El esquemático de conexiones se muestra en la Figura 12.

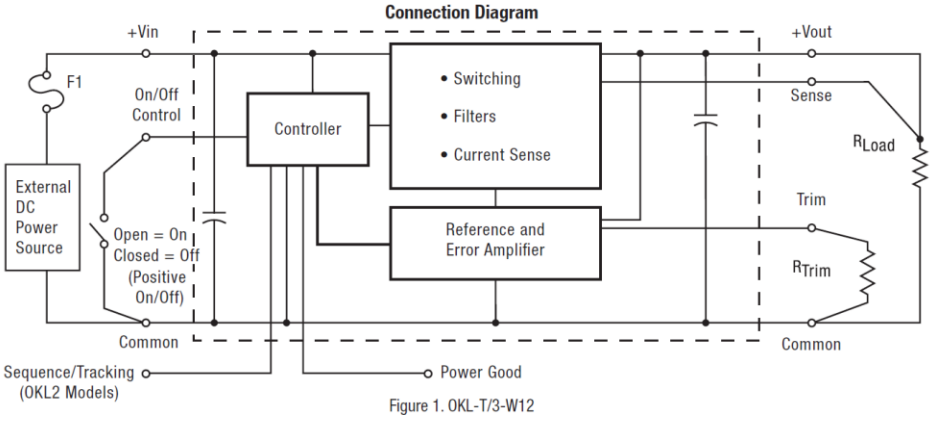


Fig. 12 Esquema de conexiones

La grafica de eficiencia para un voltaje de salida de 5V se muestra en la Figura 13.

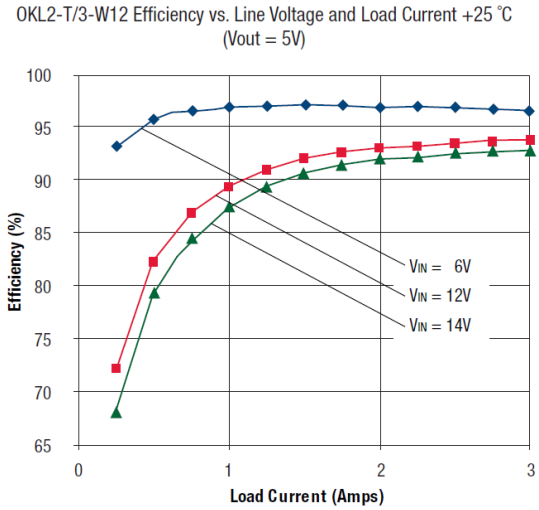


Fig. 13 Gráfica de eficiencia

Algunas observaciones son:

- Start-Up Voltage de 4.2 V.
- Undervoltage Shutdown 3.4 V.
- Voltaje máximo de entrada de 15 V.
- Se recomienda el uso de un fusible externo de 4 A.
- La unidad tiene un control remoto ON/OFF negativo por lo que el pin de On/Off Control debe dejarse abierto.
- Salida máxima de poder 16.5W
- El voltaje puede ser ajustado por un resistor externo (Rtrim). Se sugiere que se suelde el resistor lo más cercano al convertidor sin patas largas o un resistor tipo trim SMD. La salida de voltaje respecto a la resistencia se muestra en la Figura 14.

Output Voltage	Calculated Rtrim (KΩ)
5.0 V.	1.34
3.3 V.	2.18
2.5 V.	3.1
2.0 V.	4.19
1.8 V.	4.88
1.5 V.	6.50
1.2 V.	9.70
1.0 V.	14.45
0.591 V.	∞ (open)

Fig. 14 Salida de voltaje respecto a resistencia

Además, se recomienda:

- Instalar un capacitor en las terminales de entrada. El capacitor debe de ser cerámico como el Murata GRM32. Con valores entre 10 a 22µF con un rango de voltaje doble al voltaje de entrada.
- Instalar un capacitor en la salida para evitar picos o respuestas dinámicas imprevistas. Se recomienda usar un capacitor de la serie Murata GRM32 con valores entre 10 a 47 µF.

El módulo tiene un pin “Power Good Output” donde se indica cuando el convertidor se ha estabilizado y la salida esta aproximadamente en regulación.

El esquemático hecho en el software Eagle de la fuente de poder usando el módulo OKL-T/3-W12N-C se muestra en la Figura 15.

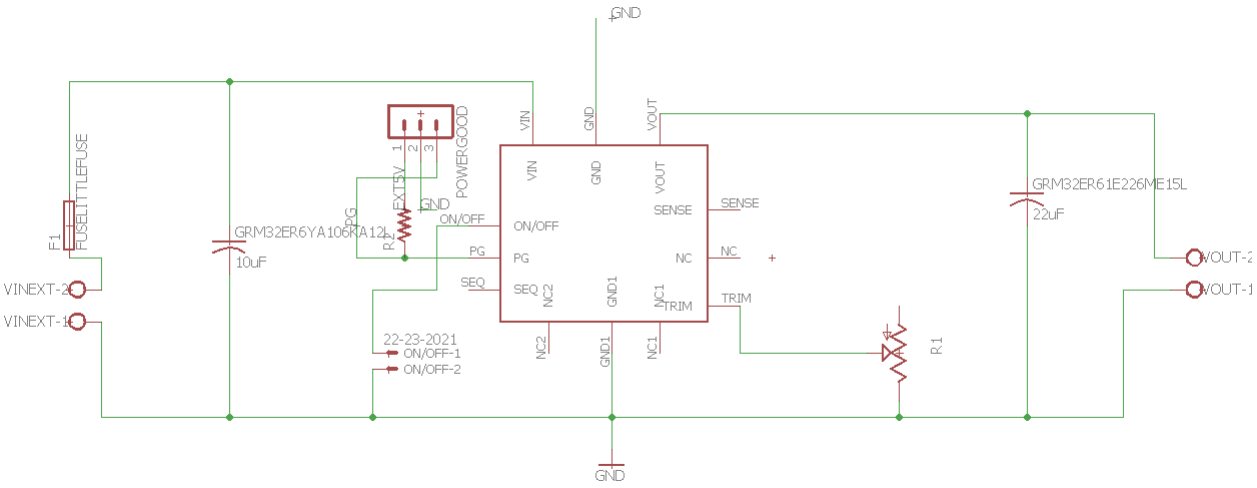


Fig. 15 Esquemático para fuente de poder del brazo robótico

Y la vista de circuito impreso a una sola cara se muestra en la Figura 16.

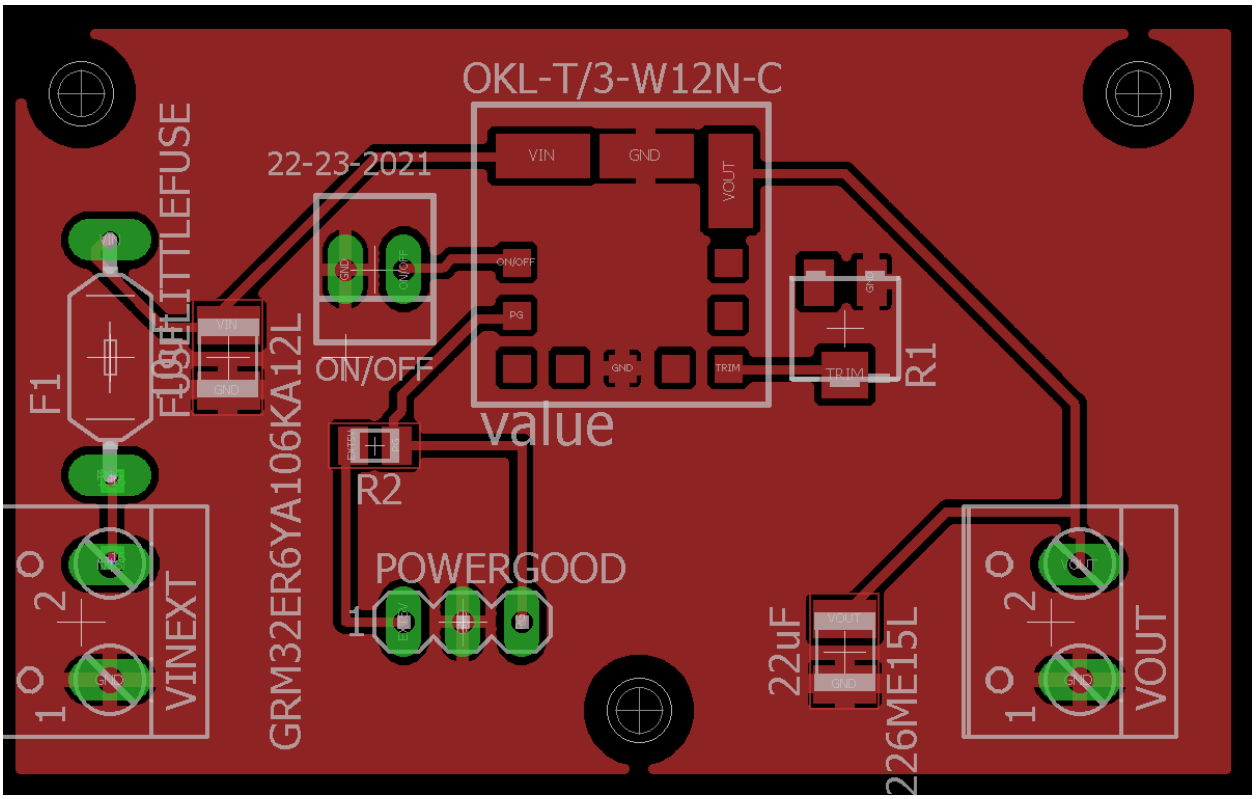


Fig. 16 Vista top de PCB

Como se puede ver en la Figura en la entrada y salida se tienen conectores tipo screw, esto para interconectar la fuente de voltaje y la tarjeta para los servomotores.

Circuito para servomotores

Para los servomotores se usarán los pines marcados con el número 3, 5, 6, 9, 11 y 13. La vista en protoboard se muestra en la Figura 17. La vista de arriba del circuito para las conexiones de los motores se muestra en la Figura 18.

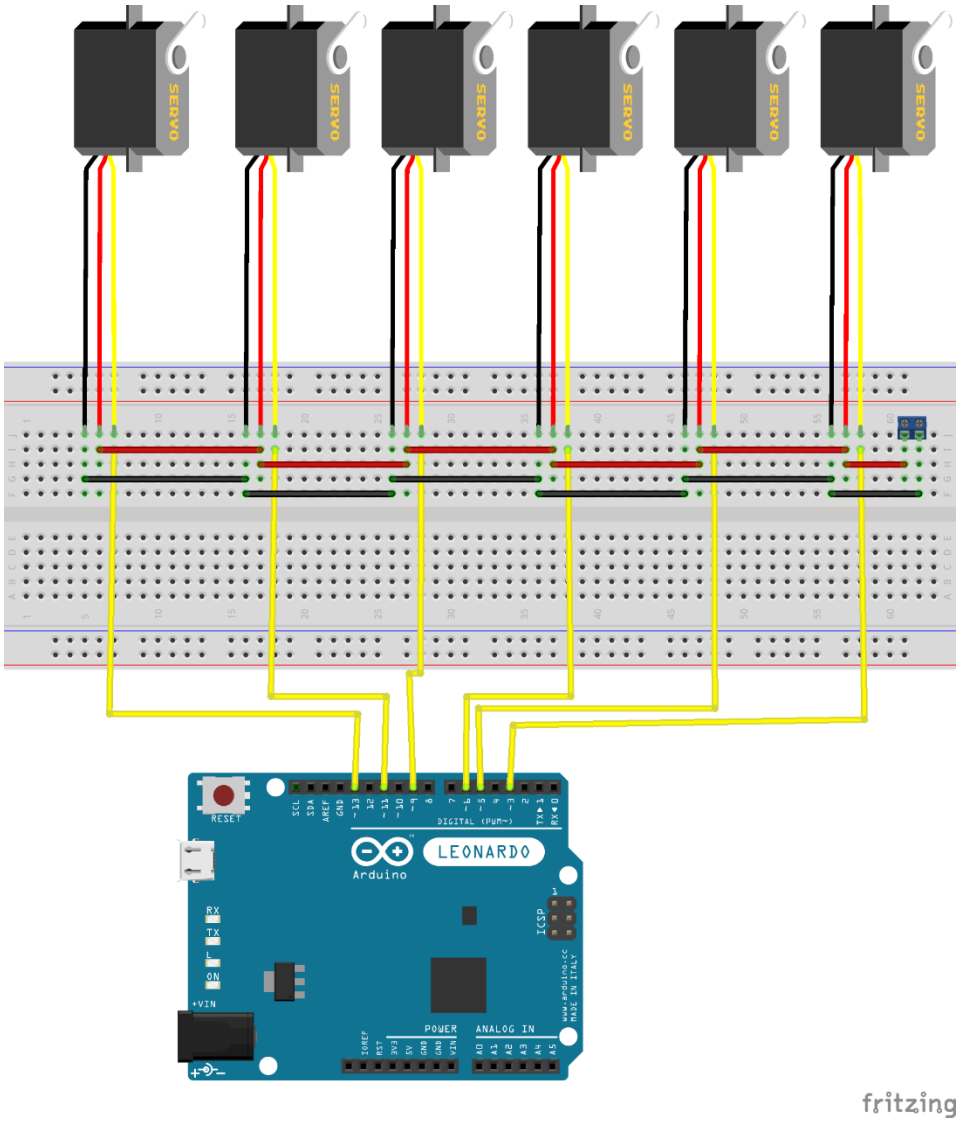


Fig. 17 Vista en protoboard para conexiones de servo motores

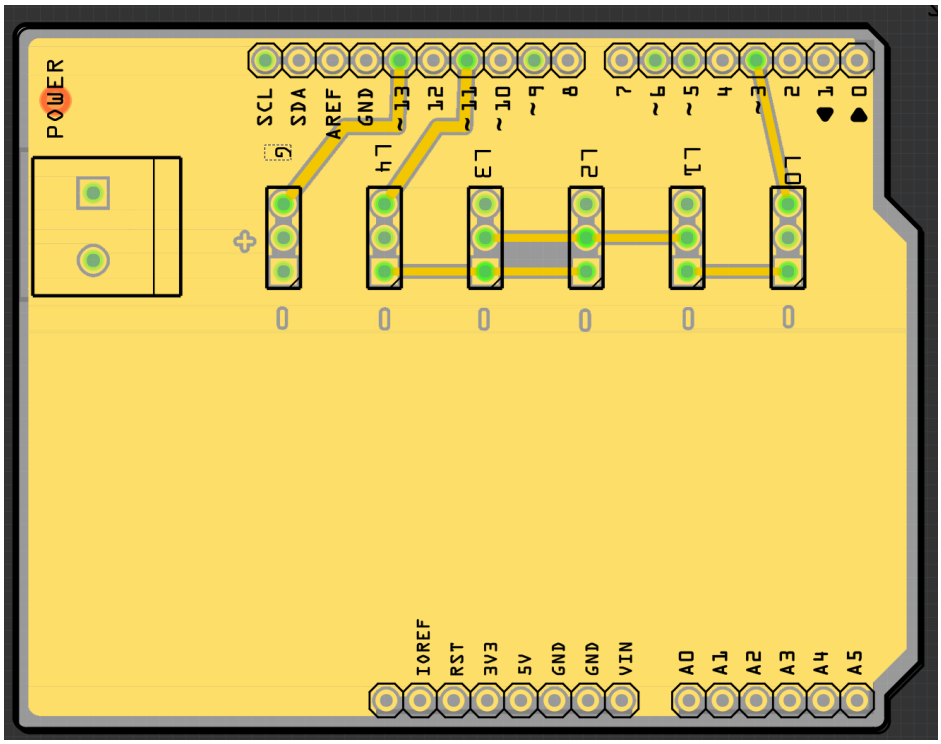


Fig. 18 Vista top para circuito de conexiones para servo motores

Y vista de debajo del circuito para servo motores se muestra en la Figura 19.

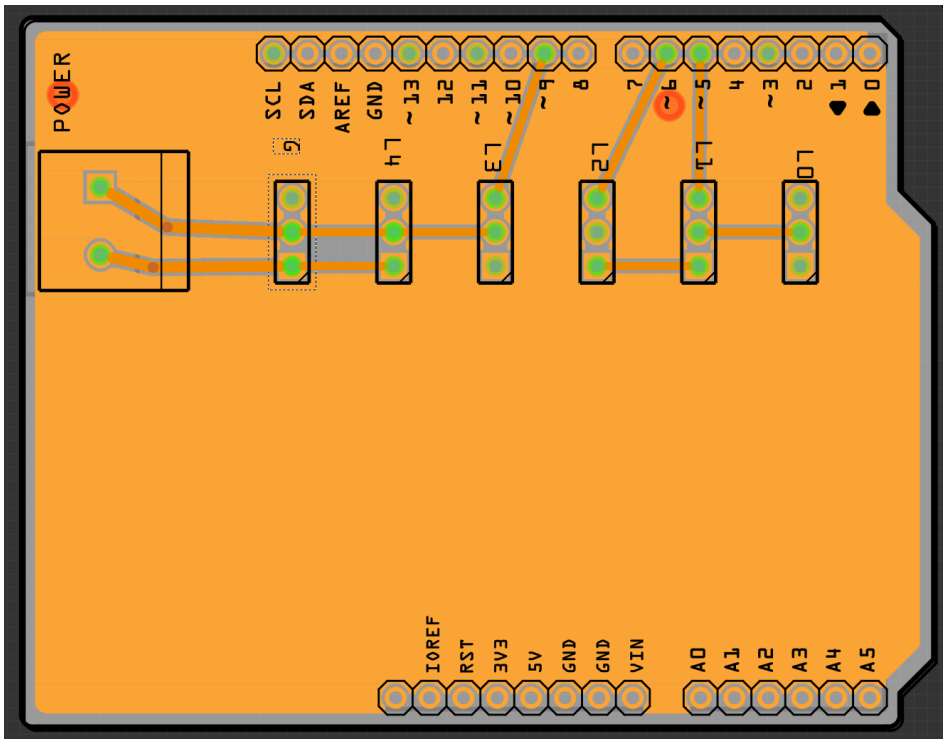


Fig. 19 Vista bottom para circuito de conexiones para servo motores

Interface

La figura 20 muestra la interface que se utilizará en el panel de control.

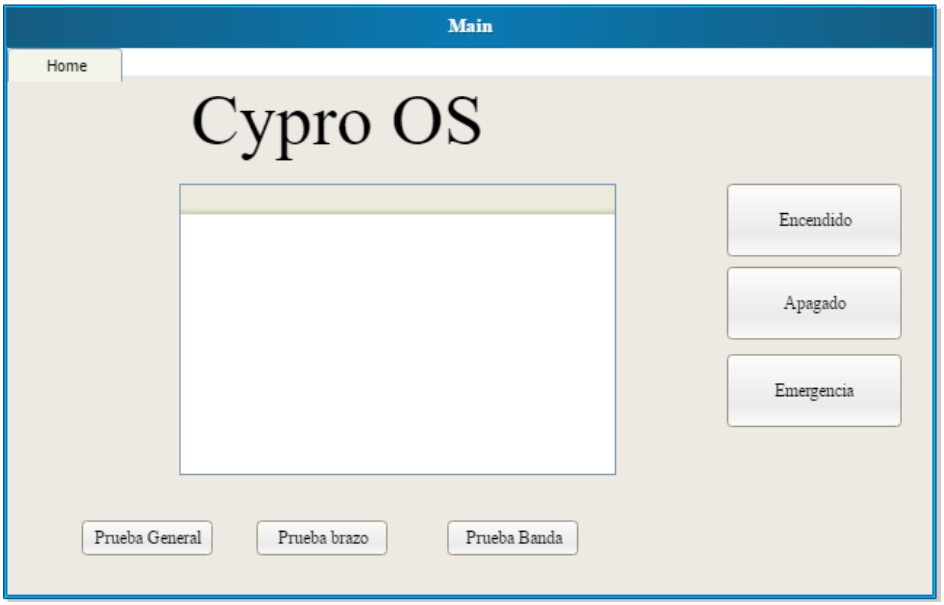


Fig. 20 Wireframe general

Pseudocódigo

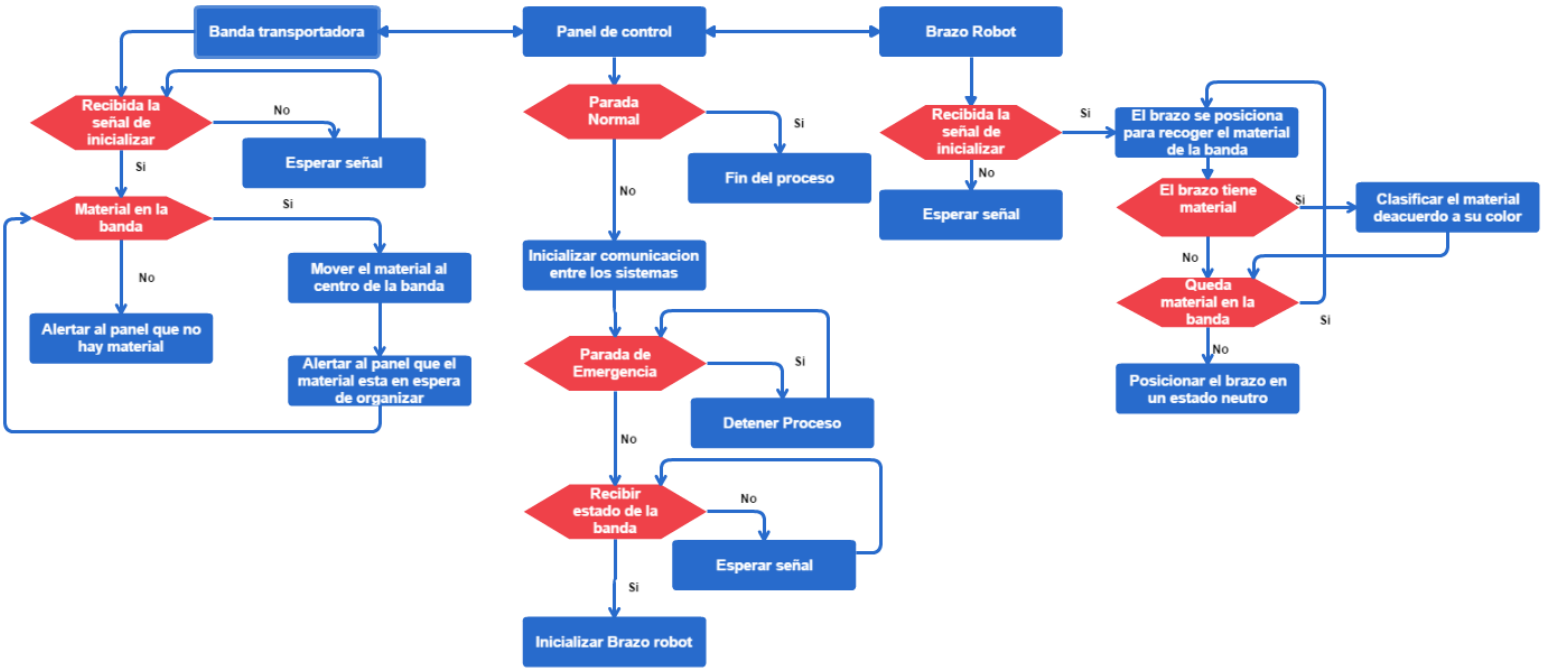


Fig. 21 Diagrama de flujo del pseudocódigo.

En el pseudocódigo, mostrado en la figura 21, se puede ver de manera clara la forma en la que se ejercen las operaciones del sistema compuesto por un panel de control, una banda transportadora y el brazo robot, los cuales están enlazados por los diferentes medios de comunicaciones entre los controladores que cada uno posee.

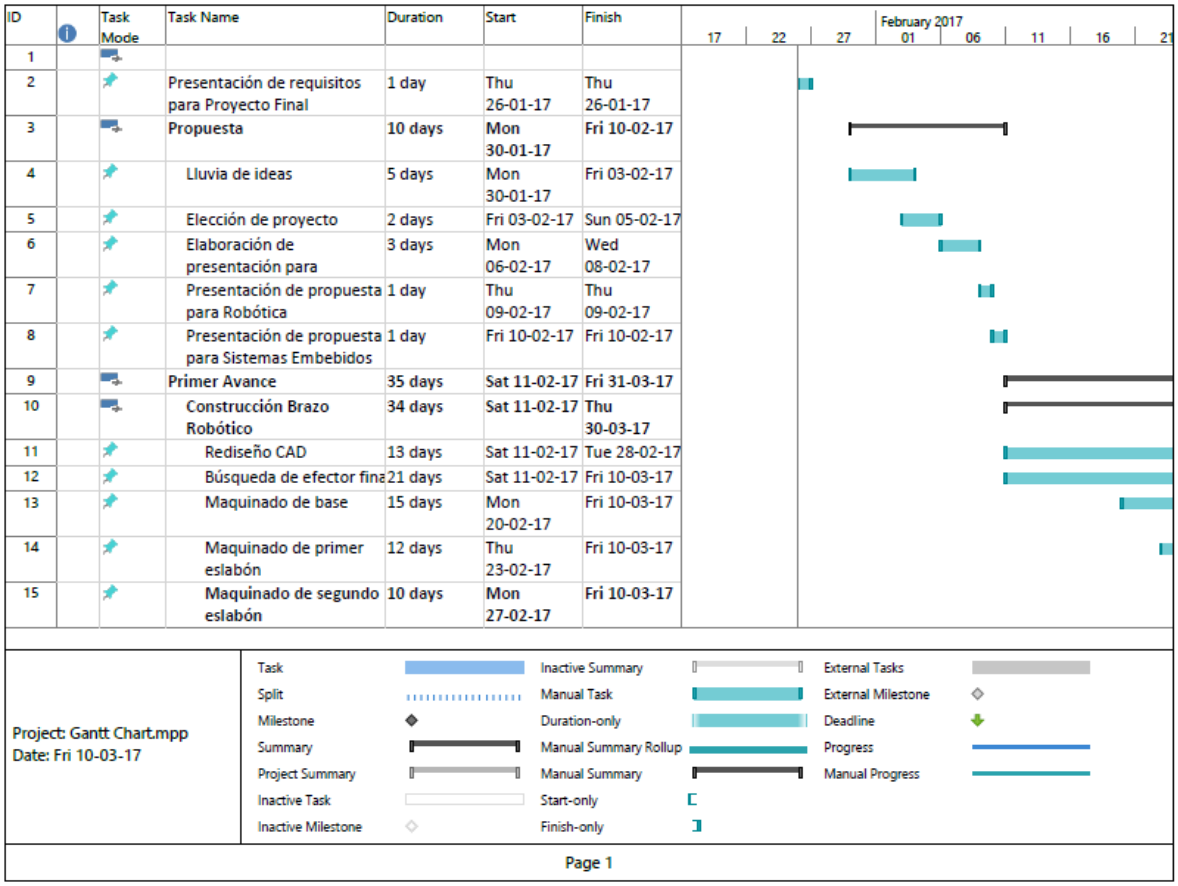
Materiales

Material	Características	Objetivo
Arduino Leonardo ATmega32u4	• Microcontrolador CMOS de 8 bits basado en AVR. • Emplea arquitectura RISC. • El núcleo AVR combina un set de instrucciones con 32 registros de uso general. • Registros conectados directamente a la ALU, permitiendo acceder a 2 registros de manera independiente. • Alcanza velocidades 10 veces más rápido que CISC. • 32K bytes de In-System Programmable Flash de los cuales 4 KB son usados para bootloader, 1 KB EEPROM, 2.5KB de SRAM. • 26 salidas de propósito general	El brazo robótico será servocontrolado (PWM) por el ATmega32u4.

	○ 20 están disponibles para el Arduino Leonardo. ○ 7 son canales PWM. ○ 12 son canales para entrada análoga • Incluye un USART, Interfaz Serial 2-wire (I2C), puerto serial SPI.	
Arduino Uno ATmega 328P	• 32K bytes de In-System Programmable ○ 0.5 KB son usados para bootloader ○ 1 KB EEPROM ○ 2 KB de SRAM • Una velocidad de reloj de 16 MHz • 14 pines para entradas y salidas digitales ○ 6 pueden usarse como salida PWM ○ 6 entradas análogas ○ 8 canales ADC. • Incluye un USART, Interfaz Serial 2-wire (I2C), puerto serial SPI.	Controlará la banda transportadora.
Raspberry Pi 1 Mod B Broadcom BCM2835	• 512 MB en la memoria RAM, dos puertos para conexión USB	Controlará el panel de control.

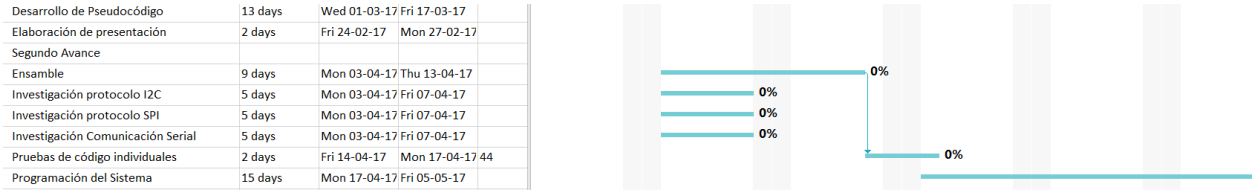
	• Comunicación mediante Ethernet • 40 canales para entradas y salidas • Incluye un UART, Interfaz Serial 2-wire (I2C), puerto serial SPI, PWM y USB.	
Sensor SR501	• Está basado en tecnología infrarroja capaz de detectar objetos en un rango de cono de 110 grados.	Es usado como dispositivo de seguridad en nuestra celda de manufactura, pues al detectar algo irregular según los parámetros del panel del control mandaría una interrupción al brazo robótico y a la banda.
Sensor Sharp GP2Y0A21YK	• Mide distancia • Tiene un rango de 10 a 80 cm • El tiempo de respuesta es de 39ms • El consumo de corriente es de 30mA	Detectará la entrada de objetos en la banda transportadora
Sensor CNY70	• Sensor reflectivo • Que incluye un emisor infrarrojo y un fototransistor empaquetado para bloqueo de luz visible	Se usará en el efector final del brazo robótico y tendrá la función de diferenciar entre blanco y negro para posteriormente hacer una clasificación de acuerdo al color.

Pendientes



ID	Task Mode	Task Name	Duration	Start	Finish
1					
2		Presentación de requisitos para Proyecto Final	1 day	Thu 26-01-17	Thu 26-01-17
3		Propuesta	10 days	Mon 30-01-17	Fri 10-02-17
4		Lluvia de ideas	5 days	Mon 30-01-17	Fri 03-02-17
5		Elección de proyecto	2 days	Fri 03-02-17	Sun 05-02-17
6		Elaboración de presentación para	3 days	Mon 06-02-17	Wed 08-02-17
7		Presentación de propuesta para Robótica	1 day	Thu 09-02-17	Thu 09-02-17
8		Presentación de propuesta para Sistemas Embebidos	1 day	Fri 10-02-17	Fri 10-02-17
9		Primer Avance	35 days	Sat 11-02-17	Fri 31-03-17
10		Construcción Brazo Robótico	34 days	Sat 11-02-17	Thu 30-03-17
11		Rediseño CAD	13 days	Sat 11-02-17	Tue 28-02-17
12		Búsqueda de efector fina	21 days	Sat 11-02-17	Fri 10-03-17
13		Maquinado de base	15 days	Mon 20-02-17	Fri 10-03-17
14		Maquinado de primer eslabón	12 days	Thu 23-02-17	Fri 10-03-17
15		Maquinado de segundo eslabón	10 days	Mon 27-02-17	Fri 10-03-17

Las tareas por hacer se muestran en la Figura 22. Aunque ya se han hecho pruebas de comunicación entre microcontroladores no se ha hecho orientado al proyecto. Una vez hecho el proyecto físico se implementarán los protocolos en conjunto simulando así la situación final de aplicación.



Desarrollo de Pseudocódigo	13 days	Wed 01-03-17	Fri 17-03-17
Elaboración de presentación	2 days	Fri 24-02-17	Mon 27-02-17
Segundo Avance			
Ensamble	9 days	Mon 03-04-17	Thu 13-04-17
Investigación protocolo I2C	5 days	Mon 03-04-17	Fri 07-04-17
Investigación protocolo SPI	5 days	Mon 03-04-17	Fri 07-04-17
Investigación Comunicación Serial	5 days	Mon 03-04-17	Fri 07-04-17
Pruebas de código individuales	2 days	Fri 14-04-17	Mon 17-04-17 44
Programación del Sistema	15 days	Mon 17-04-17	Fri 05-05-17

Fig. 22 Tareas por hacer desde el enfoque de sistemas embebidos

Conclusiones

Para nuestra entrega de proyecto en la semana 13-17 nos enfocamos en llevar a cabo el desarrollo gráfico de nuestro panel de control, el cual será controlado por una Raspberry pi 2. Para la creación de esta interface, se propone usar el lenguaje Python, ya que se encuentra integrado de manera estándar en nuestra tarjeta. Como propuesta general de este esquema, se presentan 3 botones con funciones muy simples que desarrollara nuestro sistema. El botón de encendido, como su nombre lo infiere, encenderá el sistema desde una posición inicial ya establecida, por lo que se dará inicio a un suceso de instrucciones para llegar a esa posición i en cada uno de nuestros módulos (brazo robótico, banda trasportadora).

El botón de apagado dará un apagado normal para cada uno de los módulos usados en el sistema. Este conlleva a llevar a cada uno de ellos a su posición ordinaria, y estar lista para su funcionamiento.

En cuanto a nuestro botón de emergencia, se implementará un apagado de emergencia, en donde simplemente se hará un apagado de todos los módulos en su posiciones actuales y no habrá movimiento alguno, hasta que se presione de nueva cuenta y vuelta seguir con la secuencia normal del sistema.

Toda la información del estado actual de cada uno de los módulos será desplegada en nuestra caja central, al igual que cada uno de los posibles errores que desplegaría, o pruebas a implementarse, por parte de nosotros o el sistema.

Referencias

Kothari D., Vasudevan S., Sundaram R., Murali N. (2012). Embedded Systems. New Age International Ltd., Publishers: New Delhi

Arduino - SPI. (n.d.). Retrieved March 10, 2017, from https://www.arduino.cc/en/reference/SPI

I2C Info – I2C Bus, Interface and Protocol. (n.d.). Retrieved March 10, 2017, from http://i2c.info/

Mitchell, B. (n.d.). A USB port is one of the most useful features on computers and phones. Retrieved March 10, 2017, from https://www.lifewire.com/what-is-a-usb-port-818166

Código de Honor

Nosotros, Didier, Hugo, Teresa, Víctor y William, declaramos que hemos realizado este reporte con estricto apego al Código de Honor de la Udem.