

I.F.F. (Identification Friend or Foe) System

By

Eric Meyers (emeyer7)

ECE 445 Individual Progress Report - Spring 2016

TA: Braedon Salz

March 28th, 2016

Project No. 11

Contents

1	Introduction	1
1.1	Overview	1
1.2	Personal Contribution	1
2	Design.	1
2.1	Overview	1
2.2	Circuit Schematics	1
2.2.1	Friendly Interrogator Unit	1
2.2.2	System PCB Schematic	3
2.2.3	PCB Sub-Schematics and Verification	4
2.2.4	Software	7
2.3	Conclusion	7
3	Ethics	8
	Reference	9

1 Introduction

1.1 Overview

An Infantry I.F.F. System is a difficult yet very interesting project to tackle and many approaches were considered during the brainstorming period. In the end, the team decided to devise a two-way communication system which notifies any "interrogator" that their weapon is pointing at a friendly target. This system is two pieces of equipment referred to as the "friendly interrogator unit" and the "friendly target unit", that any soldier may wear and mount to their weapon. Upon receiving a query, the friendly target must send back an encrypted message to the interrogator to verify that it is friendly.

1.2 Personal Contribution

The team purposefully split this project into two sub-units so that the work could be evenly divided amongst us. Throughout the past 8 weeks, I have been primarily working on the friendly interrogator unit along with the software for the MSP430 microcontroller.

The following accomplishments have been made since the Design Review for this project:

- Circuit Schematic & PCB for Friendly Interrogator Unit
- Circuit Schematic & PCB for Linx KH3 R.F. Receiver (SMD-to-Breadboard Adapter)
- Circuit Schematic & PCB for MSP430 (SMD-to-Breadboard Adapter)
- Microcontroller, Voltage Regulator, Crystal Oscillator, R.F. Equipment Ordered
- MSP430 Software Researched and Written (Unit Tests Included)

These above points will be expanded upon in the design section below.

2 Design

2.1 Overview

One may refer to the figure in the Design Review of this project labeled "Block Diagram of Friendly Interrogator Unit" to get a high-level understanding of how the friendly interrogator unit functions.

2.2 Circuit Schematics

2.2.1 Friendly Interrogator Unit

The circuit schematic for the friendly interrogator unit is broken into 5 primary modules for schematic viewing: the AAT1217 DC-DC step-up converter ^[1], the crystal oscillator^[2], the MCU^[3]/laser transmitter/LED indicator/power switches, and the R.F. receiver/decoder circuit ^[4]. These are displayed below.

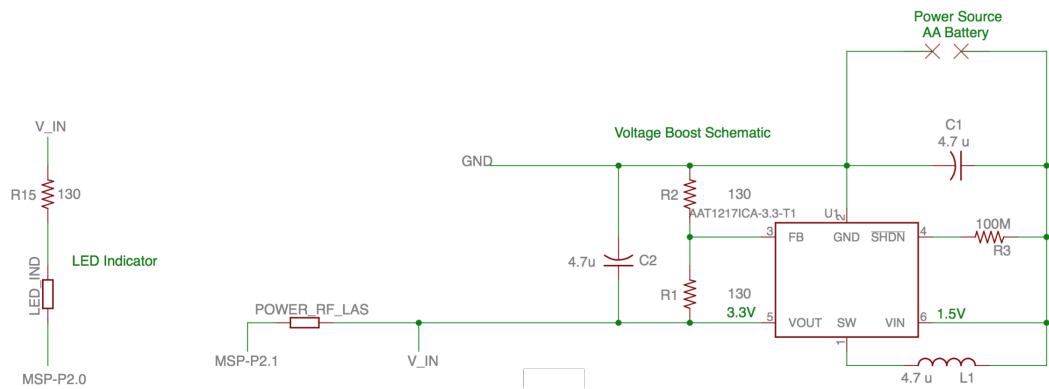


Figure 1: AAT1217 Circuit Schematic

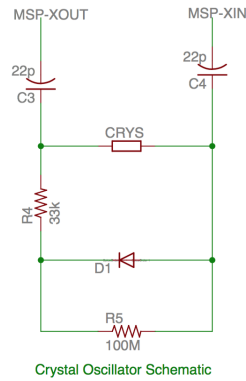


Figure 2: Crystal Oscillator Real Time Clock Circuit Schematic

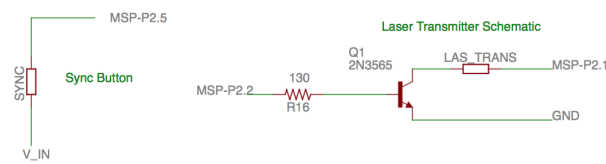


Figure 3: Sync Button and Laser Transmitter Circuit Schematic

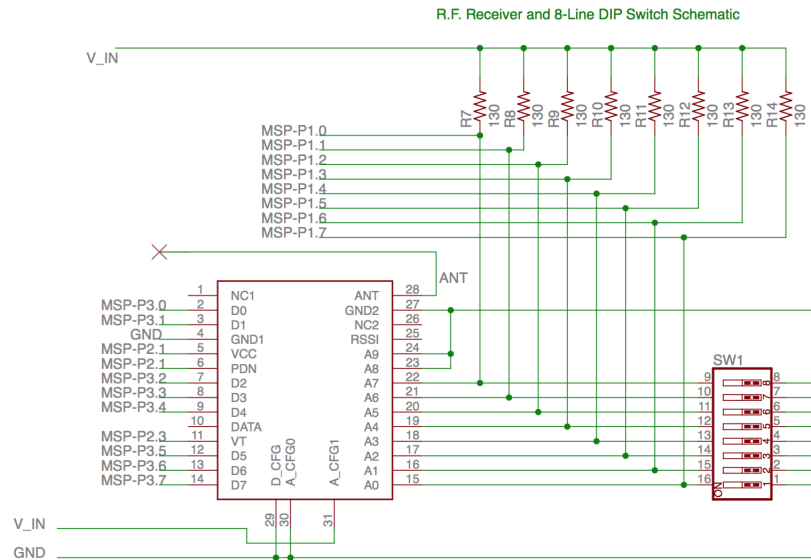


Figure 4: RF Receiver/Decoder and 8-Pin DIP Switch Schematic

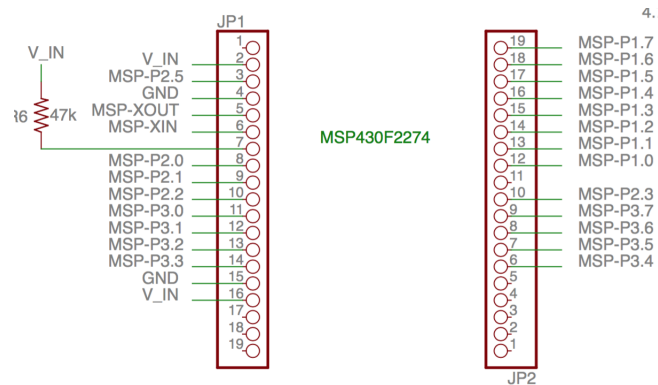


Figure 5: MSP430F2274 Schematic

2.2.2 System PCB Schematic

Shown below in Figure 6 is the entire PCB schematic of the friendly interrogator unit. This is the initial revision and will only be utilized for testing and debugging the system. There are 3 sub-schematics that will be expanded upon below.

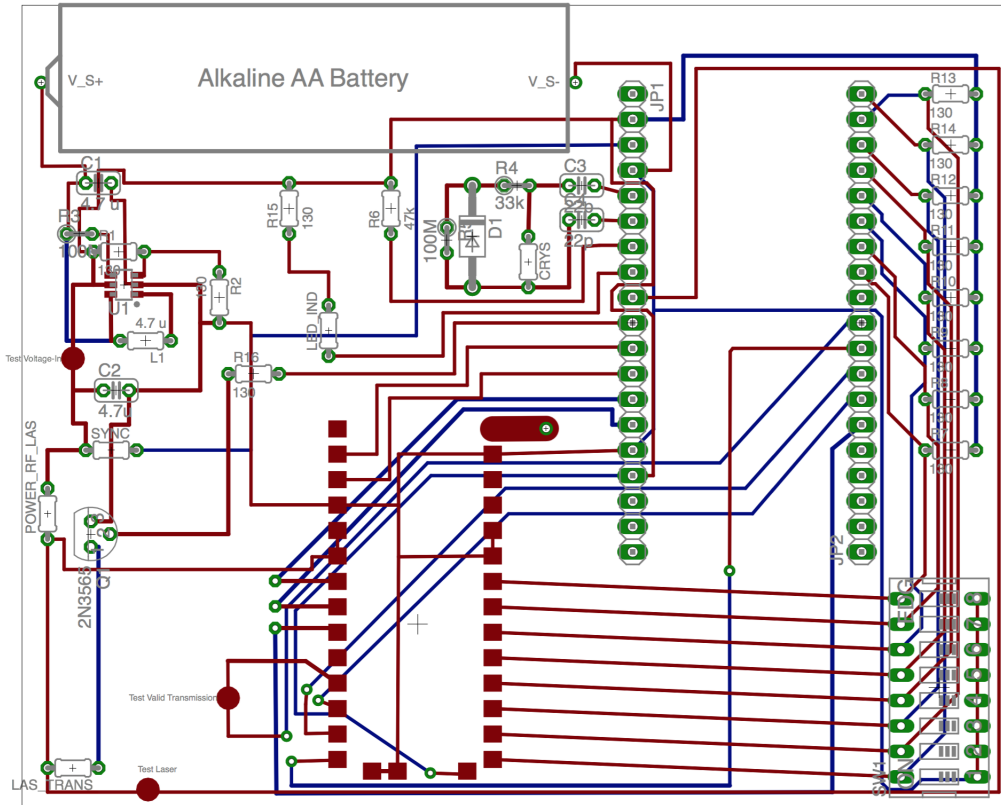


Figure 6: Overall System PCB Schematic

2.2.3 PCB Sub-Schematics and Verification

This section is not intended to review every single component on the PCB, but rather is meant to highlight some of the schematics I will utilize throughout the testing of the friendly interrogator unit.

The MSP430F2274 will act as the main processing/control unit for all peripheral inputs and outputs. All sub-schematics explained below will connect to the MSP430 in some fashion (either as input or output) and the particular connections can be seen in Figure 5. It should be noted that the MSP430 will not be soldered directly to the board, rather a "breakout" board will be utilized for programming/flashing the microcontroller (by using a breadboard with a FET Programmer). This will be used both for debugging stages and the final deliverable. This is shown below in Figure 7.

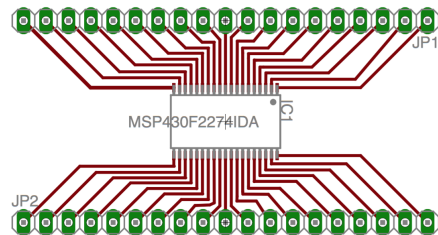


Figure 7: MSP430 TSSOP-to-DIP PCB Schematic

The voltage converter, the laser transmitter, and the R.F. receiver are the 3 sub-schematics that are of the utmost importance to the verification of the requirements. These are shown in Figure 8, 9, and 10 respectively. Above each schematic is an explanation of how the circuit will be used in the verification of the requirements for the friendly interrogator unit.

Voltage Regulator

The voltage regulator steps up a 1.5V AA battery source to 3.3V which is required to operate the MSP430, R.F. receiver, and the laser transmitter. The pad labeled as "Test Voltage-In" will be probed upon applying a 1.5V source to the top two vias labeled V_S+ and V_S-. In order to meet the requirements this voltage must output as 3.3V to supply the MSP430, R.F. receiver, and laser transmitter adequately.

This verification will take place prior to soldering anything onto the board (except for the AAT1217 voltage regulator itself) to ensure that no low/high voltages are being applied to the components when using them. Only upon verifying the output voltage ("Test Voltage-In") is 3.3V, will all other components be soldered onto the board to continue testing.

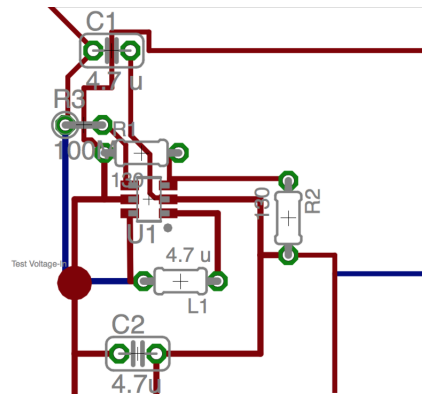


Figure 8: Voltage Converter PCB Schematic

Laser Transmitter

The laser transmitter will be necessary to pulse the unique I.D. of the interrogator to the friendly target unit. This will be done at a 40kHz rate and generated via interrupts on the MSP430. On the PCB, the pad labeled as "Test Laser" will be probed and connected to an oscilloscope to verify the rate of the signal being fed to the laser transmitter as 40kHz.

This verification will take place prior to connecting the actual laser transmitter to the board in order to confirm that the desired rate of operation is 40kHz. This step will require the MSP430 to be flashed and programmed with the correct software to generate interrupts on the MSP-P2.1 line. Once it is verified that the "Test Laser" signal can be triggered "high" at a 40kHz rate, then the software can be written to implement the main I.F.F. features. Please refer to section 2.2.4 for more details.

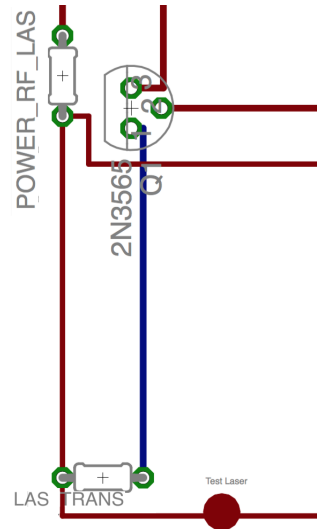


Figure 9: Laser Transmitter PCB Schematic

R.F. Receiver

Once the R.F. receiver gets a signal from the friendly target unit, it will set the V_T pin high. This will indicate to the MSP430 that new data is ready to be received from the KH3. Once the transmitter is set up properly by Noah, and the KH3 receiver has been properly connected/tested using a simple transmission on a breadboard, the receiver schematic can be tested (shown in Figure 10). This will be done by sending a signal on the transmitter and verifying the "Test Valid Transmission" pad is high on the PCB.

Another feature to note is the line running from the receiver to the connection for the antenna. This is a wider trace due to the antenna requiring an impedance match of $50\ \Omega$. This is explained in the Design Review in the Calculations section.

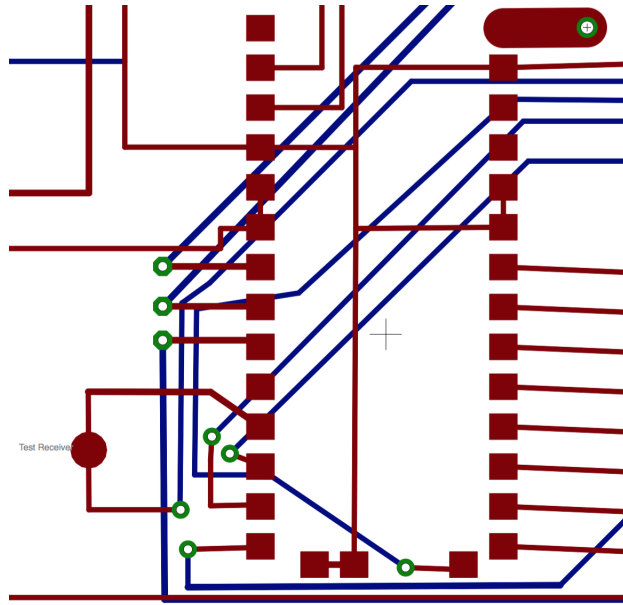


Figure 10: R.F. Receiver/Antenna PCB Schematic

The KH3 receiver first must be tested to verify that it is properly functioning before soldering to the friendly interrogator PCB. For this reason a breakout board to be used in conjunction with a breadboard was created for the testing phase. This is shown below in Figure 11.

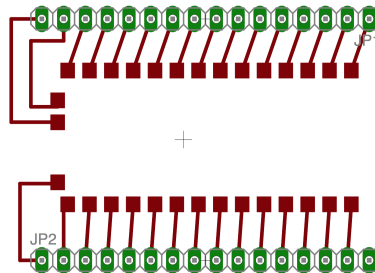


Figure 11: R.F. Receiver/Antenna PCB Schematic

2.2.4 Software

The software for the MSP430 is featured in the bottom Appendix section. Although this has not been debugged yet (for both its logic and syntax errors), the basic concept of flow is correct, and the comments explain what each line does. Some of the code written in this section was taken from the examples given online by T.I.^[5]

2.3 Conclusion

To summarize section 2, the necessary steps that must be taken over the next few weeks are as follows:

Upon receiving initial PCB:

1. Solder AAT1217 Voltage Converter to proper connection on PCB.
2. Apply 1.5V source voltage to V_{S+} and V_{S-}.
3. Verify 1.5V is being boosted to 3.3V by probing "Test Voltage-In" pad on PCB.
4. Connect and solder MSP430F2274 to MSP Breakout Board (Figure 7)
5. Flash MSP430 with MSP FET Programming Tool/breadboard with appropriate software to generate interrupts at 40kHz rate on P2.1.
6. Attach MSP430 to PCB and apply 1.5V to V_{S+} and V_{S-}.
7. Verify 40kHz rate by probing "Test Laser" with oscilloscope.
8. Connect Linx KH3 to breakout board and begin testing with KH3 transmitter (partnered task - with Noah)
9. Verify via the "Test Valid Transmission" line that a signal is properly being received from the transmitter.

There have been some minor setbacks over the past couple weeks. This was partially due to a misconnection on one of our receivers that caused a line to short, and the wrong part being ordered to connect from the MSP to a breadboard for testing. However the team has made this work up over the past week, ordered the correct parts and is still on pace to finish in time for final demonstration. There is still a lot of work to do in the next month, however, it is completely reasonable to finish this project in time (barring no major errors/setbacks).

I believe the work I have contributed relative to the project as a whole is completely fair and I believe the team as a whole has completed equal parts. Noah and I have been making good progress as a team and for this reason have a good chance to succeed in this project. As shown above, I have spent a significant amount of time developing PCBs/Schematics in Eagle and I have also spent a significant amount of time developing the software for the friendly interrogator side. All together, I believe I have put in well over the estimated 13 hours/week Noah and I came up with at the beginning of the semester. I believe I have been putting in anywhere between 15-25 hours/week (fluctuating).

3 Ethics

The team has taken a lot of consideration behind the ethics in this project. Noah and I devised our own ethics plans and came up with similar viewpoints. First and foremost, for obvious reasons, the team has not dealt or handled with any sort of weapons whatsoever. The IEEE code of ethics acts as a guideline to our ethics policies and are explained below:

1. *"To accept responsibility in making decisions consistent with the safety, health, and welfare of the public, and to disclose promptly factors that might endanger the public or the environment"*
9. *"To avoid injuring others, their property, reputation, or employment by false or malicious action"*

The team will obviously not deal with any sort of weapons during this project and will take strong pre-caution when handling the laser transmitter. The laser will only be a 5mW diode and Noah

determined that it will not harm anyone that is in a close distance. However, when dealing with any sort of laser diodes one must handle with caution and follow guidelines.

2. *To avoid real or perceived conflicts of interest whenever possible, and to disclose them to affected parties when they do exist*
3. *To be honest and realistic in stating claims or estimates based on available data*

These two points are very important to the academic integrity to this project and have been followed very strictly over the past month since the design review. We have both made an attempt to gather data for analysis and design that is as accurate as possible. In a system like this, it is imperative that numbers are precise. Also, I, along with Noah, have stayed very honest to the work we have performed and have cited our sources as necessary.

References

- [1] Skyworks, "Skyworks AAT1217 Micropower Step-Up Converter," Datasheet, accessed February 2016. [Online]. Available: http://www.skyworksinc.com/uploads/documents/AAT1217_202050B.pdf
- [2] "ECS3x8 32.768 kHz Tuning Fork Crystal Oscillators," Datasheet, accessed February 2016. [Online]. Available: <http://www.ecsxtal.com/store/pdf/ECS-3x8.pdf>
- [3] "Mixed Signal Microcontroller MSP430F22x4," Datasheet, Texas Instruments, accessed February 2016. [Online]. Available: <http://www.ti.com/lit/ds/symlink/msp430f2274.pdf>
- [4] "KH3 Series Receiver Module Data Guide," Datasheet, accessed February 2016. [Online]. Available: <https://www.linxtechnologies.com/resources/data-guides/rxm-xxx-kh3.pdf>
- [5] A. Dannenberg, "MSP430x22x4-ta-01.c." [Online]. Available: <http://www.ti.com/product/MSP430F2274/toolssoftware>

Appendix

```
//*****
// MSP430F2274 Infantry Identification Friend or Foe Software
// University of Illinois - ECE445 - Senior Design - Project #11
// Authors: Eric Meyers and Noah Prince
// Date(s): Initial Revision 3/15/2016
// -----
// FRIENDLY INTERROGATOR UNIT
//
// Description: This piece of software is meant to control the friendly interrogator unit
//              on the Infantry I.F.F. System. Upon connecting all hardware components correctly
//              this software will generate 40kHz interrupts to send a laser transmission, and
//              then proceed to poll for an acknowledgement from the friendly target unit.
//
// INPUT GPIO PINS : P1.0, P1.1, P1.2, P1.3, P1.4, P1.5, P1.6, P1.7 = R.F. Address Lines
//                  P3.0, P3.1, P3.2, P3.3, P3.4, P3.5, P3.6, P3.7 = R.F. Data Lines
```

```
//          P2.1 = Power Button for R.F. Receiver and Laser Transmitter
//          P2.5 = Sync Button to Reset/Sync Clocks
//
// OUTPUT GPIO PINS: P2.0 = LED Indicator (Friendly or Enemy)
//          P2.2 = Laser Transmitter
//
// Built with CCE Version: 3.2.2 and IAR Embedded Workbench Version: 4.11B
//*****
#include <msp430.h>
#include "pins.h"

#define PASSPHRASE 0x1234 //HARDCODED PASSPHRASE - 16 bits
#define UNIT_TEST_LASER_SIG 1

/* ==== Initialize Global Variables ==== */
int seconds = 0; //Seconds Timer

/* Main Function
 * Description: This main function performs all necessary tasks to take in the
 *             appropriate address lines, pulse the appropriate unique I.D. to the
 *             friendly target unit and wait for an acknowledgement. This "ack" from
 *             the target unit
 */
int main(void)
{
    /* ==== Initialize Local Variables ==== */
    int i = 0;                // loop variable
    int unique_id;            // unique i.d. to pulse on laser
    int ack_passphrase;       // local ackn passphrase to verify with received
    int received_ack_passphrase; // ack passphrase received from target unit
    char address;              // address line bits (condensed into a single char)
    char start;                // start bits (4 bits) - to signify to photo receiver on target
    char check_sum;            // check-sum bits (4 bits) - sum of all address bits

    /* Turn off Watch-Dog Timers */
    WDTCTL = WDTPW + WDTHOLD;

    /* ==== Set Direction of GPIO ==== */
    //PXDIRE - BITWISE : 1 OUTPUT 0 INPUT : source: page 329 MSP430x2xx */
    P1DIR |= 0xFFFF; // All pins on P1 (P1.0 - P1.7) are inputs - Address Lines
    P3DIR |= 0xFFFF; // All pins on P3 (P3.0 - P3.7) are inputs - Data Lines
    P2DIR &= (BIT0 | BIT2); //Set P2.0 & P2.2 as output - Laser Transmitter & Indicator
    P2DIR |= (BIT1 | BIT5); //Set P2.1 & P2.5 as input - Sync and Power to R.F. & Laser

    /* =====Laser Unique I.D. =====
    -----

```

```
| ST_B3 | ST_B2 | ST_B1 | ST_B0 | A7 ... A0 | CS3 | CS2 | CS1 | CS0 |
-----

with ST_B3 - ST_B0 = Start Bits (Logic Highs (1s))
A7 - A0 = R.F. Address Lines (Depending on 8-Pin DIP Switch)
CS3 - CS0 = CheckSum Bits (A7+A6+...+A1+A0)
=====*/

//Combine address into single char, set start bits to all high
address |= (P1IN & 0xFF);
start = 0x0F;

//calculate check-sum bits based off of how many ones in unique i.d.
for(; i < 8; i++) {
    check_sum += (address>>i) & BIT0;
}

//shift bits properly to set the unique_id upon starting
unique_id = (start << 12) | (address << 4) | check_sum;

/* ===== Setup Timers =====*/
//Setup Timer A1 to perform 1 Hz interrupts (for seconds counter) - 1s
char ten_sec_timer_a_flag = 0; //Validation period flag - 10 seconds
TAOCCRO = 32768; // Set count limit (32.768 kHz Clock = 32,768 ticks until one interrupt is
    registered)
TAOCCTLO = 0x10; //Enable counter interrupts - bit 4
TAOCTL = TASSEL_1 + MC_1; //Timer A0 with ACLK @ 32768Hz @ 3.0V(VERIFY), count up.

//Setup Timer B0 to do 40 kHz interrupts (for laser) - 25us
char laser_timer_b_flag = 0;
TBOCCRO = 100000; // Set count limit (16 MHz Clock = 160,000/100,000 = 160 ticks/sec)
TBOCCTLO = 0x10; //Enable counter interrupts - bit 4
TBOCTL = TBSSEL_2 + ID_2 + MC_1; //Timer A0 with SMCLK @ 160/4 = 40kHz @ 3.0V, cnt up

/*===== DEBUGGING - UNIT TEST FOR LASER SIGNAL - must be 40kHz */
#if UNIT_TEST_LASER_SIG
for (;;) {
    while (!laser_timer_b_flag); //wait for 40kHz signal
    laser_timer_b_flag = 0;      //reset flag
    P2OUT ^= 0x02;               //toggle P2.2 on MSP
}
#endif

/*=====*/

/* ==== Main Loop (Perform pulsing, poll for response?) ==== */
while (1)
{
    /* UPDATE ACKNOWLEDGEMENT PASSPHRASE */

```

```

if (ten_sec_timer_a_flag) {
    ten_sec_timer_a_flag = 0;
    ack_passphrase = PASSPHRASE | seconds; //NO ENCRYPTION YET
}

/* IF OPERATOR TURNED LASER/RECEIVER ON, BEGIN PULSING/CHECKING RECEIVED ACK*/
if (P2IN & BIT1) {
    /* PULSE UNIQUE ID, CHECK VALID TRANSMISSION SIGNAL*/
    for (i = 0; i < 16; i++) {
        while (!laser_timer_b_flag); //wait for 40kHz signal
        laser_timer_b_flag = 0;      //reset flag

        //Turn P2.2 ON or OFF depending on state of unique I.D.
        ((unique_id >> i) & BIT0) ? (P2OUT |= 0x02) : (P2OUT &= ~0x02);
    }

    /*CHECK R.F. RECEIVER VALID TRANSMISSION (V_T) SIGNAL*/
    if (P2IN & BIT3) {
        //Get 1st set of data bits from receiver on P3.0 - P3.7 - MSB
        received_ack_passphrase |= ((P3IN & 0xFF) << 8);

        for (i=0; i < 10000; i++); //DELAY - this probably will not work

        //Get 2nd set of data bits from receiver on P3.0 - P3.7 - LSB
        received_ack_passphrase |=((P3IN & 0xFF));

        //Verify with our local copy
        if (received_ack_passphrase == ack_passphrase)
            //TURN ON INDICATION LED!! (FRIENDLY TARGET IDENTIFIED)
            P2OUT |= BIT0;
    }
}

}

}

/* ISR : Timer_A3
 * Description: This timer acts as our "RTC" and increments the "seconds" variable every second,
 * and also depending on the value of "seconds" it sets the ten_sec_timer_a_flag.
 */
#pragma vector = TIMERO_A0_VECTOR
__interrupt void Timer0_A0 (void) {
    seconds++;
    ten_sec_timer_a_flag = (seconds % 10 == 0);
}

```

```
/* ISR : Timer_B3
* Description: This timer triggers the laser transmitter flag at 40 kHz (hopefully).
*/
#pragma vector = TIMER0_B0_VECTOR
__interrupt void Timer0_B0 (void) {
laser_timer_b_flag = 1; //Set the laser_timer_b_flag to 1
}
```
