Creating an Autonomous Car with the Stellaris® LM3S316 Microcontroller

Application Note
Copyright

Copyright © 2006–2009 Texas Instruments, Inc. All rights reserved. Stellaris and StellarisWare are registered trademarks of Texas Instruments. ARM and Thumb are registered trademarks, and Cortex is a trademark of ARM Limited. Other names and brands may be claimed as the property of others.

Texas Instruments
108 Wild Basin, Suite 350
Austin, TX 78746
Main: +1-512-279-8800
Fax: +1-512-279-8879
http://www.luminarymicro.com
Table of Contents

Introduction ......................................................................................................................................................... 4
Building the Car .................................................................................................................................................. 4
Circuit Board Design ........................................................................................................................................ 5
Overview of the Software .................................................................................................................................. 6
  Normal Running Mode .................................................................................................................................. 7
  Diagnostic Mode ............................................................................................................................................. 8
Schematics ......................................................................................................................................................... 9
Conclusion ........................................................................................................................................................ 11
References ....................................................................................................................................................... 11
Introduction

The Stellaris® LM3S316 Microcontroller Autonomous Car is a robotic vehicle that is capable of roaming around a controlled but random area and avoiding colliding with obstacles. The car makes use of the peripherals of the LM3S316 microcontroller including the Analog-to-Digital Converter (ADC) module which is used to read the infrared ranging sensors, and the pulse-width modulator (PWM) block which is used to drive the motors. This application note describes the construction of the car, the design of the circuit board, and the software that runs the vehicle.

Building the Car

When building a robotic vehicle, the mechanical construction is typically one of the most complex parts of the task. Some robotics sites on the internet, such as Lynxmotion (www.lynxmotion.com), provide ready-to-assemble vehicle chassis kits. The LM3S316 car uses the Terminator Sumo Kit from Lynxmotion, which contains an 8 inch x 8 inch chassis (20 cm x 20 cm), four brushed DC motors, a Lexan body with aluminum supports, and a convenient cavity inside the vehicle for mounting the battery.

Using this kit, an initial prototype was created that uses the Stellaris Family Development Board. The additional components required (such as the dual H-bridge chip) were breadboarded onto the development board and a working prototype was created. This prototype was used as a proof-of-concept for the entire design as well as a test vehicle to use for software development while the custom circuit board was designed.

The car's mechanical design uses three GP2D12 infrared ranging sensors to determine the distance to obstacles in three directions; forward, forward left, and forward right. The two side sensors are mounted at an angle of approximately 30 degrees from straight forward. One sensor property that must be accounted for is the range of uncertainty for distances closer than 4 inches (10 cm). For example, the sensor provides the same output for an object that is 3 inches away as one that is 6 inches away. To avoid problems caused by this phenomenon, the sensors are mounted 4 inches from the front of the chassis; therefore, the car would have to drive through a wall (which is not physically possible) before that wall can enter the range of uncertainty.

The infrared sensors also provide more accurate results, especially at the edge of a wall, when mounted vertically and so a lucite “windshield” was created to mount the sensors. The windshield is mounted across the chassis halfway between the front and back, which places the sensors approximately four inches from the front edge of the vehicle.

In addition to the sensor mounts and windshield, the following were also added to the vehicle:

■ A large push button was mounted to the side of the circuit board. This button is much easier to press when the car is running, and helps provide some static insulation for the board (versus attempting to press a small push button that is mounted on the circuit board itself).

■ A pair of mini-banana plugs was also mounted to the side of the circuit board. This allows the internal battery to be recharged without having to extract it from the vehicle

Note: The switch on the board must be turned off so that the charger is not also powering the circuit board while the internal battery is being recharged.
The end result is a self-contained, autonomous vehicle with a Stellaris microcontroller for a brain as shown in Figure 1.

**Figure 1. Stellaris LM3S316 Autonomous Car**

---

**Circuit Board Design**

There were several design considerations that drove the design of the circuit board. First, the board needed to run from the 7.2 V battery pack that is used to power the 7.2 V motors on the vehicle. The board also needed to provide interfaces to the three infrared ranging sensors, the four brushed DC motors, the on-chassis push button, the four LEDs (headlights and taillights), and a compact JTAG debug interface (the standard 20-pin JTAG header is too big). The final design consideration is that the board needed to fit into the roughly 2.5 inch (6.4 cm) square area near the back of chassis that was available for the circuit board.

With these goals in mind, a pair of power regulators were placed on the board. One power regulator generates the 5 V required by the infrared ranging sensors and the dual H-bridge chip, and the other generates the 3.3 V required by the Stellaris microcontroller and all the other control electronics.

An STL298 dual H-bridge chip was placed on the board for driving the brushed DC motors. The motors are paired, with one H-bridge driving the two motors on the left side of the vehicle and the other H-bridge driving the two motors on the right side of the vehicle. The inputs to each bridge are simple; if both inputs are the same, then the motor is in a dynamic braking mode. If the inputs are different, then the motor spins either forward or backward, depending on which input is logic high and which is logic low. Providing a PWM signal to the logic high input allows the speed of the motor to be controlled; the four inputs to the STL298 (two per H-bridge) are connected to the four PWM outputs of the LM3S316 microcontroller. A keyed 4-way connector was placed on the board for connection to the wiring harness added to the motors on the chassis; the keying helps to prevent plugging the motors into the board incorrectly.
The ranging sensors only require 5 V power, ground, and a connection to an analog input. The LM3S316 microcontroller has a four-channel ADC, so three of the channels are used to sample the analog value generated by the three ranging sensors. A block of 0.1 inch 1 x 3 stake headers are provided for connecting to the corresponding connector on the infrared sensors.

The headlight and taillight LEDs are similarly simple. They require only a GPIO on one side of the LED and a current-limiting pull-up resistor on the other side. A block of 0.1 inch, 1 x 2 stake headers are provided for connecting to the corresponding connector that was added to the LEDs (along with a length of wire so that the LEDs could be mounted at the front and back of the car). The LEDs are connected to GPIOs that can also be configured as PWM outputs from the timer blocks. This allows the LEDs to be driven by the PWM outputs to provide variable brightness control.

A small push button switch was placed on the board, along with a 0.1 inch, 1 x 2 stake header to allow an external push button switch to be installed as well. The push button does not have any hardware debouncing circuitry (such as a capacitor), so debouncing of the switch input is left entirely to software.

In lieu of the standard 20-pin JTAG connector (of which almost half of the pins are dedicated to ground connections), a more compact, but non-standard, 8-pin JTAG connector was created. This has the four required JTAG signals, chip reset, power, and ground. In order to use a JTAG emulator, a custom 20-pin to 8-pin conversion cable is required.

The board also includes a photocell connected to one of the analog comparators on the LM3S316 microcontroller which allows the headlights to be turned on automatically when it gets too dark, and a current sense amplifier on the bridge current connected to the fourth ADC channel which allows monitoring of the bridge current.

Given these on-board components, the design easily fits onto a 2.5 inch (6.4 cm) square board. All of the connectors for off-board items are placed along the edges and utilize right-angle connectors, keeping the profile low when the connectors are plugged in.

Figure 4 on page 10 shows the schematics for the circuit board.

**Overview of the Software**

The software for the LM3S316 microcontroller autonomous car is built in layers. Abstractions are provided for all of the hardware features (such as the brushed DC motors), and the main control code for the car uses those abstracted interfaces to obtain inputs and drive outputs. This simplifies the main control code since it does not need to know the details of the input and output devices operation, and allows the operation of the input and output devices to be enhanced (or even changed to a completely different type of device) without affecting the main control loop. Figure 2 shows the software block diagram.
Abstract interfaces are provided for the motors, sensors, push button switch, and lights. Each of these interfaces provides a set of functions for operating the device as required by the overall application (such as setting the speed of a motor or turning on a headlight). Not all hardware features of a device are abstracted; for example, the hardware allows the headlights and taillights to be set to a variety of brightnesses; the software abstraction does not support this feature.

The car supports two operating modes; normal running mode (see “Normal Running Mode” on page 7 for more information) and diagnostic mode (see “Diagnostic Mode” on page 8). Normal running mode, in which the car wanders around avoiding obstacles, is the typical operating mode for the car. Diagnostic mode methodically checks each device on the car to verify that the connectors are plugged into the board correctly and that everything is in working order.

When the car is not in normal running mode or diagnostic mode, it simply stays in place with the lights and motors turned off.

**Normal Running Mode**

When the car is in normal running mode, it exhibits one of the following four behaviors.

- The first behavior occurs when the car does not see any obstacles. In this case, the car simply drives straight forward.

- If the car sees an obstacle that is too close to the car (that is, within 20 inches, or 51 cm), the car continues to drive in a generally forward direction and veers away from the obstacle. The rate at which the car veers depends on the distance to the obstacle and the rate of approach to the obstacle; if the obstacle is still far away or is not approaching quickly, the car veers slowly. On the other hand, if the obstacle is fairly close and/or it is approaching quickly, the car veers quickly.

- If the car sees an obstacle that is approximately 9 inches (23 cm) away and appears to be running parallel to the car's path, the car attempts to follow the obstacle. It drives straight ahead, making minor adjustments to the left or right to keep the obstacle 9 inches (23 cm) from the side of the car.

- If the car gets extremely close to an obstacle (that is, within 7 inches, or 18 cm), or if a random time-out occurs, the car stops and spins in place. This continues until the obstacles are far away.
enough away to proceed forward without colliding with any obstacles. While the car is spinning in place, the taillights are turned on.

During all of these normal running mode behaviors, the car turns on its headlights when the ambient light conditions get too dark. When the ambient light gets brighter again, the headlights automatically turn back off again. This behavior is similar to the automatic dusk/dawn lights on automobiles.

A simplified version of the state machine for normal running mode is shown in Figure 3. The “Turning” state is actually eight distinct states; there is a set of four states that are used for turns that result from the car getting to close to an obstacle, and another four states that are used for random turns. “Turning” was drawn as a single state to simplify the diagram. Also, any of the states that are not the “Stop” state can transition to the “Stop” state by a press of the push button (these transitions were left off to simplify the diagram).

**Figure 3. Software State Machine Diagram**

Diagnostic Mode
Diagnostic mode uses a sequence of steps to verify that each functional piece of the car is in good working order. Diagnostic mode runs the steps as follows:

1. Turns on the headlights and taillights, but leaves the motors off. This step is mainly for photographic purposes; the car stays in place with all of the lights on, allowing for good still photography.

2. Blinks only the left headlight and runs only the left motors in the forward direction. This step shows that the left headlight is correctly connected, that the left motors are correctly connected, and that the left motors can be driven in the forward direction.

3. Blinks only the left taillight and runs only the left motors in the backward direction. This step shows that the left taillight is correctly connected, that the left motors are correctly connected, and that the left motors can be driven in the backward direction.

4. Blinks only the right headlight and runs only the right motors in the forward direction. This step shows that the right headlight is correctly connected, that the right motors are correctly connected, and that the right motors can be driven in the forward direction.
5. Blinks only the right taillight and runs only the right motors in the backward direction. This step shows that the right taillight is correctly connected, that the right motors are correctly connected, and that the right motors can be driven in the backward direction.

6. Blinks all four lights, runs all motors in the backward direction, and sets the speed of the motors based on the raw sensor reading from the left sensor, which can be seen by moving an object (such as a hand) closer and farther from the left sensor (with no effect being seen if the same is done to the right and center sensors). This step shows that the left sensor is correctly connected.

7. Blinks all four lights, runs all motors in the backward direction, and sets the speed of the motors based on the raw sensor reading from the right sensor, which can be seen by moving an object (such as a hand) closer and farther from the right sensor (with no effect being seen if the same is done to the left and center sensors). This step shows that the right sensor is correctly connected.

8. Blinks all four lights, runs all motors in the backward direction, and sets the speed of the motors based on the raw sensor reading from the center sensor, which can be seen by moving an object (such as a hand) closer and farther from the center sensor (with no effect being seen if the same is done to the left and right sensors). This step shows that the center sensor is correctly connected.

Any perceived problems with the operation of the car can be easily identified by running through the steps of diagnostic mode. If one of the steps fails to operate as expected (for example, Step 6, changing the motor speed based on the right sensor instead of the left sensor), then the wiring problem can be corrected and it is likely that the problem will go away. If all of the diagnostic steps pass, then the problem is likely with the code that handles the normal mode of operation.

When running, the car samples the sensors 500 times per second, samples the push button 200 times per second, and makes decisions 10 times per second. All processing occurs within interrupt handlers; the full application only requires .25 MIPS of processing power.

The application for the car uses approximately 6 KB of flash and 384 bytes of SRAM (256 bytes of which is the stack). Since the ADC is used by the application, and the ADC requires that the PLL be used, the processor is run at 12.5 MHz, which is the slowest that the processor can be clocked when running from the PLL. The .25 MIPS consumed by the application uses approximately 2% of the processor.

The software for the LM3S316 microcontroller autonomous car is provided in the accompanying ZIP file. The contents of this ZIP file should be extracted to the same directory to which StellarisWare™ was extracted. For example, if the StellarisWare™ ZIP file was extracted to C:\, then this ZIP file should also be extracted to C:\. This creates a C:\StellarisWare\AppNotes\an01245 directory that contains the source code for the car. See the an01245sw-nnn.pdf file in this directory (where nnn is replaced by a version number) for full details of the functions, global variables, and so on, in the source code.

**Schematics**

Figure 4 shows the schematics for the car.
Figure 4. Schematics for the LM3S316 Autonomous Car
Conclusion

The intelligent, autonomous car effectively demonstrates the use of the Stellaris LM3S316 microcontroller features, including the ADCs, analog comparators, timers, motion-control PWMs, and GPIOs. The vehicle software, written in easy-to-maintain C, shows how the 32-bit ARM® Cortex™-M3 core enables a straightforward software architecture that is completely interrupt-driven—an important consideration in product development as software development costs continue to balloon. While this is a fun microcontroller application, the concepts demonstrated here are directly applicable to products in many different applications.

References

The following are available for download at www.luminarymicro.com:

- Stellaris® LM3S316 microcontroller data sheet, Publication Number DS-LM3S316
- StellarisWare® Driver Library
- StellarisWare® Driver Library User’s Manual, publication number SW-DRL-UG
- Creating an Autonomous Car with the Stellaris® LM3S316 Microcontroller collateral, number an01245-nnn.zip (where nnn is the revision number)
Important Notice

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use. TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

<table>
<thead>
<tr>
<th>Products</th>
<th>Amplifiers</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Converters</td>
<td>amplifier.ti.com</td>
<td>Audio</td>
</tr>
<tr>
<td>DLP® Products</td>
<td>dataconverter.ti.com</td>
<td>Automotive</td>
</tr>
<tr>
<td>DSP</td>
<td><a href="http://www.dlp.com">www.dlp.com</a></td>
<td>Broadband</td>
</tr>
<tr>
<td>Clocks and Timers</td>
<td><a href="http://www.ti.com/clocks">www.ti.com/clocks</a></td>
<td>Digital Control</td>
</tr>
<tr>
<td>Interface</td>
<td>interface.ti.com</td>
<td>Medical</td>
</tr>
<tr>
<td>Logic</td>
<td>logic.ti.com</td>
<td>Military</td>
</tr>
<tr>
<td>Power Mgmt</td>
<td>power.ti.com</td>
<td>Optical Networking</td>
</tr>
<tr>
<td>Microcontrollers</td>
<td>microcontroller.ti.com</td>
<td>Security</td>
</tr>
<tr>
<td>RFID</td>
<td><a href="http://www.ti-rfid.com">www.ti-rfid.com</a></td>
<td>Telephony</td>
</tr>
<tr>
<td>RF/IF and ZigBee® Solutions</td>
<td><a href="http://www.ti.com/prf">www.ti.com/prf</a></td>
<td>Video &amp; Imaging</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Wireless</td>
</tr>
</tbody>
</table>

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2009, Texas Instruments Incorporated