Laser Tag Game System: A Microcontroller-based Design

Design and Implementation of a simple Laser Tag game system using microcontrollers that communicate with each other through laser and with a central computer server through Wifi.

Hussam Ashraf El-Araby
Menna Yehia
Mostafa Konsowa
Youssef Gaber
Ziad Osama

20th December 2015

Project Overview:
- What is Laser Tag?
- Laser Tag components:
- Functional Requirements:
- Non-functional Requirements:
- Market Value:

Hardware Design:
- Player Unit Block Diagram
- Server Block Diagram
- System components communication:
- Microcontroller Ports:

Software Design:
- System tasks:

Implementation:
- Hardware Design Notes, Problems and Changes:
- Software Implementation:
- Software Design Notes, Problems and Changes:
- Implemented Functional Requirements

Future Improvements:

Appendix A (Code):

Appendix B (Hardware Design):
- Laser Tag Schematic Design
- Laser Tag Board Design
- Laser Tag Bill of Materials

Project Overview:

The following report is about our semester project for the course “Microcontroller-based System Design” with Dr. Shaalan during Fall 2015. We designed and implemented a laser tag game system. It is based on a set of microcontrollers and a central game server.

Note: Any attached pictures or files are also available in their original formats in the submission folder containing this report.

What is Laser Tag?

Laser tag is a team or individual sport or recreational activity where players attempt to score points by tagging targets, typically with a hand-held infrared-emitting targeting device. Infrared-sensitive targets are commonly worn by each player and are sometimes integrated within the arena in which the game is played (Wikipedia).

Usually, a central server/entity exists that collects information about the game status, deaths, kills, and other game related information. However, some arenas lack a central entity and need to collect the player units after the game to calculate the scores.

In our project , we have a central computer server and several player units. The player units are microcontroller based and represent the laser shooting element (weapon) and the laser sensing element (vest). The score calculation element is the central game server (wifi connected computer).

Laser Tag components:

Hardware view:

1. Player units: Microcontroller + LCD + Laser Sensing Element + Laser Emitting Element + Wifi
2. Central game server: Computer + Wifi

Logical view:

1. Weapon
2. Vest
3. Player LCD
4. Scoreboard and configuration server

Functional Requirements:

The functional requirements are divided by system component (logical view)

1) **Weapon:**
    a) Weapon shoots bullets with information about the damage the bullet incurs and the information about the shooter.
    b) Weapon has current bullet count, maximum ammo, and number of magazines.
    c) Weapon has shoot button, reload button and change weapon button.
    d) Weapon is upgradable (according game mechanics).
2) **Vest:**
    a) Vest senses if shot and by whom and then automatically deducts the bullet damage from health.

b) Vest detects if player is killed by bullet (health reaches zero) and by whom.
c) Vest keeps track of current health value as well as setting a maximum health value.
d) Vest contains an LED to show other players whether the player is dead or alive.

3) **Player LCD:**
  a) LCD shows different information about weapon (as stated above)
  b) LCD shows different information about health (as stated above)
4) **Scoreboard and Configuration Server:**
  a) Scoreboard shows ranks, kills, death, and other statistics about all the players.
  b) Configuration server sets game mode (Free for all vs teams)
  c) Configuration server creates matches associating different players with a match being played.
  d) Configuration server initializes weapons, scoreboard, and other related items at start of the match.
  e) Configuration server has direct access to weapon and vest, used to upgrade the weapons, and other miscellaneous actions.
  f) Configuration server ends the matches and shows final results.

Non-functional Requirements:

1. **Responsiveness and High Performance**
   a. Response times should be small to the degree the game appears instant to humans
      i. Some processing can be done locally on the player unit instead of sending it to server, hence improving overall performance (less communications).
2. **Reliability**
   a. Any data exchanged with the server should be verified as correct using mechanisms such as CRC.
   b. Laser should be detected reliably at the specified maximum distance.
   c. Ambient light should not interfere with the laser detection.
   d. The game system should not loose information during the matches and the scoreboard should reflect the true performance of the players.
3. **Scalability**
   a. The system should be able to handle the maximum amount of players without any apparent performance issues.
   b. The server should be able to handle multiple concurrent matches if needed.
4. **Safety**
   a. The laser used should not significantly harm any of the players or spectators, if accidently shined into their eyes.

Market Value:

Laser tag is a fun game for people of a wide variety of different age groups. It would appeal to many Egyptians looking for alternative activities to do. The product, manufactured locally, would be of interest to many business owners, looking to run such an activity, but are overwhelmed by the costs of purchasing expensive equipment and building an arena. High-end laser tag equipment can cost as much as $61,800 for only 16 vests which is an extremely high price for business owners in Egypt (http://www.laser-tron.com/index.php/products/pricing-and-financing-options).

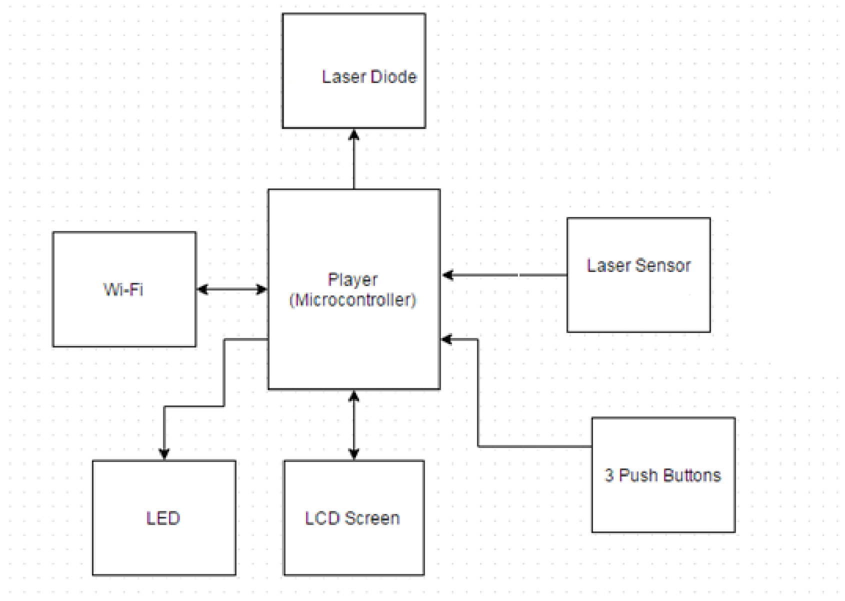
Hardware Design:

The player unit required hardware design. Our hardware design goal was to minimise cost and yet be able to achieve the requirements.

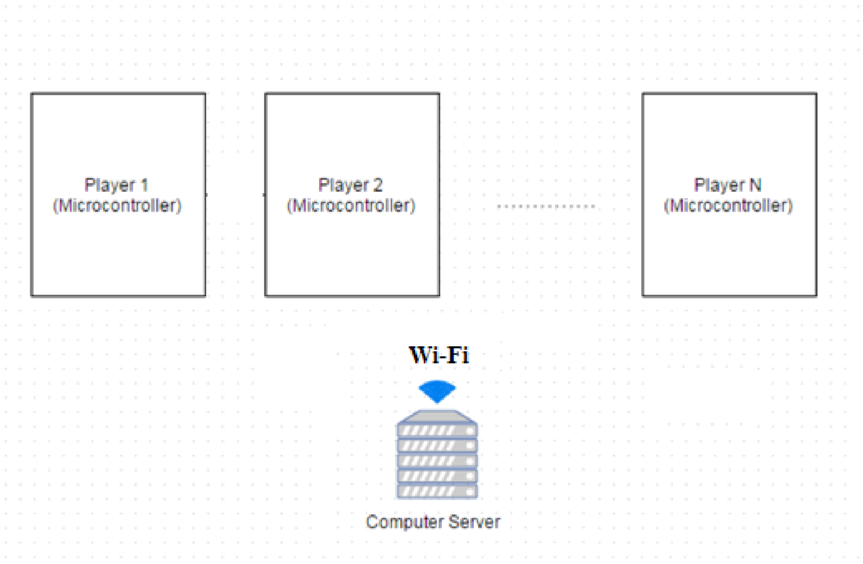
The circuit schematic and PCD board design of the player unit are attached in the end of this document in Appendix B.

The following diagrams describe the architecture of the system.

Player Unit Block Diagram



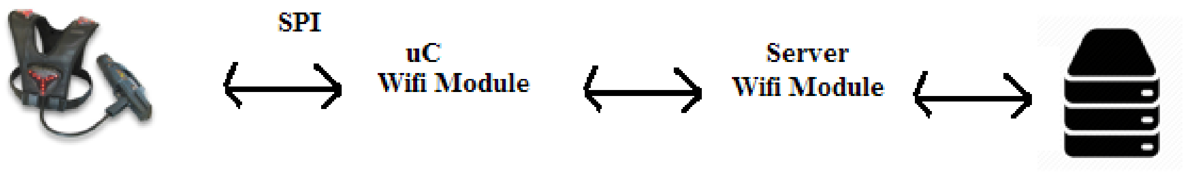
Server Block Diagram


Player 1 (Microcontroller)
Player 2 (Microcontroller)
Player N (Microcontroller)
Wi-Fi
Computer Server

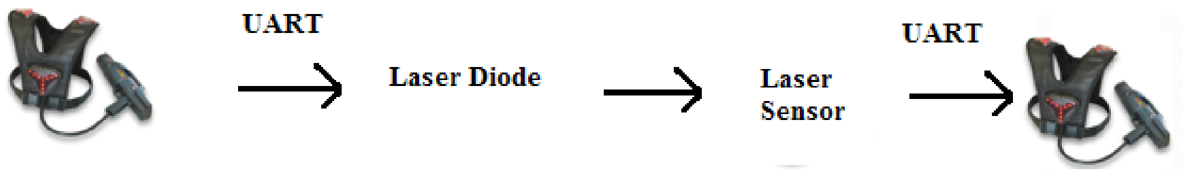
System components communication:

Two communication channels exist within the system.

Player ⇔Server



Player ⇔Player



Player ⇔Server Communication Protocol:

Player equipment and the server will communicate over TCP/IP.

Microcontroller Ports:

- Laser TX: 1 I/O Pin
- Laser Sensor RX: 1 I/O Pin
- LCD Screen: 8 Data Pins, 4 Control Pins
- WiFi: 4 SPI Pins + 1 Enable Pin + 1 Interrupt Pin
- LED: 1 I/0 Pin
- 3 Buttons: 3 I/0 Pins
- Minimum I/O Pins: 24 pins

Software Design:

The system is mainly an interrupt driven application.

System tasks:

1. Gun:

N	Task	Periodic/Aperiodic	Notes
1	**Trigger:** The fire button is pressed **Action:** Check if player has enough ammo. If yes: · Sends data (player ID + bullet damage) to UART to be sent as laser pulses to hit the target · Decrease the ammo by one. · Communicate ammo change to server. (task below) · Update LCD with ammo change (task below) If no: · Do nothing.	Aperiodic	
2	**Trigger:** The reload button is pressed **Action:** Check if player has enough ammo. If yes: · Reload the ammo. · Communicate ammo change to server. (task below) · Update LCD with ammo change (task below) If no: · Do nothing.	Aperiodic	
3	**Trigger:** The switch weapon button is pressed **Action:** Switch to another weapon Communicate ammo change to server (task below) Update LCD with weapon change (task	Aperiodic	

	below)		
4	**Trigger:** Data has been placed in UART transfer register, to be sent as laser pulses. **Action:** Interrupt the TX port of the UART · When data signal is 0, disable the PWM. · When data signal is 1, enable the PWM	Aperiodic	PWM to be initialized to have a duty cycle of 50%. It is enabled or disabled according to data signal

2. Vest

	Task	Periodic/Aperiodic	Notes
1	**Description:** Laser sensor continuously receives data (light) Demodulation of data is needed before passing the data to RX port of microcontroller	Periodic, 10KHz	Task of highest priority
2	**Trigger:** Bullet information resulting from data demodulation **Action:** Decrement the health, according to bullet damage info. Communicate health change to server (task below) Update LCD with health change (task below) If player is dead: · Turn off their LED on their vest · Communicate who killed player to server (task below)	Aperiodic	

3. Player LCD

1	**Trigger:** Information to display on LCD changed **Action:** Display is updated	Aperiodic	

4. Server Communication

	Task	Periodic/Aperiodic	Notes

1	**Trigger:** Server receives request from player to send data to server. **Action:** After establishing connection, server receives data from one player and identify the player through player ID Server has array of players, each player is a class which has private data members corresponding to the state of the player (health, weapon, ammo, score, …etc) Server updates scoreboard / other information.	Aperiodic, happens when server receives request	
2	**Trigger:** Server requests connection to send data to player for example weapon upgrade **Action:** After establishing connection, server send data to one player and identify the player through player ID Player unit updates itself using this data.	Aperiodic, happens when server send request	

Implementation:

We implemented the most important subset of the requirements for a demonstration on December 15th, 2015 (more details later on the implemented subset).
The implementation code is in Appendix A.

Hardware Design Notes, Problems and Changes:

One main problem that faced us while designing the hardware was the choice of the laser emitting and laser sensing elements.

The laser emitting element was hard to find because we wanted to find one that:

1. Worked on 3.3V-5V.
2. Could be controlled digitally (on/off modes only).
3. Emitted a powerful enough laser to be sensed from a reasonable distance.

The laser sensing element was hard to find because:

1. Most of the light sensors on the market worked with infrared not visible light laser.
    a. So it was difficult to choose both emitter and sensor of compatible frequencies.

Also it is useful to note that we implemented the project using a very small sensor. This does not reflect reality where the vest is actually has a very large surface area, but that is what we obtained to work with. Using a real vest later will require changes in the hardware design, but we assume the software would remain almost the same.

Software Implementation:

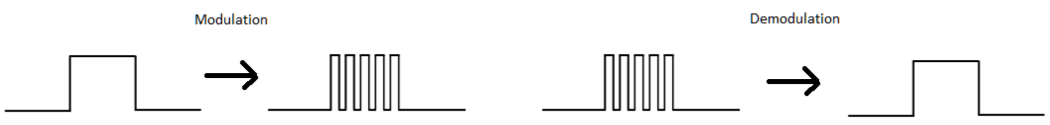
Our implementation is based on a set of classes to manage the scoreboard, beside establishing connections with more than one player and receiving and sending packets from and to each player. We have class *Player* to model each player with ID, health, ammos, shots, deaths, hits, status(alive or not). We also have classes *GameServerHandler* inheriting from *IServerHandler*, class *AshynchronousServer* and *Packet* to manage establishing connection and desterilizing data from packet.
We begin by instantiating objects from *GameServerHandler* and *AsynchronousServer* classer, the object of the latter class keeps listening on a predefined port until it receives a request for connection. The server then continuously call *showStatistics*() and *doAction*() methods in Round Robin fashion.

The *showStatistics*() method sorts the list of players according to their score, and displays for each player Player ID, alive/dead, Kills, Deaths, Kills/Deaths Ratio, Hit Ratio, Shots, Health.

The *doAction*() method executes the required action depending on the status of the player (dead/alive).
We also have class *BasicClient* to manage client code of sending packets to the server.

Software Design Notes, Problems and Changes:

To achieve reliability of laser sensing we intended to modulate the laser information sent by the sending end and later demodulate it at the receiving end.



Modulation/Demodulation process

The modulation/demodulation process is explained more here (http://www.schoolphysics.co.uk/age14-16/Wave%20properties/text/Modulation_and_demodulation/index.html).

Modulating the information sent (lower frequency) on a carrier signal (higher frequency) helps avoid the interference of ambient light (noise) with the laser signal.

The problem resides in the fact that the demodulation should occur all the time. The sensor cannot tell without demodulation the light entering it if a valid laser signal was sent and the contents of that signal.

The demodulation algorithm did not work on the microcontroller as we were not able to implement it sequentially within the code. It consumes most CPU cycles, blocking all the other tasks on the player unit.

It needed a separate parallel circuit, one which an FPGA can provide.

We also tried looking for hardware demodulation but due to lack of the required ICs, we just resorted to not modulating the laser signal. This affected the reliability significantly and the demonstration needed to work away from bright lights that will cause noise.

Moreover, we changed from an interrupt driven application to a round robin - state machine hybrid application. The reason for that was that we did not find that the application's performance would suffer using the round robin method as the human reaction time covers for any delays and would not become apparent. Advantage that round robin - state machine hybrid is a simpler implementation.

Implemented Functional Requirements

Due to limited time and pressure from the courses and upcoming finals we chose a subset to implement of the functional requirements. (check table below)

We chose the subset to showcase the most important parts of the system:

1. Component Communications (Player - Player and Player- Server)
2. Crucial Game Mechanics (Firing, Reloading, Kills, Deaths, and Scoring)

Regarding non-functional requirements, we assumed we achieved them to a sufficient level. Except for reliability due to the ambient light problem (explained above). Moreover, validating the non-functional requirements would require a larger deployment of the system and more real life use by game testers. All this was not available to us, cost wise or time wise.

Note: The index is the same as one written previously in functional requirements section.

Functional Requirement Index	Implemented?	Notes
1a	Partial	Bullet damage was not implemented, all bullets same damage.
1b	Yes	
1c	Partial	Change weapon button was not implemented, as we did not do primary/secondary weapons
1d	No	
2a	Partial	Related to 1a note. No specific bullet damages
2b	Yes	
2c	Yes	
2d	No	
3a	Yes	
3b	Yes	
4a	Yes	
4b	Partial	We only implemented a free for all mode.
4c	Yes	
4d	Yes	
4e	Partial	The server has the ability to initialize the players but a true direct access to all the player's weapon and vest information is missing.
4f	Yes	

Future Improvements:

Many parts of the project may be improved.

Such as:

1. Implementing the partially and not implemented functional requirements.
2. Merge the laser emitting code with the laser sensing code (ie. unify the player unit code).
3. More deployment and testing to assert that non-functional requirements are achieved.
4. Refactor the code to make it more extensible rather than hard coded.
5. Re-evaluate whether round robin - state machine hybrid used is sufficient or is there a better method.

Appendix A (Code):

Note: Refer to the files included in the submission.

Explanation of code arrangement:

Inside the “Demo Code” folder:

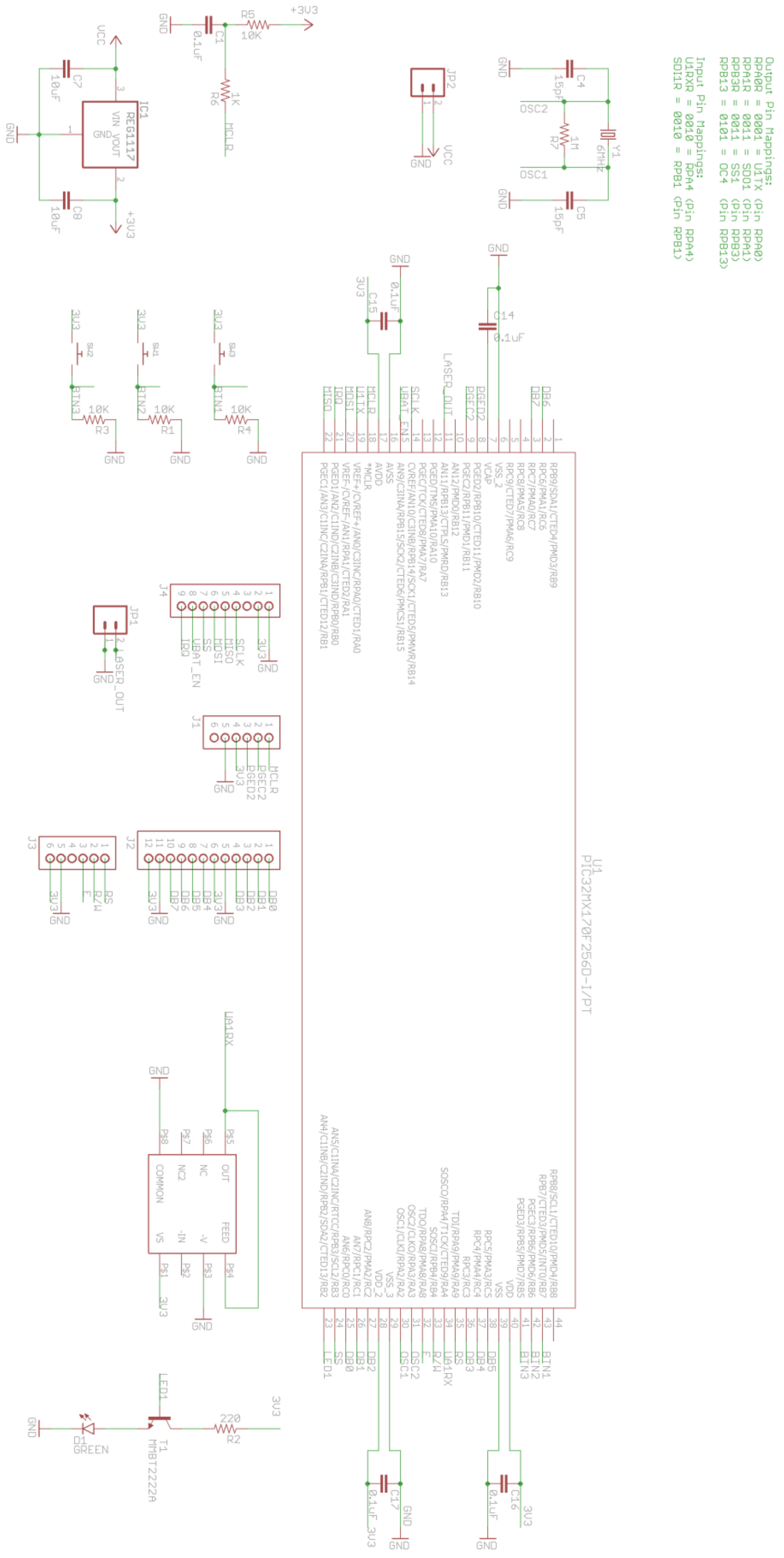
- “BasicServer” folder
    - C# code that implements the game server and server functions.
    - Classes used, include:
        - Event Type: class to describe all the possible game events
        - Packet: class to describe the packet sent/received
        - Player: class to describe the player
        - Other classes and functions.
- “Laser” folder
    - C code that implements the laser emitting functions.
- “Sensor” folder
    - C code that implements the laser sensing functions.

Appendix B (Hardware Design):

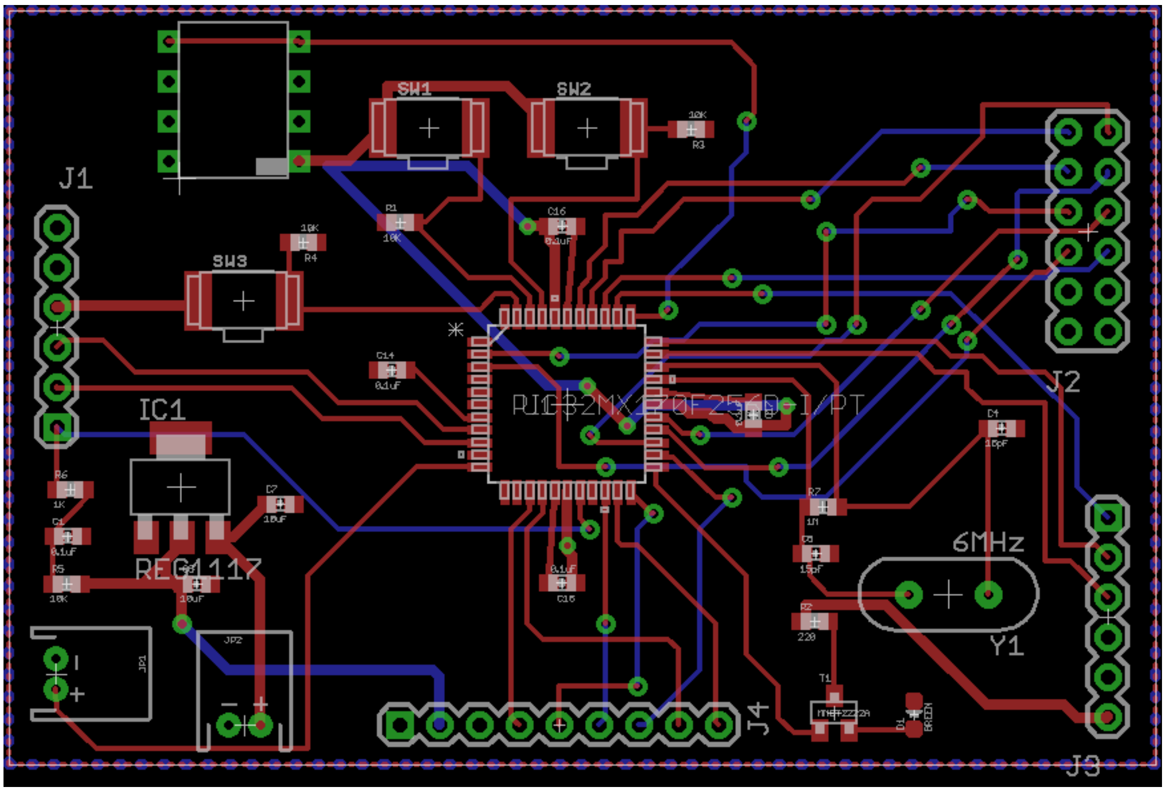
Attached below is:

1. Eagle Schematic Design (.sch)
2. Eagle Board Design (.brd)
3. Bill of Material (BOM) (.xlsx)

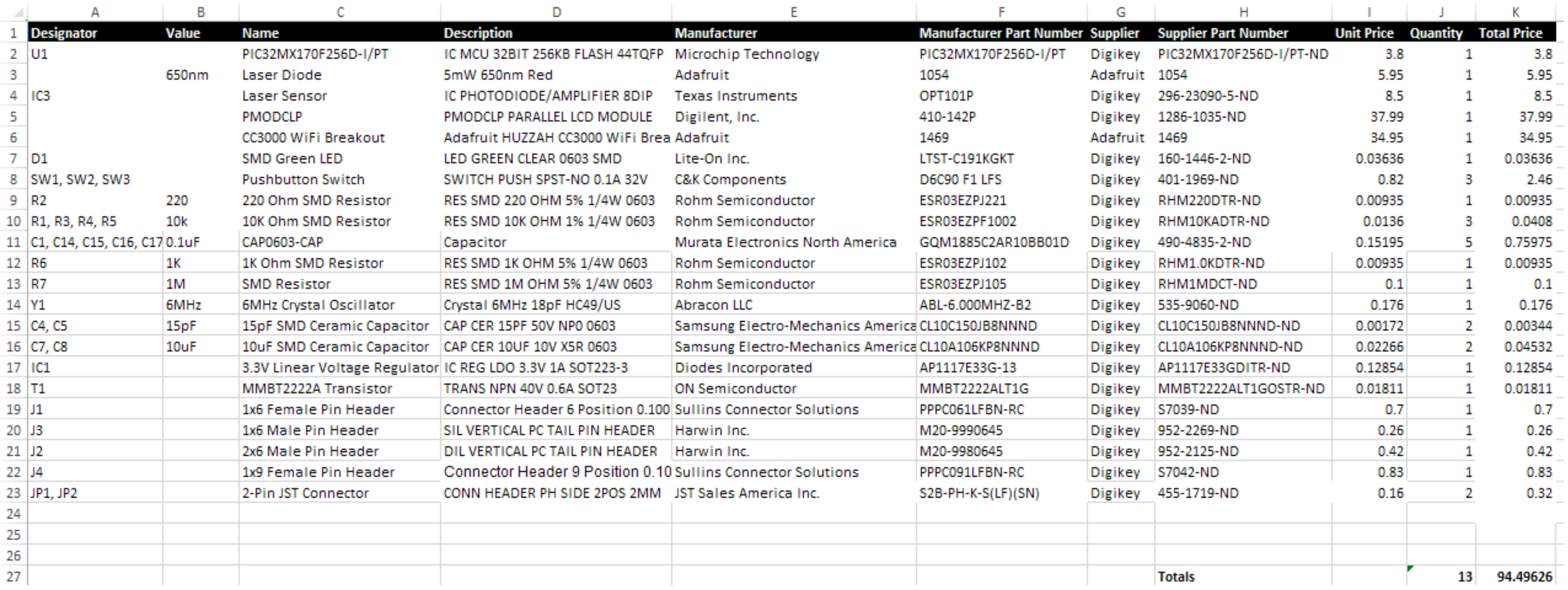
Laser Tag Schematic Design



Laser Tag Board Design



Laser Tag Bill of Materials



	A	B	C	D	E	F	G	H	I	J	K
1	Designator	Value	Name	Description	Manufacturer	Manufacturer Part Number	Supplier	Supplier Part Number	Unit Price	Quantity	Total Price
2	U1		PIC32MX170F256D-I/PT	IC MCU 32BIT 256KB FLASH 44TQFP	Microchip Technology	PIC32MX170F256D-I/PT	Digikey	PIC32MX170F256D-I/PT-ND	3.8	1	3.8
3		650nm	Laser Diode	5mW 650nm Red	Adafruit	1054	Adafruit	1054	5.95	1	5.95
4	IC3		Laser Sensor	IC PHOTODIODE/AMPLIFIER 8DIP	Texas Instruments	OPT101P	Digikey	296-23090-5-ND	8.5	1	8.5
5			PMODCLP	PMODCLP PARALLEL LCD MODULE	Digilent, Inc.	410-142P	Digikey	1286-1035-ND	37.99	1	37.99
6			CC3000 WiFi Breakout	Adafruit HUZZAH CC3000 WiFi Brea	Adafruit	1469	Adafruit	1469	34.95	1	34.95
7	D1		SMD Green LED	LED GREEN CLEAR 0603 SMD	Lite-On Inc.	LTST-C191KGKT	Digikey	160-1446-2-ND	0.03636	1	0.03636
8	SW1, SW2, SW3		Pushbutton Switch	SWITCH PUSH SPST-NO 0.1A 32V	C&K Components	D6C90 F1 LFS	Digikey	401-1969-ND	0.82	3	2.46
9	R2	220	220 Ohm SMD Resistor	RES SMD 220 OHM 5% 1/4W 0603	Rohm Semiconductor	ESR03EZPJ221	Digikey	RHM220DTR-ND	0.00935	1	0.00935
10	R1, R3, R4, R5	10k	10K Ohm SMD Resistor	RES SMD 10K OHM 1% 1/4W 0603	Rohm Semiconductor	ESR03EZPF1002	Digikey	RHM10KADTR-ND	0.0136	3	0.0408
11	C1, C14, C15, C16, C17	0.1uF	CAP0603-CAP	Capacitor	Murata Electronics North America	GQM1885C2AR10BB01D	Digikey	490-4835-2-ND	0.15195	5	0.75975
12	R6	1K	1K Ohm SMD Resistor	RES SMD 1K OHM 5% 1/4W 0603	Rohm Semiconductor	ESR03EZPJ102	Digikey	RHM1.0KDTR-ND	0.00935	1	0.00935
13	R7	1M	SMD Resistor	RES SMD 1M OHM 5% 1/4W 0603	Rohm Semiconductor	ESR03EZPJ105	Digikey	RHM1MDCT-ND	0.1	1	0.1
14	Y1	6MHz	6MHz Crystal Oscillator	Crystal 6MHz 18pF HC49/US	Abracon LLC	ABL-6.000MHZ-B2	Digikey	535-9060-ND	0.176	1	0.176
15	C4, C5	15pF	15pF SMD Ceramic Capacitor	CAP CER 15PF 50V NP0 0603	Samsung Electro-Mechanics America	CL10C150JB8NNND	Digikey	CL10C150JB8NNND-ND	0.00172	2	0.00344
16	C7, C8	10uF	10uF SMD Ceramic Capacitor	CAP CER 10UF 10V X5R 0603	Samsung Electro-Mechanics America	CL10A106KP8NNND	Digikey	CL10A106KP8NNND-ND	0.02266	2	0.04532
17	IC1		3.3V Linear Voltage Regulator	IC REG LDO 3.3V 1A SOT223-3	Diodes Incorporated	AP1117E33G-13	Digikey	AP1117E33GDITR-ND	0.12854	1	0.12854
18	T1		MMBT2222A Transistor	TRANS NPN 40V 0.6A SOT23	ON Semiconductor	MMBT2222ALT1G	Digikey	MMBT2222ALT1GOSTR-ND	0.01811	1	0.01811
19	J1		1x6 Female Pin Header	Connector Header 6 Position 0.100	Sullins Connector Solutions	PPPC061LFBN-RC	Digikey	S7039-ND	0.7	1	0.7
20	J3		1x6 Male Pin Header	SIL VERTICAL PC TAIL PIN HEADER	Harwin Inc.	M20-9990645	Digikey	952-2269-ND	0.26	1	0.26
21	J2		2x6 Male Pin Header	DIL VERTICAL PC TAIL PIN HEADER	Harwin Inc.	M20-9980645	Digikey	952-2125-ND	0.42	1	0.42
22	J4		1x9 Female Pin Header	Connector Header 9 Position 0.10	Sullins Connector Solutions	PPPC091LFBN-RC	Digikey	S7042-ND	0.83	1	0.83
23	JP1, JP2		2-Pin JST Connector	CONN HEADER PH SIDE 2POS 2MM	JST Sales America Inc.	S2B-PH-K-S(LF)(SN)	Digikey	455-1719-ND	0.16	2	0.32
24											
25											
26											
27								Totals		13	94.49626