Desarrollo de un sistema embebido de bajo costo para fines educativos

K. A. Pichardo Rivas[a], I. I. Fernández Morales[a], I. A. Arriaga Trejo[b], J. Flores Troncoso[a], S. Ibarra Delgado[a], R. Sandoval Aréchiga[a], J. Villanueva Maldonado[b], J. R. Gomez-Rodriguez[a]

[a]*Centro de Investigación y Desarrollo en Telecomunicaciones Espaciales (CIDTE), Universidad Autónoma de Zacatecas, Unidad Académica de Ingeniería Eléctrica.*
Av. López Velarde No. 801, Zacatecas, Zac, 98000, México.
http://www.uaz.edu.mx/
[b]*Cátedra CONACyT, CIDTE, Universidad Autónoma de Zacatecas, Unidad Académica de Ingeniería Eléctrica.*
Av. López Velarde No. 801, Zacatecas, Zac, 98000, México.
http://cidte.uaz.edu.mx/

2017 Published by $\mathcal{DIFU}_{100}ci@$ *http://difu100cia.uaz.edu.mx*

Resumen

En el documento presente se aborda la construcción de un sistema embebido de bajo costo, con dimensiones de 8cm de ancho por 7.5cm de longitud. El sistema propuesto utiliza un microcontrolador ATMEGA328P, con 23 líneas de entrada/salida de propósito general, admite programas por medio de ICSP y USB serial, siendo compatible con el software Arduino IDE mientras se tenga en la memoria del microcontrolador el bootloader. Se utilizó un programador USBASP y el programador USB serial uUSB-PA5 para cargarle los programas. Además, presenta una etapa de control y regulación de energía que le permite alimentarse por medio de una entrada USB o barrel Jack, siendo posible conectar ambas fuentes de manera simultánea sin conflicto con la suma de corrientes.

Palabras clave: Sistema embebido, microcontrolador, programación.

1. Introducción

Los sistemas embebidos son cada vez más utilizados en los aparatos electrónicos debido a las diversas aplicaciones que pueden generarse con ellos. Una persona promedio cuenta con 30 o más sistemas embebidos distribuidos en todos los aparatos en su hogar. Por lo regular están hechos para realizar una sola tarea, la cual ejecuta de manera permanente, en un circuito sencillo [1].

Debido a las necesidades de aplicaciones las cuales consumen mayor energía, se ha aumentado constantemente la exigencia del hardware en el cual se ejecutan tales aplicaciones. Esto ha permitido el advenimiento de sistemas embebidos y hardware cada vez más robusto y reducido en masa y volumen [2]. Los sistemas embebidos son utilizados por universidades debido a que con ellos se implementan procesos o aplicaciones específicas para monitoreo y manipulación de señales, promoviendo también el desarrollo de aplicaciones con fines educativos.

Un sistema embebido encapsula en un dispositivo todo el hardware y software requerido para implementar un sistema con propósitos específicos. Están orientados

a resolver problemas en tiempo real y cuentan con los recursos necesarios para realizar su objetivo [3]. Los sistemas embebidos trabajan con un firmware, que es el conjunto de instrucciones de programa grabado en una memoria tipo no volátil (ROM, EEPROM, flash, etc.) controlando los circuitos electrónicos para el propósito específico [4].

En el caso de sistemas embebidos que usan microcontroladores, en la memoria existe un firmware que da las instrucciones al microcontrolador para que cumpla su función o tarea asiganada. Para escribir el firmware se hace uso de un programador que funciona como el nexo entre la computadora y el sistema embebido, algunos ejemplos de programadores son el USBASP, USBTiny, USB serial, etc. Los programadadores seriales necesitan que en el microcontrolador haya un bootloader, un código que se ejecuta al momento de reiniciar el sistema, normalmente colocado en una parte de la memoria flash que esté protegido contra el borrado al momento de reiniciar el sistema [5].

2. Descripción del sistema embebido

El diseño que se detalla en este artículo sienta las bases para desarrollos posteriores. Por ejemplo, sirve para el diseño de una computadora a bordo para picosatélites con fines educativos.

El sistema embebido a realizar tiene como objetivo contar con entradas y/o salidas analógicas, digitales y de modulación por ancho de pulso, con un diseño de bajo costo y de dimensiones reducidas, programable mediante un puerto ICSP o un convertidor USB-Serial, igualmente se busca que acepte la entrada de programas tipo HEX.

Se tomó como referencia para el desarrollo, el sistema Arduino UNO [6], debido a que cumple con los objetivos de diseño planteados. Con el fin de representar de manera general el funcionamiento del sistema embebido desarrollado, éste se dividió en tres etapas. En la Figura 1 se muestra cada una de las etapas del sistema embebido con sus principales elementos.

2.1. Desarrollo

La etapa de regulación de energía se encarga de suministrar de energía a todo el sistema, utiliza un conector barrel Jack, que admite una entrada de 7V a 25V, un diodo de protección que evita el daño de componentes en caso de polarización inverza en el barrel jack y un regulador de voltaje a 5V que es el encargado de alimentar la etapa del microcontrolador. Se tiene también

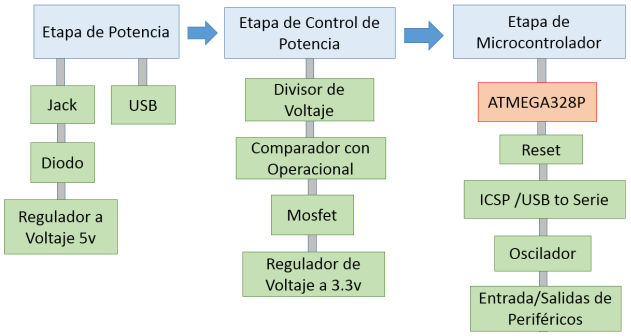

Figura 1. Etapas del funcionamiento del sistema embebido.

la opción de alimentar el sistema por medio de un USB, suminstrando con 5V de manera directa.

Debido a que se pueden tener dos distintas fuentes de voltaje alimentando el sistema de manera simultánea, se generaría un problema al momento de conectarlas, la corriente generada podría llegar a dañar el sistema o a alguna de las fuentes de alimentación. La etapa de control de energía es la encargada de suprimir la corriente de la entrada USB cuando el barrel jack está conectado. Por lo tanto, podemos tener tres casos distintos:

- Cuando el barrel Jack está desconectado y el USB conectado.
- Cuando tanto el barrel Jack y el USB están conectados.
- Cuando el barrel Jack está conectado y el USB desconectado.

En el primer caso, el diodo parásito del mosfet de canal P permite el flujo de corriente de drenador a surtidor, por lo que el USB alimenta el circuito completo.

En el segundo caso, un divisor de voltaje reduce a la mitad la tensión suministrada por el barrel jack, la salida pasa hacia un amplificador operacional que actúa como comparador de voltaje, y debido a que el nivel de voltaje será mayor a 3.3V, en la salida del comparador habrán 5V, por lo tanto el mosfet se polarizará e impedirá el paso de corriente de surtidor a drenador, protegiendo así el puerto USB de la corriente de salida del regulador de 5V. Por otro lado la corriente que proviene del USB no pasará a través del mosfet debido a que para que el diodo parásito permita conducción el voltaje en el drenador tiene que ser mayor al del surtidor, dejando así, ambas fuentes completamente aisladas.

En el tercer caso el barrel Jack es el que alimenta el circuito completo, el diodo del mosfet evita el paso de corriente hacia el USB evitando que se pueda dañar el puerto.

La etapa del microcontrolador se encarga del funcionamiento de la tarea determinada, almacenada en la parte de la memoria flash del microcontrolador, donde se decidió utilizar el ATMEGA328P para que el sistema fuera compatible con el software Arduino IDE. Existe un botón de reset que tiene la función de reiniciar el programa almacenado. Se tienen dos maneras de escribir un programa en el microcontrolador: la primera es por medio de los pines ICSP (In Circuit Serial Programming), mientras que la segunda es a través de los pines Rx y Tx, haciendo uso de un convertidor USB-Serial. Para poder usar el Arduino IDE es necesario que el microcontrolador tenga el bootloader de Arduino en la memoria. En caso de que se suba algún otro programa por medio del ISCP, el bootloader de Arduino será borrado y será necesario realizar el proceso de escritura del bootloader. El circuito contiene un oscilador de cristal de 16MHz para que los pulsos generados por el microcontrolador sean más estables y fiables, en algunas aplicaciones se trabaja a grandes frecuencias o la precisión que se ocupa es mayor.

Se realizó una simulación de la etapa de regulación de energía y control de energía para probar que funciona correctamente,verificando que las dos fuentes de alimentación trabajan correctamente y no hay conflicto al momento de estar ambas activas. En la Figura 2 se muestra el comportamiento de la etapa de regulación y control de energía cuando el barrel Jack y el puerto USB están conectados, observándose que el diodo parásito del mosfet evita el paso de corriente del barrel Jack hacia el puerto USB.

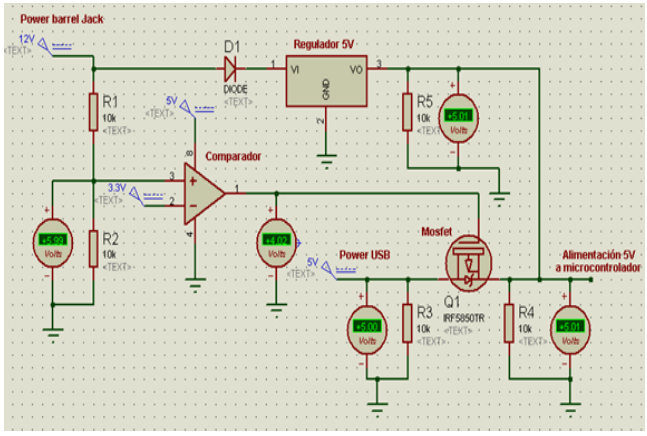

Figura 2. Etapa de regulación y control de energía con la fuente de alimentación del barrel Jack y entrada USB conectada.

En la Figura 3 se muestra el comportamiento de la etapa de regulación y control de energía cuando el puerto USB está conectado, siendo alimentado todo el sistema por el puerto USB.

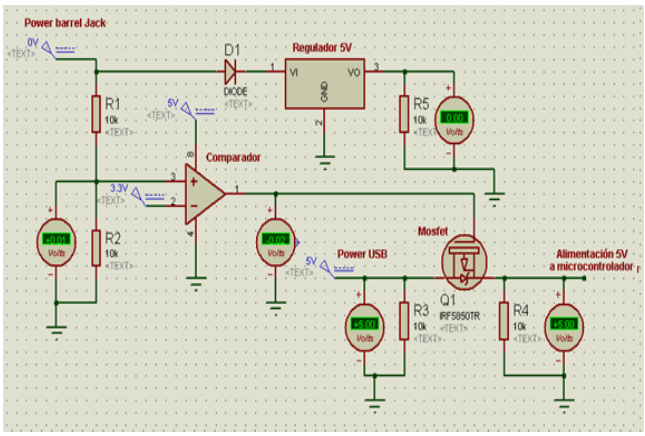

Figura 3. Etapa de regulación y control de energía con la fuente de alimentación USB conectada.

Posterior a la simulación se realizó el esquemático de todo el sistema con el software Kicad. El diagrama electrónico de la etapa de regulación y control de energía se muestra en la Figura 4, la etapa del microcontrolador se muestra en la Figura 5.

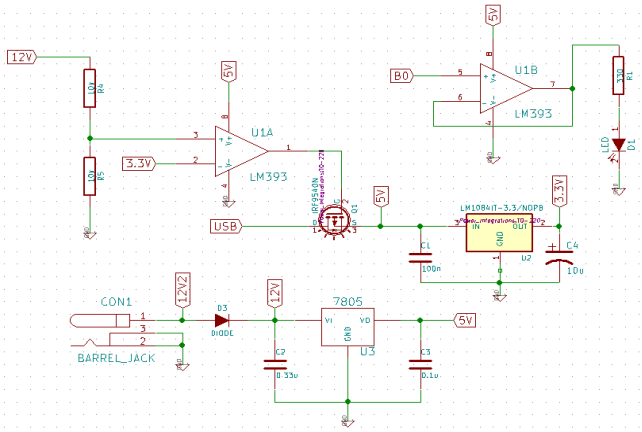

Figura 4. Esquema en Kicad de la etapa de regulación y control de energía del sistema embedido.

El diseño final de la placa se muestra en la Figura 6.

Se colocó una tira de pines con distintas salidas de voltaje en caso de que se desee utilizar para alimentar algún componente, la tira tiene salida de GND, 3.3V, 5V y el voltaje de entrada que se utilice en el barrel Jack en caso de que esté conectado.

Se usó un microcontrolador ATMEGA328P sin bootloader en el sistema embebido y haciendo uso del progamador USBASP se le cargó un programa con el software ATMEL STUDIO 7.0 que controla el estado intermitente del led de prueba. Posteriormente se le cargó el bootloader de Arduino para poder usar el software de Arduino IDE.

Todos los microcontroladores tienen una firma (signa-

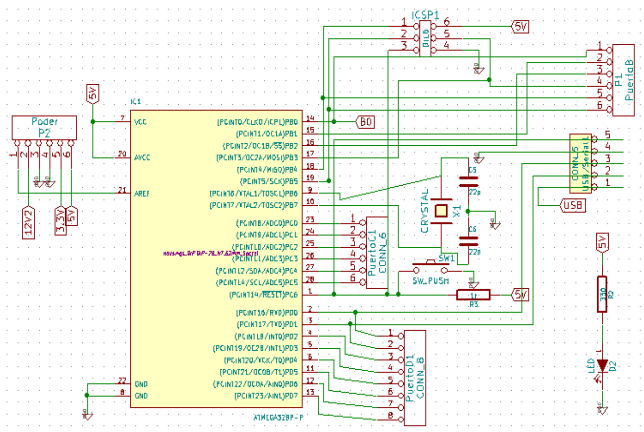

Figura 5. Esquema en Kicad de la etapa del microcontrolador del sistema embedido.

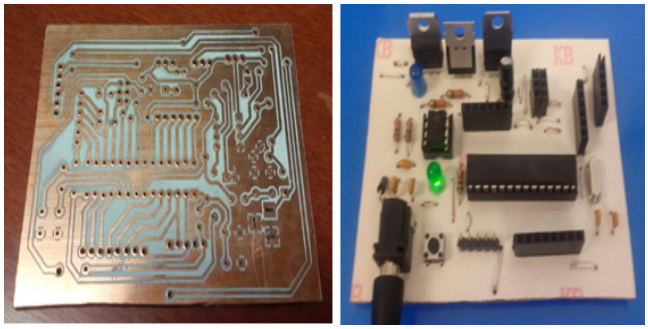

Figura 6. PCB del sistema embedido antes y después de soldar los componentes.

ture), que es un código único que los identifica a cada uno, los ATMEGA328P que se consiguen de fábrica tienen la firma 0x1e 0x95 0x14, sin embargo, los ATMEGA328P que poseen los Arduino UNO tienen la firma de 0x1e 0x95 0x0F. Para cargar el bootloader a los microcontroladores de fábrica hay que cambiar la firma que espera el Arduino IDE por la de 0x1e 0x95 0x14, para esto se tiene que modificar el archivo avrdude.conf desde la liga C:/Arduino/hardware/tools/avr/etc/avrdude.conf, una vez que se le haya cargado el bootloader, será necesario regresar el archivo avrdude.conf a su estado original para cargarle programas.

El programador USB serial que se usó fue un uUSB-PA5 [7], por medio de los pines Tx y Rx del microcontrolador se le mandó desde Arduino IDE un programa de prueba que al igual que con el USBASP controlaba el encendido y apagado de un led. A la hora de cargar el programa de esta manera es necesario presionar el botón de reset justo después de que el Arduino IDE compila el programa. La Figura 7 muestra la conexión del programador USBASP y el USB serial.

El ATMEGA328P tiene 23 pines para propósito gene-

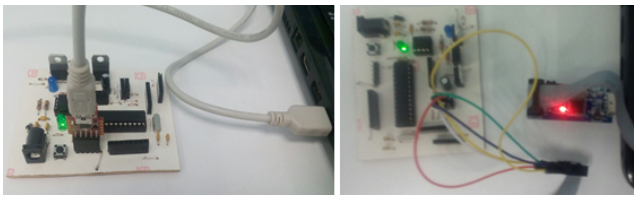

Figura 7. Sistema embebido conectado al programador USBASP y USB serial.

ral, en este caso para probar las entradas y salidas del sistema se le colocó un acelerómetro ADXL345.

3. Resultados

El sistema embebido funcionó correctamente al cargarle programas por medio del USBASP como por el USB-serial, tanto con el software ATMEL STUDIO 7.0 como con Arduino IDE, logrando controlar el encendido y apagado del led de prueba del sistema.

Para probar el funcionamiento del acelerómetro ADXL345 se usó Arduino IDE para cargarle el programa y por medio del monitor serie se observó el ángulo respecto al eje X y Y del acelerómetro. En la figura 8 se muestran los registros del acelerómetro.

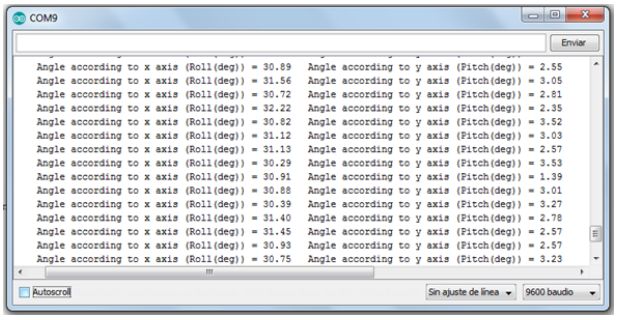

Figura 8. Ángulos respecto al eje X y Y mostrados en el monitor serie de Arduino IDE del acelerómetro conectado al sistema embebido.

3.1. Costos y materiales

Para el sistema embebido desarrollado se utilizaron los siguientes componentes mostradas en la Tabla.

Componente	Cantidad	Precio
Diodo 1N4007	1	$2.20
Oscilador de crystal de 16MHz	1	$13.20
Conector barrel Jack	1	$12
Socket de 28 pines	1	$6.60
Mosfet STP10P6F6	1	$29.80
Regulador a 5V 7805	1	$12.40
Regulador a 3.3V LD1117V33C	1	$11.80
Op-amp LM358	1	$5
Resistencias	5	$10
Capacitores cerámicos	5	$18
Capacitor electrolítico	1	$5
Push buttom	1	$2.50
Leds	2	$6
ATMEGA328P	1	$66
Socket de 8 pines	1	$4
Tira de 40 pines mancho	1	$7
Tira de 40 pines hembra	1	$20
Placa de cobre de 10x10cm	1	$10.50
	Total	**$ 242**

Tabla 1. Lista de materiales y precios del sistema embebido desarrollado. Precios en moneda nacional mexicana.

Haciendo una comparación con otros sistemas embebidos equivalentes del mercado, se encontró que tienen un costo de aproximadamente $440 pesos [8].

3.2. Diferencias con Arduino UNO

El sistema que se desarrolló tiene la posibilidad de ser compatible con la plataforma Arduino IDE, lo único necesario es cargarle el bootloader de Arduino desde el puerto ICSP al microcontrolador, sin embargo, si no se desea trabajar con la plataforma Arduino IDE, se puede trabajar con otras plataformas compatibles con los AVR, como la plataforma Atmel Studio o mikroBasic PRO for AVR.

La diferencia en el hardware radica principalmente en que el acabado de la tarjeta no es profesional y los componentes utilizados no son de montaje superficial, lo que abarata el costo total. La tarjeta desarrollada no incorpora un convertidor USB-Serial, como en el caso del Arduino IDE [6].

4. Conclusiones

Se realizó un sistema embebido de bajo costo y de propósito general. Este se puede programar por medio de ICSP y USB serial, tiene entradas y salidas analógicas y digitales para conectar los periféricos que se deseen. Se puede alimentar el sistema por medio de un conector barrel Jack, una entrada USB o alguna batería que entregue 5V de corriente directa, permitiendo también usar de manera simultanea estas fuentes sin posibilidad de dañar el sistema por la suma de corrientes generada. Tiene un led que indica el encendido del sistema y un led de prueba. Se verificó que el sistema desarrollado funciona conforme a lo esperado.

Referencias

[1] Catsoullis J., "Designing Embedded Hardware" *Oreilly*, 2005, pp. 21-50.

[2] Vicente, C., Toledo, A., Fernández, C., & Sánchez, P., "Generación Automática de Aplicaciones Mixtas Sw/Hw mediante la Integración de Componentes COTS". *IEEE Latin America Transactions*, vol. 4, no. 2, 2006.

[3] Sánchez Dams, Rubén Darío., Controlador lógico programable. Una mirada interna. *Editorial Universitaria de la Costa. EDUCOSTA*,ed. 1, 2009.

[4] Sánchez Dams, Ruben Darío. "State of the art of embedded systems from an integrated perspective between hardware and software". *Revista Colombiana de Tecnologías de Avanzada*, vol. 2, no. 22, 2013.

[5] Anton Otilia, Gelineau Brice, & Sauget Jérémy. "Firmware and bootloader". Disponible en: https://rose.telecom-paristech.fr/2012/wp-content/uploads/2012/03/Firmwares-and-bootloaders.pdf, 2012.

[6] Disponible en: https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf

[7] Disponible en: http://www.mouser.com/ds/2/451/uUSB-PA5-Datasheet-REV1-472126.pdf

[8] Disponible en: https://store.arduino.cc/product/A000066