Homework 5: Theory of Operation and Hardware Design Narrative

Team Code Name: Pop'em Drop'em Robots **Group No.** 21

Team Member Completing This Homework: Thomas Pansino

E-mail Address of Team Member: tpansino@purdue.edu

NOTE: This is the second in a series of four "design component" homework assignments, each of which is to be completed by one team member. The body of the report should be 3-5 pages, **not** including this cover page, references, attachments or appendices.

Evaluation:

SEC	DESCRIPTION	MAX	SCORE
1.0	Introduction	5	
2.0	Theory of Operation	20	
3.0	Hardware Design Narrative	20	
4.0	Summary	5	
5.0	List of References	10	
App A	System Block Diagram	10	
App B	Schematic	30	
	TOTAL	100	

Comments:

1.0 Introduction

Our project is an electronic version of the game "Rock'em Sock'em Robots", in which up to two players can control the movements of one of two boxing robots via a Microsoft Kinect. Player punches and dodges are translated into robot movements and hits recorded until either player's health bar value reaches zero. The game also features selectable battle music and a single-player mode in which the player competes against the computer.

Based on our component selections, our project has a number of circuit design constraints. Power regulation circuitry is needed to provide both a 5V and 3.3V power supply for various circuit components. This circuitry must also be tuned to the current needs of the components operating at either voltage. The project uses a fair number of LEDs and several motors, so appropriate mechanisms are needed for component driving and control. It includes an external motherboard operating at a different voltage level than the microcontroller, meaning that logic level translation must be performed on any communication channels between these two devices. Finally, the circuit design must contain various headers and connections to enable electronic components to be mounted away from the main PCB as well as for debugging purposes during PCB population.

2.0 Theory of Operation

Based on the block diagram in Appendix A, the circuitry of our project can be broken down into the following subsections: microcontroller and motherboard interfacing, solenoid power and control, stepper motor power and control, LCD interfacing, Hall-Effect sensor interfacing, LED control, and power regulation. The motherboard only controls the LED subsystem while the microcontroller handles all of the remaining subsystems mentioned above, with the exception of power regulation. Additional circuitry is not needed for interfacing and control of the Microsoft Kinect, the SD card, or the audio subsystem as these are all built-in capabilities and connections on the Raspberry Pi, our chosen motherboard.

Early in the planning stages of our project, we selected 5V as the primary operating voltage of our device because it is widely used and supported in digital circuits. An ability to operate at a supply voltage of 5V thus became one of the selection criteria for most of our components. The only two components which do not operate on a 5V power supply are the Raspberry Pi [1] and the TCS20DLR Hall-effect sensors [2], both of which use 3.3V logic. Since the Raspberry Pi can only operate at 3.3V and the microcontroller operates at 5V, logic level translation between the two devices is needed. The two devices communicate via UART protocol with a simple Request-to-Send (RTS) and Clear-to-Send (CTS) handshake for flow control. This choice of communication protocol requires four total pin connections between the devices, with two connections being made in either direction. The MAX3391 logic level translator [3] supports the conversion of the logic signals needed for the operation described by providing two sets of unidirectional logic conversion channels, with two in each direction. All signals coming to and from the Raspberry Pi are connected to 2x13 pin header which is be connected via ribbon cable to the PCB of the R-Pi.

Solenoid power and control for the four solenoids is accomplished through the use of four 4N33 optocouplers [4], each connected to one of four TIP120 NPN Darlington transistors [5] which are each in turn connected to a solenoid. A single digital input from a microcontroller GPIO pin activates each solenoid via this circuit. The optocouplers prevent any inductive kickback from damaging the pins or circuitry of the microcontroller while the Darlington transistors provide sufficient current gain to activate and drive the solenoid coils. Similar to the Raspberry Pi, the solenoids are connected to this circuitry via a 1x2 header cable which enables them to be mounted away from the PCB and to move with the bases of the robots.

The two stepper motors are controlled by a L297 motor controller [6] and a motor-driver chip that is built into the stepper motor that we selected. The motor controller requires GPIO connections for the microcontroller to provide a direction and a clock pulse line for each motor. With each pulse of the clock line, the motor moves one “step” in the direction indicated on the direction pin. The controller translates these two inputs into outputs for each of the stepper motor coils, designated A, B, C, and D. Like the solenoids, the steppers are connected to the PCB via a 1x4 header cable, allowing the motors to be mounted above the PCB and just under the fighting arena.

LCD interfacing with the two NHD-0216K1Z RGB backlit LCDs [7] is accomplished through the use of a single 74HC164 shift register [8] connected to the SPI interface of the microcontroller, along with several GPIO pins that control the LCD read/write mode, data/command register select, and the register write enable of each LCD. Commands are shifted out byte-by-byte to the two LCDs, both of which, are connected to the same shift register. The relevant LCD's control bits are then set and its register enable pulsed to store the new data in the LCD's internal control registers. The use of the SPI interface reduces the number of pins needed for the microcontroller to manage the LCDs, and each LCD has its own header so that it can be mounted on the front of the device packaging and connected via ribbon cable to the PCB, in a manner similar to that of the motors.

The Hall-effect sensors have only three pins each: a 3.3V power supply input, ground, and an open-drain output. According to [2], when the magnetic flux level detected by the sensor exceeds the $B_{ON}S$ threshold (typically 3.4 mT), the sensor output transitions from a high-impedance state to a logic low. It returns to a high-impedance state when the flux level detected falls below $B_{OFF}S$ (typically 2.0 mT). This allows us to wire several sensors to a single common output pin that is connected to the microcontroller and use a pull-up resistor to hold the pin high when all sensors are off. If any one of the sensors then detects a high enough magnetic flux value, the line will be pulled low. By placing several of these sensors inside the chest cavity of each robot and mounting small magnets inside each robot's fists, a robot's punches can be registered as "hits" against an opponent each time the opponent's Hall-effect sensor line goes low, and the opponents health bar value updated accordingly.

The SPI of the Raspberry Pi motherboard is connected to seven W2RF004RM LED controllers [9] linked serially. Each device contains its own internal shift register with an asynchronous pass-through for the input so that multiple devices can be linked serially, reducing fan-in while still enabling devices to receive commands in parallel with respect to each other. Each of these controllers is capable of driving 3 RGB-color LEDs, thus up to 21 LEDs can be controlled by this configuration and can be used both to indicate player positioning with respect to the Kinect and for general effect purposes. The individual LED outputs of each chip are accessed through the use of 40-bit commands, each of which specifies a chip address, output pin, maximum output brightness value, and a time interval in which the output should gradually fade between the maximum brightness and the off setting. Since the motherboard is also handling sound playback, giving it direct control of the LEDs, as opposed to control via commands sent over UART to the microcontroller, allows us to better synchronize light and sound effects and improve the overall user experience. The additional MAX3391 logic level translator is needed to convert the two 3.3V SPI signals from the Raspberry Pi to the 5V levels needed for the LED controllers.

Finally, two LM25576 buck converters [10] are used for voltage regulation, one each for 5V and 3.3V. Both receive input from a 12V switching power supply connected to a standard wall outlet and are tuned with passive component selection to produce proper output voltage values with low current ripple.

3.0 Hardware Design Narrative

Our project makes use of the following subsystems of the AT32UC3C2128C microcontroller: Serial Peripheral Interface (SPI), Universal Synchronous/Asynchronous Receiver Transmitter (USART), Joint Test Action Group interface (JTAG), and of course general-purpose input/output (GPIO). From the preliminary schematic, located in Appendix B, one can see on the left side of the microcontroller that the SPI system is used for interfacing with the LCDs. This was done primarily to reduce the number of pins required for LCD interfacing and to reduce programming overhead. On the bottom side of the microcontroller, the USART0 system is visible. It is used for inter-device communication between the microcontroller and Raspberry-Pi via the MAX3391 logic level translators. The USART systems is also used for general debugging purposes, as evidenced by the USART1 connection to a debugging header on the left side of the microcontroller. Near the top right corner of the microcontroller, one can see the pins JTAG interface connected to a header for circuit debugging after the microcontroller is soldered to our project's PCB. Finally, GPIO from the microcontroller is used for connection to various logic inputs and outputs in every subsystem that it controls. As previously stated, uses of the GPIO pins include stepper motor direction and pulsing, LCD register select, data latching, and chip enabling, and solenoid activation,

The power supply interfacing circuitry for the microcontroller was taken from page 5 of the 32-Bit AVR UC3 C Series Schematic Checklist [11]. The component values we selected were the same as those suggested in the checklist. Additionally, the checklist specifies that separate ground planes should be used for the digital and analog ground pins, a fact that will be taken into consideration during the PCB layout design phase. Note that the microcontroller does not perform any power-intensive tasks such as driving motor coils or several LEDs directly. All of the major power-using tasks are distributed over several different drivers and controllers, thus reducing the amount of current required to operate the microcontroller and the individual power consumption and heat dissipation needs of each component.

As mentioned in the previous section, the decision to interface the microcontroller with a Raspberry Pi requires the use of a logic level translator between the connections of the two devices. This requirement places somewhat of a practical limit on the number of connections that can be made due to the PCB space and passive component needs of each translator, as well as the overhead associated with soldering and connecting additional translators to the circuit. Also, the decision to use SPI to interface with the LCD as opposed to using GPIO adds the requirement of a shift register, which is needed to convert the serial data back into parallel data for use by the LCDs. Though the shift register requires no passive components and takes up very little PCB space, it is nevertheless an extra component required as a result of the decision to use the microcontroller's SPI. In the end though, we believe that the advantages SPI provides make it worth adding this component to our design.

4.0 Summary

This report discussed the various circuit design considerations of our project, along with design decisions and rationale used in creating the preliminary design. Topics discussed include power supply design, microcontroller and motherboard interfacing, motor power and control, LED control, LCD control, and overall device operation at the hardware level.

5.0 List of References

[1] "RPi Low-level peripherals". Embedded Linux Wiki [Online]. Available: http://elinux.org/RPi_Low-level_peripherals. [Accessed: Feb. 14, 2013]. (Link)

[2] Toshiba, "CMOS Digital Integrated Circuit Silicon Monolithic - Digital Output Magnetic Sensor," TCS20DLR datasheet, Feb. 2011. (Link)

[3] Maxim Integrated, "MAX3372E–MAX3379E/ MAX3390E–MAX3393E ±15kV ESD-Protected, 1μA, 16Mbps, Dual/Quad Low-Voltage Level Translators in UCSP," MAX3391 datasheet, Nov. 2011. (Link)

[4] Vishay Semiconductor, "Optocoupler, Photodarlington Output, High Gain, with Base Connection," 4N33 datasheet, 2 Oct. 2012. (Link)

[5] Fairchild Semiconductor, "TIP120/TIP121/TIP122 NPN Epitaxial Darlington Transistor," TIP120 datasheet, Oct. 2008. (Link)

[6] STMicroelectronics, "STEPPER MOTOR CONTROLLERS," L297 datasheet, n. d. (Link)

[7] Newhaven Display International, "NHD−0216K1Z−NS(RGB)FBW−REV1 Character Liquid Crystal Display Module," NHD−0216K1Z−NS(RGB)FBW−REV1 datasheet, 29 Nov. 2011. (Link)

[8] Texas Instruments, "CD54HC164, CD74HC164, CD54HCT164, CD74HCT164 High-Speed CMOS Logic 8-Bit Serial-In/Parallel-Out Shift Register," CD74HC164 datasheet, Aug 2003. (Link)

[9] Omron, "LED Control IC W2RF004RM," W2RF004RM datasheet, n. d. (Link)

[10] Texas Instruments, "LM25576/LM25576Q SIMPLE SWITCHER® 42V, 3A Step-Down Switching Regulator," LM25576 datasheet, Mar. 2009. (Link)

[11] Atmel, *AVR32768: 32-Bit AVR UC3 C Series Schematic Checklist* . Atmel Corporation, pp. 5. (Link)

Appendix A: System Block Diagram

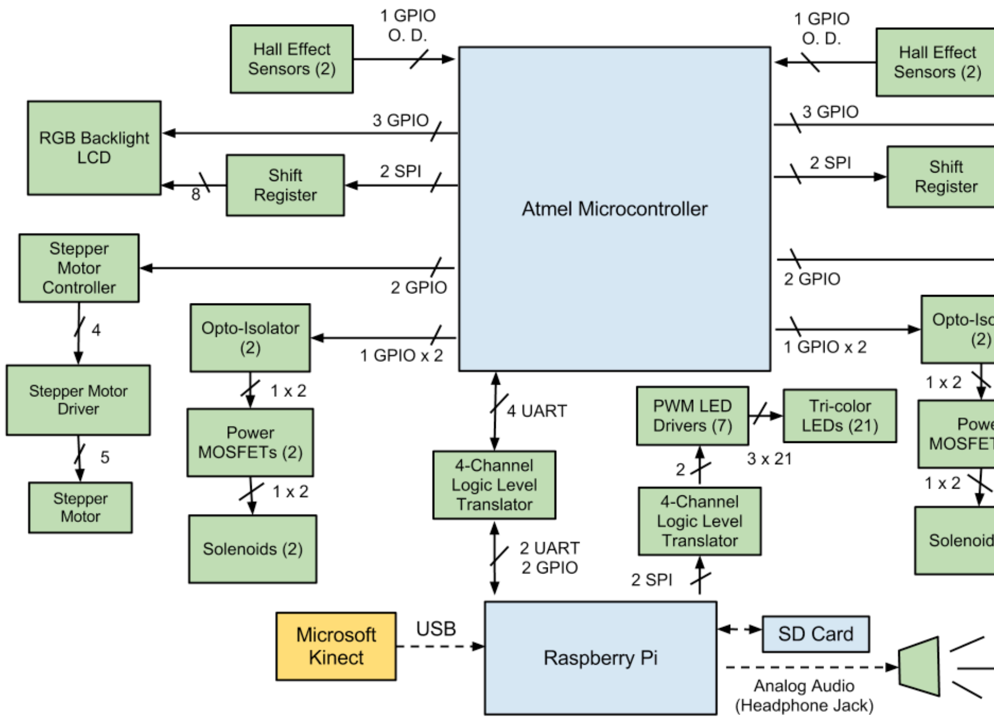

Figure 1: Block Diagram

Appendix B: Schematic

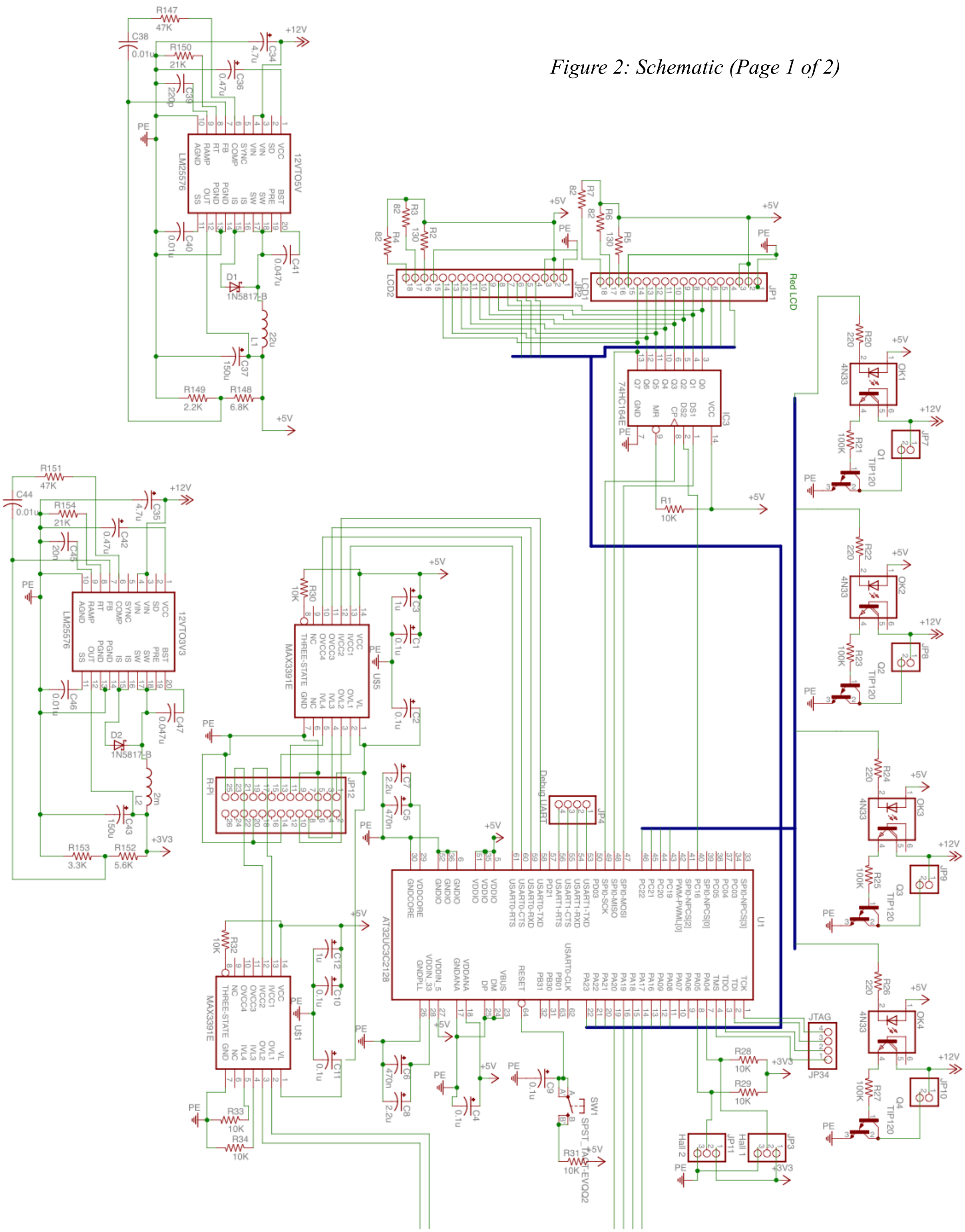

Figure 2: Schematic (Page 1 of 2)

Figure 3: Schematic (Page 2 of 2)